CEC
3.6.0.0

用户接入——网页版客户端集成（RESTful）

文档版本	01
发布日期	2024-08-19

华为技术有限公司

华为技术有限公司

地址：	深圳市龙岗区坂田华为总部办公楼 邮编：518129
网址：	https://www.huawei.com
客户服务邮箱：	support@huawei.com
客户服务电话：	4008302118

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 概述

第三方系统可以通过本手册，学习使用网页客户端接入接口进行网页版的用户侧Chat聊天工具开发。

具体接口请参见《 接口参考 》中网页客户端接入相关接口。

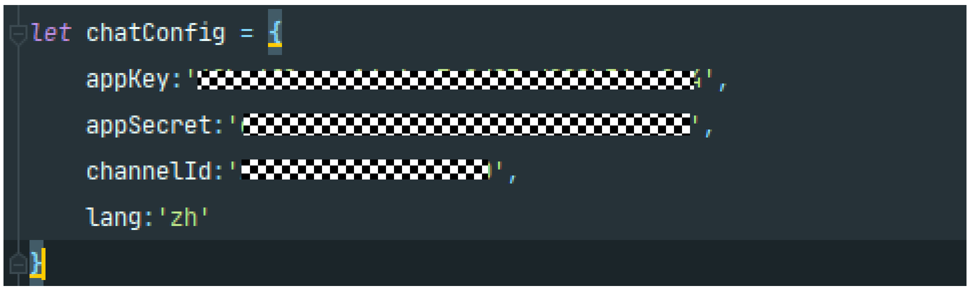
2 开发前准备

- 第三方已经向AICC申请了租户信息，系统运维管理员已经为第三方系统添加了租户信息。
- 获取demo包，请登录华为云AICC相关论坛下载。

3 鉴权方式

当前纯前端的Demo，鉴权信息均在前端文件中，若后续需要在正式的环境中使用，请将相关鉴权信息转移到服务端。

步骤1 验证使用的鉴权信息。找到目录中的文件：src/api/config.js ：

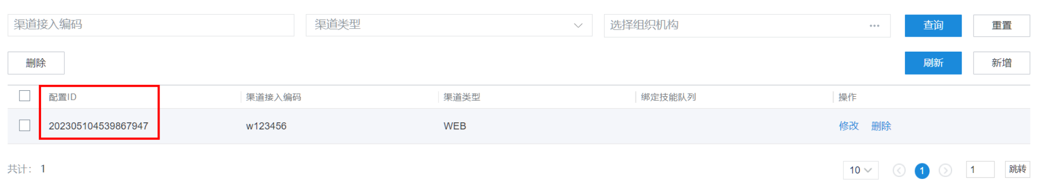


```
let chatConfig = {
    appKey:'[redacted]',
    appSecret:'[redacted]',
    channelId:'[redacted]',
    lang:'zh'
}
```

其中，appkey和appSecret为apifabric接口调用需要使用到的aksk，请联系运营人员获取。

channelId为需要对接的渠道ID，具体值来源请以租户管理员登录AICC，点击进入“配置中心>接入配置>渠道配置”中，如下：

图 3-1 渠道 ID



	配置ID	渠道接入编码	渠道类型	绑定技能队列	操作
☐	202305104539867947	w123456	WEB		修改 删除

共计：1

步骤2 前端鉴权。代码的路径 src/api/webChat.js，具体代码参考如下：

```
/**
 * 申请api-fabric的token
 *
 * @returns {Promise<void>}
 */
async applyToken() {
    if (this.appKey && this.appSecret) {
        let apiResult = await axios({
            url: '/apigovernance/api/oauth/tokenByAkSk',
            method: 'POST',
            headers: {
                'Accept': 'application/json',
                'Content-Type': 'application/json;charset=UTF-8'
            },
            data: {
                app_key: this.appKey,
                app_secret: this.appSecret
            }
        });
        if (apiResult.status !== 200) {
            return;
        }
        this.apiToken = apiResult.data['AccessToken'];
        if (this.userName) {
            await this.getMessageToken(true);
        }
        if (this.applyTask) {
            return;
        }
        //每10分钟刷新token
        this.applyTask = setInterval(()=>{
            this.applyToken()
        }, 10 * 60 * 1000);

    }
}
```

步骤3 修改成后端鉴权。

1. 在正式使用的场合中，不建议将appkey和AppSecret直接写在前台代码中，可以通过请求服务端返回apifabric生成的token。 可以参考后台代码如下：该代码会返回appkey和apifabric生成的token。

```
public class GetRequestTokenController {

    @Autowired
    RestTemplate restTemplate;

    @Value("${api.fabric.appKey}")
    String appKey;

    @Value("${api.fabric.appSecret}")
    String appSecret;

    @Value("${api.fabric.address}")
    String appAddress;

    @PostMapping("/getTokenAndAppKey")
    public JSONObject getTokenAndAppKey(){
        String token = getToken();
        JSONObject resp = new JSONObject();
        if(StringUtils.hasText(token)) {
            resp.put("token",token);
            resp.put("appKey",appKey);
        }
        return resp;
    }

    private String getToken(){
        JSONObject reqBody = new JSONObject();
        reqBody.put("app_key", appKey);
        reqBody.put("app_secret",appSecret);
        UriComponentsBuilder builder = UriComponentsBuilder.fromUriString(appAddress);
        ResponseEntity<JSONObject> responseEntity =
                restTemplate.exchange(
                        builder.build(true).toUri(),
                        HttpMethod.POST,
                        new HttpEntity<>(reqBody, null),
                        JSONObject.class);
        JSONObject response = responseEntity.getBody();
        if (response == null || response.isEmpty()) {
            return "";
        }
        return response.getString("AccessToken");
    }
}
```

2. 该代码为Springboot框架中的controller，请在配置文件中添加以下配置：

```
api.fabric:
  appKey: xxx
  appSecret: xxx
  appAddress: https://ip:port
```

3. RestTemplate的生成请参考以下代码：

```
import org.apache.http.conn.ssl.NoopHostnameVerifier;
import org.apache.http.conn.ssl.SSLConnectionSocketFactory;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClientBuilder;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.ssl.SSLContexts;
import org.apache.http.ssl.TrustStrategy;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.http.client.HttpComponentsClientHttpRequestFactory;
import org.springframework.web.client.RestTemplate;
import javax.net.ssl.SSLContext;
/**
 * HttpConfig
 *
 * @author x30005416
 * @since 2021-11-22
 */
@Configuration
public class HttpConfig {
    /**
     * 可访问无证书https请求的restTemplate
     *
     * @return restTemplate
     * @throws Exception exception
     */
    @Bean
    public RestTemplate restTemplate() throws Exception {
        TrustStrategy acceptingTrustStrategy = (x509Certificates, authType) -> true;
        SSLContext sslContext = SSLContexts.custom().loadTrustMaterial(null,
acceptingTrustStrategy).build();
        SSLConnectionSocketFactory connectionSocketFactory =
                new SSLConnectionSocketFactory(sslContext, new NoopHostnameVerifier());
        HttpClientBuilder httpClientBuilder = HttpClients.custom();
        httpClientBuilder.setSSLSocketFactory(connectionSocketFactory);
        CloseableHttpClient httpClient = httpClientBuilder.build();
        HttpComponentsClientHttpRequestFactory factory = new
HttpComponentsClientHttpRequestFactory();
        factory.setHttpClient(httpClient);
        factory.setConnectTimeout(20000);
        factory.setConnectTimeout(20000);
        return new RestTemplate(factory);
    }
}
```

4. 前台需要调用后台的服务，来获取Token和AppKey和apifabric的token，结合上述后台代码，前台可对 applyToken 方法进行改造。

```
async applyToken () {
        let apiResult = await axios({
            url: '/getTokenAndAppKey',
            method: 'GET',
            headers: {
                'Accept': 'application/json',
                'Content-Type': 'application/json;charset=UTF-8'
            }
        });
        if (apiResult.status !== 200) {
            return;
        }
        this.apiToken = apiResult.data['token'];
        this.appKey = apiResult.data['appKey'];
        if (this.userName) {
            await this.getMessageToken();
        }
        if (this.applyTask) {
            return;
        }
        //每55分钟刷新token
        this.applyTask = setInterval(this.applyToken, 55 * 60 * 1000);
}
```

----结束

4 代码使用示例-获取 Message Token

获取Message Token的目的是为了为后续的接口提供用户信息，将用户ID，用户名称，渠道ID，在Message服务生成一个Token映射，用于识别接入的用户。

关于如何获取用户信息，可以参考用户接入部分，或根据用户接入部分的方式进行改造。

参考代码路径： src/api/webChat.js

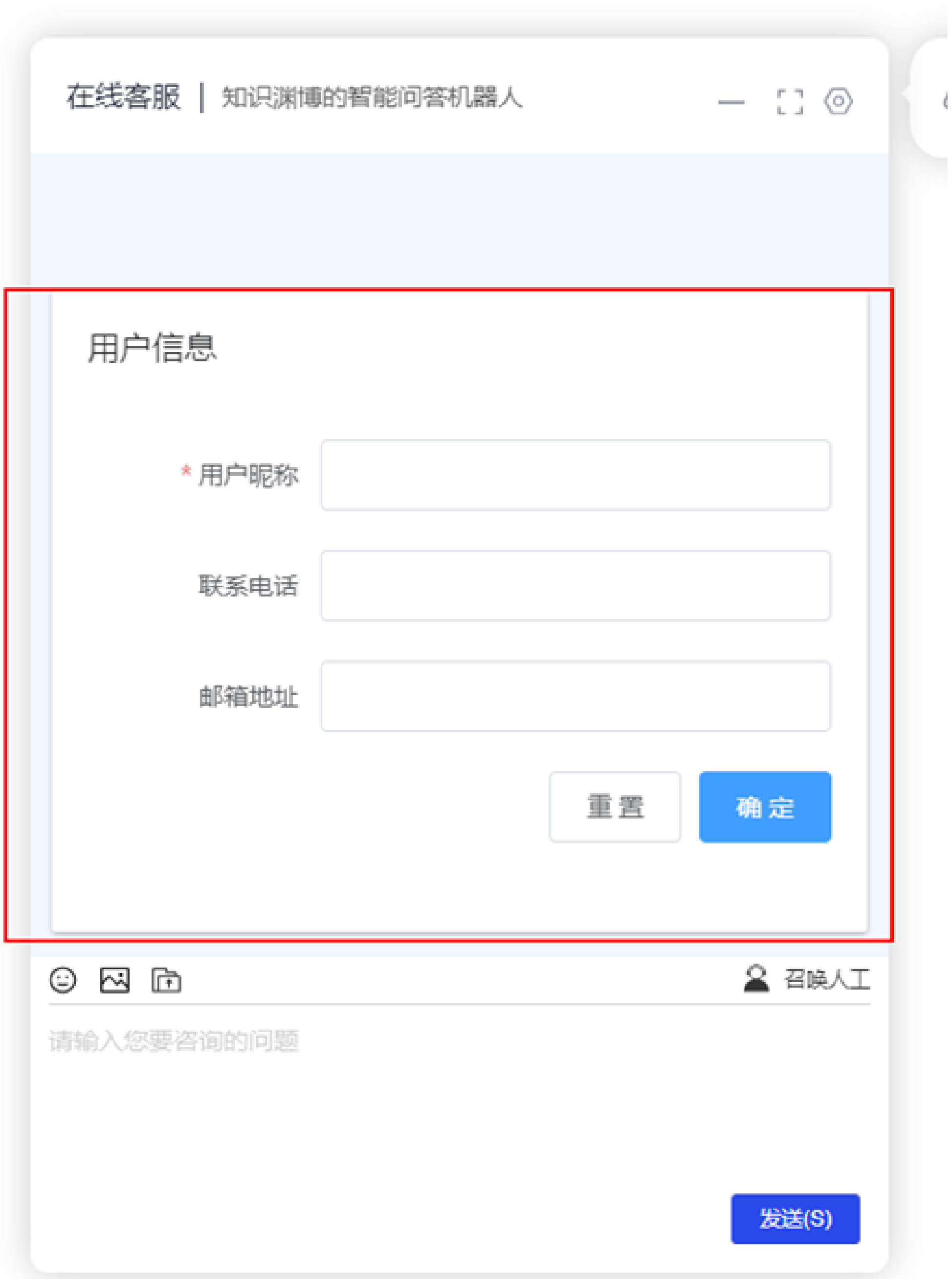
```
/**
 * 获取message的token
 *
 * @returns {Promise<*>}
 */
async getMessageToken(isRefresh = false) {
    //申请cc-messaging Token
    if (this.messageToken && !isRefresh) {
        return this.messageToken;
    }
    let apiResult = await axios({
        url: '/apiaccess/ccmessaging/applyToken',
        method: 'POST',
        headers: {
            'Accept': 'application/json',
            'Content-Type': 'application/json;charset=UTF-8',
            'Authorization': 'Bearer ' + this.apiToken,
            'x-app-key': this.appKey
        },
        data: {
            userId: this.userId,
            userName: this.userName,
            channelId: this.getChannelId(),
            locale: this.getLang()
        }
    });
    if (apiResult.status === 200) {
        this.messageToken = apiResult.data['token'];
    }
}
```

5 代码使用示例-用户接入

步骤1 当前demo首次进入时，需要输入用户信息，该代码位于/src/layout/UserForm.vue的mouted方法中的initUser方法。

```
//初始化进入聊天时需要处理的用户信息
async initUser() {
    let storage = window.localStorage;
    let userInfo = storage.getItem("sc_chat_user");
    if (userInfo) {
        let data = JSON.parse(userInfo);
        this.$Chat.userName = data.userName;
        this.$Chat.userId = data.userId;
        if (!this.$Chat.messageToken) {
            await this.$Chat.applyToken();
        }
        this.sendConnect();
        return;
    }
    this.dialogTableVisible = true;
}
```

方法会从前端的localStorage中获取用户信息，如果没有相关信息，则展示如下的用户信息输入框，需要输入用户昵称。如果存在用户信息，则调用上述**3 鉴权方式**中的鉴权，开始准备走接入流程；最后的sendConnect方法就是用户发送接入请求。



步骤2 当输入完用户信息并点击确认后，会进入到配置方法。

```
//配置用户信息
async configUserInfo() {
    let storage = window.localStorage;
    let data = {
        userName: this.form.userName,
        userId: "" + new Date().getTime() + this.$Utils.uuid(8, 16),
        userPhone: this.form.userPhone,
        userEmail: this.form.email
    }
    let dataString = JSON.stringify(data);
    storage.setItem("sc_chat_user", dataString);
    this.$Chat.userName = data.userName;
    this.$Chat.userId = data.userId;
    this.dialogTableVisible = false;
    if (!this.$Chat.messageToken) {
        await this.$Chat.applyToken();
    }
    this.sendConnect();
}
```

步骤3 该方法运用一个生成userId的逻辑，将用户信息存入到前端的LocalStorage中，并调用**3 鉴权方式**的鉴权方法获取ApiFabric的Token，随后调用sendConnect 发起申请连接到渠道的请求。申请方法如下

```
//发送连接请求
sendConnect(){
    if (!this.$Chat.messageToken){
        this.$alert('接口校验信息错误！ ')
        return
    }
    let connectionData = {
        channel: 'WEB',
        controlType: "CONNECT",
        mediaType: "TEXT",
        content: "hello",
        sourceType: "CUSTOMER",
        to: this.$Chat.getChannelId(),
        "from": this.$Chat.userId,
        senderNickname: this.$Chat.userName
    }
    this.$Chat.send(connectionData,()=>{
        EventBus.$emit("startPoll");
    })
}
```

步骤4 其中 send方法如下

```
/**
 * apiFabric send发送接口
 *
 * @param data
 * @param callbacks 回调函数，默认为空
 */
 send(data, callbacks = null) {
     if (data['content'].indexOf("data:image")>-1){
         let imgReg = new RegExp(/<img.*?(?:>|\/>)/gi);
         let arr_img = data['content'].match(imgReg, 'g');
         let matchStr = /data:image\/png;base64,(.*?)"/
         let matchArr = data['content'].match(matchStr)
         let requestParam = {
             fileType: 'png',
             fileStream: matchArr[matchArr.length-1],
             channel: 'WEB'
         }
         this.uploadFileStream((resp) => {
             if (resp && resp['objectKey']) {
                 let messageData = {
                     channel: 'WEB',
                     controlType: "CHAT",
                     mediaType: 'FILE_IMAGE',
                     content: resp['objectKey'] + ',png',
                     sourceType: "CUSTOMER",
                     to: this.getChannelId(),
                     "from": this.userId,
                     senderNickname: this.userName
                 };
                 this.send(messageData);
             }
         },requestParam)
         data ['content'] =  data['content'].replaceAll(arr_img[arr_img.length - 1],"");
     }
     if(data ['content'] === "" || data ['content'].length < 1) {
         return;
     }
```

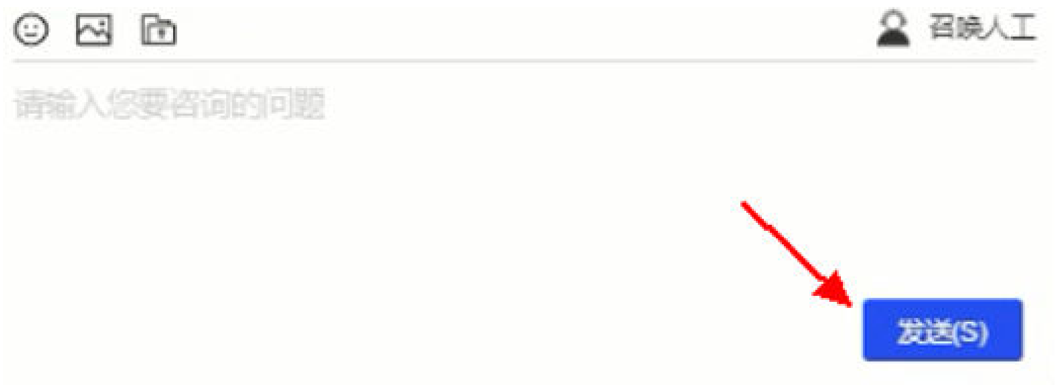
```
    axios({
        url: '/apiaccess/ccmessaging/send',
        method: 'POST',
        headers: {
            'Accept': 'application/json',
            'Content-Type': 'application/json;charset=UTF-8',
            'Authorization': 'Bearer ' + this.apiToken,
            'x-app-key': this.appKey,
            'ccmessaging-token': this.messageToken
        },
        data: data
    }).then(response => {
        if (response.status === 200) {
            if (data.controlType === 'CONNECT') {
                this.isChatting = true;
            }
            if (data.controlType === 'DISCONNECT') {
                this.isChatting = false;
            }
            if (callbacks != null) {
                callbacks(response.data);
            }
        }
    });
}
```

该接口入参为data发送参数，callback回调函数，该方法与发送聊天框中的信息为共用方法，会先校验发送的内容中有无图片信息，如果有，则调用apiFabric的chat聊天中的上传接口，如方法前半段所示。真正调用send的请求为/apiaccess/ccmessaging/send，该方法成功后会执行传入参数的回调函数。在 /src/layout/UserForm.vue中的sendConnect方法中，回调函数为向事件栈EventBus发送一个标识startPoll，即开始轮询获取座席侧发送的消息。

----结束

6 代码使用示例-用户发送消息

以页面的发送按钮为例，对应的方法为/src/layout/ Footer.vue文件的doSend方法。



```
//聊天框发送消息
doSend() {
    let content = this.$refs && this.$refs.inputBox.innerHTML;
    if (content !== "") {
        let sendContent=this.$Utils.getContext(content)
        if (this.countSubstr(sendContent,'data:image') > 1) {
            this.$message({
                message: '当前不支持发送超过一张图片',
                type: 'warning'
            });
        }
        if(sendContent.length - this.getPastePicLength(content)>=500){
            this.$message({
                message: '消息长度超出最大限制500',
                type: 'warning'
            });
        }else{
            this.dataString = sendContent
            this.$refs.inputBox.innerHTML = "";
            this.sendMessage()
        }
    }
}
```

该方法会调用sendMessage方法，sendMessage方法会调用webChat.js中的send方法，同用户接入的方法。

```
//发送消息到座席的方法
sendMessage() {
    let messageData = {
        channel: 'WEB',
        controlType: "CHAT",
        mediaType: "TEXT",
        content: this.dataString,
        sourceType: "CUSTOMER",
        to: this.$Chat.getChannelId(),
        "from": this.$Chat.userId,
        senderNickname: this.$Chat.userName
    };
    this.$Chat.send(messageData);
    messageData["pushType"] = 0;
    let msg = {
        avatar: "zph",
        text: this.$Utils.textChangeToImage(this.dataString) ,
        type: messageData["pushType"],
        time: this.$Utils.getDateString(),
        float: "right",
    };
    EventBus.$emit("pushInRecords", JSON.stringify(msg));
    this.dataString = '';
},
```

EventBus.$emit("pushInRecords", JSON.stringify(msg)); 为向事件栈 EventBus发送消息推入到聊天框的事件，对应的监听方法在MainContent.vue的mouted方法中。

```
//用户发消息
EventBus.$on("pushInRecords", (messageData) =>
    this.pushMessageInRecord(messageData)
);

//处理消息发送到聊天框
pushMessageInRecord(message) {
    let data = JSON.parse(message);
    if (data['text'].indexOf("data:image/png;base64") > -1) {
        let b = /<img src="data:image\/png;base64/g;
        data['text'] = data['text'].replaceAll(b,"<img style =\"max-width:350px;\"
onclick='window.showBgImg(this)' src=\"data:image/png;base64")
    }
    this.records.push(data);
    let div = this.$refs.box;
    setTimeout(() => {
        div.scrollTop = div.scrollHeight;
    }, 200);
},
```

用户发送多媒体文件的方法参考Footer.vue的uploadFile方法。

```
//文件 图片上传
uploadFile(type) {
    if (type === 'img') {
        this.$refs.imageInput.click();
    }
    if (type === 'file') {
        this.$refs.fileInput.click();
    }
}
```

该方法会调用getFile。

```
//文件 图片上传后处理
getFile(event) {
    const files = event.target.files;
    let size = files[0].size;
    let filename = files[0].name;
    const fileReader = new FileReader()
    let index = filename.lastIndexOf(".");
    let index2 = filename.length;
    let fileType = filename.substr(index + 1, index2);
    let sendFileType =  this.fileType[fileType];
    if (!sendFileType){
        this.$message({
            message: '当前文件类型不支持',
            type: 'warning'
        });
        return;
    }
    //内置方法new FileReader()   读取文件
    fileReader.addEventListener('load', () => {
        let fileData = fileReader.result;
        let fileBase64DataString = fileData.split(",")[1];

        let requestParam = {
            fileType: sendFileType,
            fileStream: fileBase64DataString,
            channel: 'WEB'
        }
        let that = this;
        //回调函数
        let callbacks = function (data) {
            event.target.value = "";
            if (data && data['objectKey']) {
                let messageData = {
                    channel: 'WEB',
                    controlType: "CHAT",
                    mediaType: that.mediaType[sendFileType],
                    content: data['objectKey'] + ',' + sendFileType,
                    sourceType: "CUSTOMER",
                    to: that.$Chat.getChannelId(),
                    "from": that.$Chat.userId,
                    senderNickname: that.$Chat.userName
                };
                if ('FILE' === that.mediaType[sendFileType]) {
                    messageData['content'] = filename.substr(0, index ) + ','
                    + size + ','
                        + data['objectKey'] + ','
                        + sendFileType
                }
                that.$Chat.send(messageData);
                let header = fileData.split(",")[0];
                let bytes = window.atob(fileBase64DataString);
                let arrayBuffer = new ArrayBuffer(bytes.length);
                let uint8Array = new Uint8Array(arrayBuffer);
                for (let i = 0; i < bytes.length; i++) {
                    uint8Array[i] = bytes.charCodeAt(i);
                }
                let blobFile = new Blob([uint8Array], {
                    type: header.match(/(.*?)/)[1]
                });
                let objectUrl = window.URL.createObjectURL(blobFile);
                let fileSize = size;
                if (fileSize < 1024 * 1024) {
                    fileSize = (fileSize / 1024).toFixed(2) + "KB";
                } else {
                    fileSize = (fileSize / 1024 / 1024).toFixed(2) + "MB";
                }
                let messageInRecords = {
                    avatar: "zph",
                    text: objectUrl,
                    type: that.showType[sendFileType],
                    time: that.$Utils.getDateString(),
                    float: "right-media",
                    fileName:filename,
                    fileType:sendFileType,
                    fileSize:fileSize
                }
                if (that.showType[sendFileType] === 1) {
                    let imgList = [];
                    imgList.push(objectUrl);
                    messageInRecords["imgList"] = imgList;
                }
                EventBus.$emit("pushInRecords", JSON.stringify(messageInRecords));
            }
        }
        this.$Chat.uploadFileStream(callbacks, requestParam);

    })
    fileReader.readAsDataURL(files[0])
}
```

在获取文件后，会调用uploadFileStream接口去发送多媒体文件到客服座席侧。

7 代码使用示例-用户接收消息

在用户接入时，发送连接请求携带着回调函数：开始轮询客服发来的消息，可参考UserForm.vue中的sendConnect方法。

```
sendConnect(){
    if (!this.$Chat.messageToken){
        this.$alert('接口校验信息错误！')
        return
    }
    let connectionData = {
        channel: 'WEB',
        controlType: "CONNECT",
        mediaType: "TEXT",
        content: "hello",
        sourceType: "CUSTOMER",
        to: this.$Chat.getChannelId(),
        "from": this.$Chat.userId,
        senderNickname: this.$Chat.userName
    }
    this.$Chat.send(connectionData,()=>{
        EventBus.$emit("startPoll");
    })
}
```

其中在send方法执行成功后，会执行EventBus.$emit("startPoll");的回调函数，意为向事件栈中发送startPoll事件，监听方法在MainContent.vue的mouted方法中：

```
//收消息
EventBus.$on("startPoll", this.pushAgentMessage);
```

会调用pushAgentMessage方法，具体方法如下：

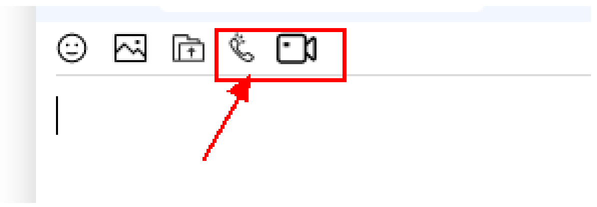
```
//处理座席侧所有消息总函数
pushAgentMessage() {
    let that = this;
    let agentFunc = function (data) {
        if (data && data["downlinkMessages"]) {
            let downLinkMessage = data["downlinkMessages"];
            for (let i = 0; i < downLinkMessage.length; i++) {
                if (downLinkMessage[i]["sourceType"] === "AGENT") {
                    that.toAgent = true;
                    that.tipsObject.show = false;
                    that.tipsObject.showCancel = false;
                    if (that.isFirstToAgent) {
                        EventBus.$emit("changeTalkStatus", "toAgent")
                        that.isFirstToAgent = false;
                        EventBus.$emit("changeAgent", downLinkMessage[i]["senderNickname"])
                    }
                }
                if (downLinkMessage[i]["sourceType"] === "ROBOT") {
                    that.dealWithRobot(downLinkMessage[i]);
                    continue;
                }
                if (that.mediaType[downLinkMessage[i]["mediaType"]] != null) {
                    that.dealWithMedia(downLinkMessage[i]);
                    continue;
                }
                if (downLinkMessage[i]["controlType"] === "DISCONNECT") {
                    that.dealWithDrop();
                    continue;
                }
                //放入到展示区
                let content = that.$Utils.extractUrl(downLinkMessage[i]["content"])
                let msg = {
                    avatar: "zph",
                    text: that.$Utils.textChangeToImage(content),
                    type: 0,
                    time: that.$Utils.getDateString(),
                    float: "left",
                    userName: downLinkMessage[i]["senderNickname"]
                };
                that.pushInRecords(msg);
                if (downLinkMessage[i]["queueFlag"]) {
                    that.tipsObject.show = true;
                    that.queryQueue();
                }
            }
        }
        if (that.$Chat.isChatting) {
            EventBus.$emit("startPoll");
        }
    };
    setTimeout(() => {
        this.$Chat.poll(agentFunc);
    }, 100)
},
```

该方法是一个一直在轮询的方法，其中对于座席发来的不同消息类型，有不同的处理方式，关于座席返回的消息内容，可以参考接口参考中开放接口的poll方法；

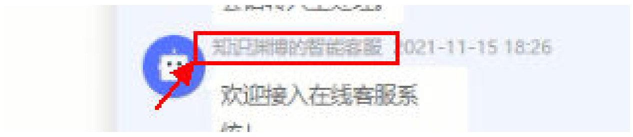
其中一些方法的说明：

```
if (downLinkMessage[i]["sourceType"] === "AGENT") {
    that.toAgent = true;
    that.tipsObject.show = false;
    that.tipsObject.showCancel = false;
    if (that.isFirstToAgent) {
        EventBus.$emit("changeTalkStatus", "toAgent")
        that.isFirstToAgent = false;
        EventBus.$emit("changeAgent", downLinkMessage[i]["senderNickname"])
    }
}
```

当接收到第一条座席发来的信息时，会发送 changeTalkStatus 和 changeAgent 事件，其中changeTalkStatus事件会使监听方法去检查当前对话的座席是否支持点击通话（即音视频交谈），以及当前用户环境是否支持语音交谈，如果支持，会在demo的聊天工具栏中展示可以点击的音视频通话按键。



changeAgent事件会使监听方法改变对话者的名称。



```
if (downLinkMessage[i]["sourceType"] === "ROBOT") {
    that.dealWithRobot(downLinkMessage[i]);
    continue;
}
```

上述代码中会处理客服侧发来的消息，为机器人发来的消息类型。

```
if (that.mediaType[downLinkMessage[i]["mediaType"]] != null) {
    that.dealWithMedia(downLinkMessage[i]);
    continue;
}
```

上述代码中会处理客服侧发来的消息，为多媒体的消息类型。

```
//处理客服侧发送的媒体类型消息
dealWithMedia(data) {
    let fileId = data['content'];
    let mediaFileType = this.mediaType[data['mediaType']];
    let requestParam = {
        fileId: fileId,
        channel: 'WEB',
        fileType: mediaFileType,
        multiMedia: 'multiMedia'
    }
    let that = this;
    let itemType = this.itemType[data['mediaType']]
    let callbacks = function (respData) {
        if (respData['resultCode'] === '0') {
            let msg;
            if (itemType === 1) {
                let imgSrc = 'data:image/jpeg;base64,' + respData['fileStream'];
                let imgList = [];
                imgList.push(imgSrc);
                msg = {
                    avatar: "zph",
                    text: imgSrc,
                    type: itemType,
                    time: that.$Utils.getDateString(),
                    float: "left",
                    imgList: imgList
                };
            } else {
                let typeHeader = 'data:audio/mp3;base64,';
                let type = "audio/mp3";
                if (itemType === 2) {
                    typeHeader = 'data:video/mp4;base64,';
                    type = 'video/mp4';
                }
                let audioSource = typeHeader + respData['fileStream'];
                let arr = audioSource.split(',');
                let array = arr[0].match(/:(.*?);/);
                let mime = (array && array.length > 1 ? array[1] : type) || type;
                let bytes = window.atob(arr[1])
                let arrayBuffer = new ArrayBuffer(bytes.length);
                let uint8Array = new Uint8Array(arrayBuffer);
                for (let i = 0; i < bytes.length; i++) {
                    uint8Array[i] = bytes.charCodeAt(i);
                }
                let blobFile = new Blob([uint8Array], {
                    type: mime
                });
                let objectUrl = window.URL.createObjectURL(blobFile);
                msg = {
                    avatar: "zph",
                    text: objectUrl,
                    type: itemType,
                    time: that.$Utils.getDateString(),
                    float: "left",
                };
            }

            that.pushInRecords(msg);
        }
    }
    this.$Chat.downloadFileStream(callbacks, requestParam);
}
```

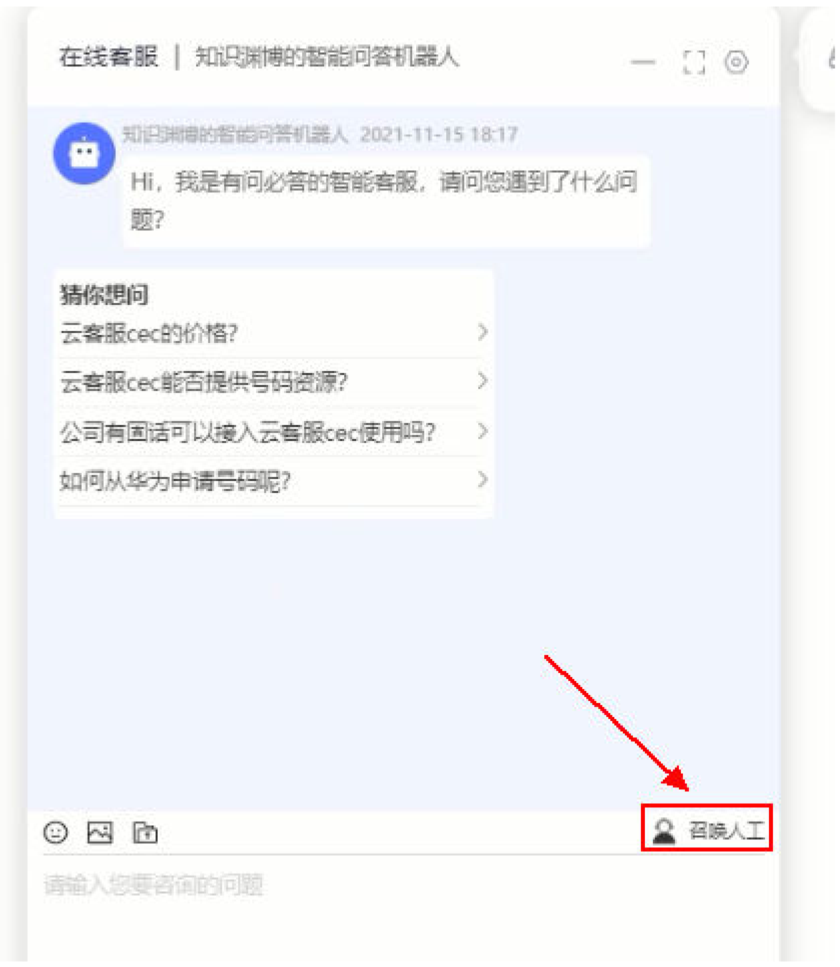
在识别到是多媒体消息时，会调用downloadFileStream的接口。

```
if (downLinkMessage[i]["controlType"] === "DISCONNECT") {
    that.dealWithDrop();
    continue;
}
```

上述代码中会处理客服侧发来的消息，为客户结束会话的消息类型。

8 代码使用示例-用户转人工

参考Demo举例：


在线客服 | 知识渊博的智能问答机器人

知识渊博的智能问答机器人 2021-11-15 18:17

Hi，我是有问必答的智能客服，请问您遇到了什么问题?

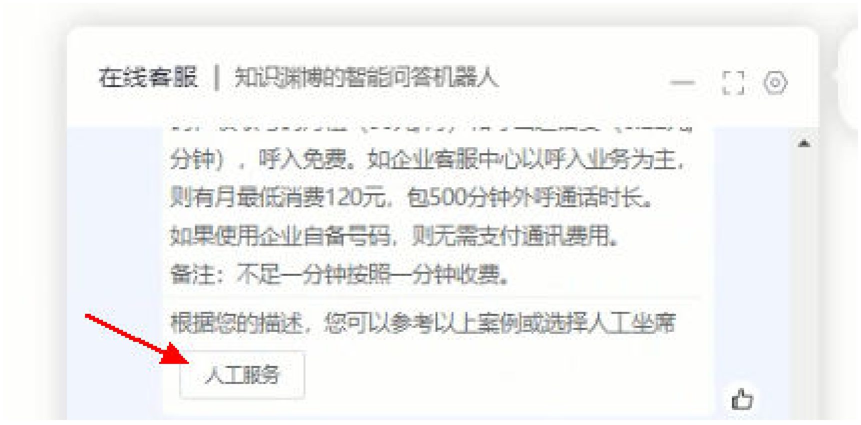
猜你想问

云客服cec的价格?

云客服cec能否提供号码资源?

公司有固话可以接入云客服cec使用吗?

如何从华为申请号码呢?


在线客服 | 知识瑞博的智能问答机器人

分钟），呼入免费。如企业客服中心以呼入业务为主，
则有月最低消费120元，包500分钟外呼通话时长。
如果使用企业自备号码，则无需支付通讯费用。
备注：不足一分钟按照一分钟收费。

根据您的描述，您可以参考以上案例或选择人工坐席

上述两种召唤人工的方式均可实现，采用相同的调用方式，可以参考MainContent.vue的transform方法。

```
//召唤人工
transform() {
    let transformData = {
        channel: 'WEB',
        controlType: "TRANS2AGENT",
        mediaType: "TEXT",
        content: 'hello',
        sourceType: "CUSTOMER",
        to: this.$Chat.getChannelId(),
        "from": this.$Chat.userId,
        senderNickname: this.$Chat.userName
    };
    let that = this;
    let transSuccess = function (data) {
        if (data['resultCode'] === '0') {
            that.sendUserInfo();
        } else {
            EventBus.$emit("toAgentFailed");
        }
    }
    this.$Chat.send(transformData, transSuccess);
},
```

Demo当前在send方法调用失败时，会发送toAgentFailed事件，该事件会触发用户留言的方法。成功时，会向客服座席发送用户在接入时填写的用户信息，可以参考用户接入中的信息；具体方法如下：

```
//向座席发送用户信息
sendUserInfo() {
    let storage = window.localStorage;
    let userInfo = storage.getItem("sc_chat_user");
    if (userInfo) {
        let data = JSON.parse(userInfo);
        let message = '';
        if (data.userPhone) {
            message += ('用户电话:' + data.userPhone + ",");
        }
        if (data.userEmail) {
            message += ('用户邮箱:' + data.userEmail);
        }
        if (message !== '') {
            let connectMessage = {
                channel: 'WEB',
                controlType: "CHAT",
                mediaType: "TEXT",
                content: message,
                sourceType: "CUSTOMER",
                to: this.$Chat.getChannelId(),
                "from": this.$Chat.userId,
                senderNickname: this.$Chat.userName
            };
            this.$Chat.send(connectMessage);
        }
    }
}
```

9 代码使用示例-用户查询排队

在用户接收消息中，存在一个排队的消息：

```
if (downLinkMessage[i]["queueFlag"]) {
    that.tipsObject.show = true;
    that.queryQueue();
}
```

可以参考排队查询方法

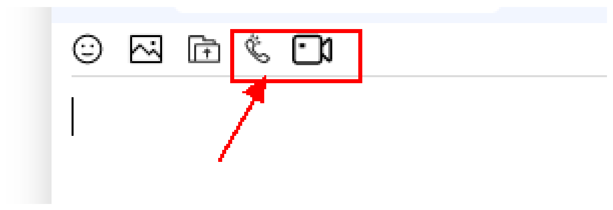
```
//查询排队信息
queryQueue() {
    let that = this;
    let callbacks = function (data) {
        if (data['resultCode'] !== "0") {
            return;
        }
        that.tipsObject.message = '您当前排在第' + data['position'] + '预计还需等待' + 
data['estimateWaitTime'] + '秒';
        that.tipsObject.showCancel = true;
        setTimeout(that.queryQueue, 10000);
    }
    this.$Chat.queryQueueInfo(callbacks);
}
```

当前存在setTimeout延时方法，在查询排队成功后，会在10秒后继续调用该方法查询。在接口调用返回的resultCode不为0时，停止查询。

10 代码使用示例-用户发起点击通话

发起点击通话的前提，是在接收用户消息时，通过了webRTC环境的校验。

当demo中出现了以下按钮，代表可以发起点击通话。



发起的方法，可以参考Footer.vue中的createCall方法。

```
/音视频通话
createCall(callType) {
    this.mode = callType === '0' ? 'audio' : 'video';
    let callbacks = function () {
        EventBus.$emit("startCallPoll");
    }
    this.$Chat.createClickToCall(callType, callbacks);
    this.isTalking = true;
    this.callType = callType;
}
```

该方法会调用createClickToCall接口，当方法得到成功返回后，会调用callbacks回调函数，回调函数中的方法EventBus.$emit("startCallPoll")意为发送一个startCallPoll事件。

事件的监听方法如下：

```
//轮询座席消息
EventBus.$on("startCallPoll", () => {
    this.getCallEvent();
});
```

会开始调用getClickToCallEvents接口，开始轮询点击通话事件。

```
//轮询获取通话事件
getCallEvent() {
    setTimeout(() => {
        this.$Chat.getClickToCallEvents(this.callbacks);
    }, 100);
}
```

这里面的callbacks方法如下，其中new AudioCodesUA()来自奥科的SDK

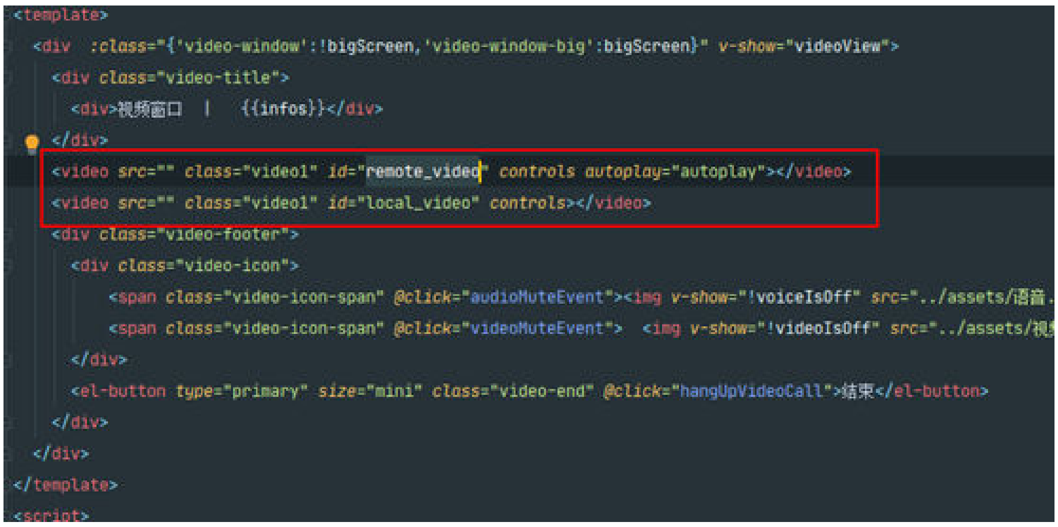
```
//音视频相关回调函数
callbacks(data) {
    if (data && data['resultCode'] === '0') {
        let eventId = data['eventId'];
        if (eventId) {
            if (eventId === 168101) { //已接入到座席
                const msgContent = data['content'];
                // eslint-disable-next-line no-undef
                this.webRtcConfig.phone = new AudioCodesUA();
                this.webRtcConfig.callTo = msgContent['accessCode']
                this.webRtcConfig.serverConfig.domain = msgContent['domain']
                this.webRtcConfig.serverConfig.addresses = msgContent['gwAddresses']
                this.webRtcConfig.account.user = msgContent['clickToCallCaller']
                this.webRtcConfig.account.displayName = msgContent['clickToCallCaller']
                this.initSipStack();
            } else if (eventId === 168102) {
                //that.$Chat.guiInfo('排队中....')
            } else if (eventId === 168106) { // 呼叫转移
                //that.$Chat.guiInfo('呼叫转移中')
            } else if (eventId === 168110) { // 呼叫释放
                this.isTalking = false
            } else if (eventId === 168103) { // 呼叫排队超时
                this.isTalking = false
            } else if (eventId === 168105) { // 呼叫失败
                this.isTalking = false
                this.$message({
                    message: '建立通话失败！',
                    type: 'warning'
                });
            }else {
                this.$message({
                    message: '建立通话失败！失败原因码为:' + eventId,
                    type: 'warning'
                });
                if (this.callType === '1'){
                    EventBus.$emit("videoDrop");
                }else {
                    EventBus.$emit("audioDrop");
                }
            }
        }
    } else {
        this.isTalking = false;
    }
    if (this.isTalking) {
        EventBus.$emit("startCallPoll");
    }
}
```

在收到168101事件代表已成功接入到座席，这时候会调用initSipStack方法参考如下。

```
initSipStack() {
    let phone = this.webRtcConfig.phone
    phone.setServerConfig(this.webRtcConfig.serverConfig.addresses,
        this.webRtcConfig.serverConfig.domain,
        this.webRtcConfig.serverConfig.iceServers)

    phone.setAccount(this.webRtcConfig.account.user,
        this.webRtcConfig.account.displayName,
        this.webRtcConfig.account.password)

    // Set phone API listeners
    let that = this
    phone.setListeners({
        loginStateChanged: function (isLogin, cause) {
            switch (cause) {
                case 'connected':
                    that.ac_log('phone>>> loginStateChanged: connected')
                    if (that.webRtcConfig.activeCall !== null) {
                        that.ac_log('Already exists active call')
                    } else {
                        if (that.mode === 'video') {
                            that.webRtcConfig.activeCall = phone.call(phone.VIDEO,
                                that.webRtcConfig.callTo)
                        } else {
                            that.webRtcConfig.activeCall = phone.call(phone.AUDIO,
                                that.webRtcConfig.callTo)
                        }
                        EventBus.$emit("showAudio");
                    }
                    break
                case 'disconnected':
                    that.ac_log('phone>>> loginStateChanged: disconnected')
                    if (phone.isInitialized()) {
                        that.ac_log('Cannot connect to SBC server')
                    }
                    if (that.callType === '1'){
                        EventBus.$emit("videoDrop");
                    }else {
                        EventBus.$emit("audioDrop");
                    }
                    that.ac_log('service disconnected')
                    break
                case 'login failed':
                    that.ac_log('phone>>> loginStateChanged: login failed')
                    break
                case 'login':
                    that.ac_log('phone>>> loginStateChanged: login')
                    break
                case 'logout':
                    that.ac_log('phone>>> loginStateChanged: logout')
                    break
            }
        },

        // eslint-disable-next-line no-unused-vars
        outgoingCallProgress: function (call, response) {
            that.ac_log('phone>>> outgoing call progress')
            EventBus.$emit("callMessage", "呼叫中");
        },
        // eslint-disable-next-line no-unused-vars
        callTerminated: function (call, message, cause, redirectTo) {
            that.ac_log('phone>>> call terminated callback, cause=%o', cause)
            if (call !== that.webRtcConfig.activeCall) {
                that.ac_log('terminated no active call')
                return
            }
            that.webRtcConfig.activeCall = null
            that.ac_log('Call terminated: ', cause)
            phone.deinit() // Disconnect from SBC server.
            that.isTalking = false // 轮询结束
            console.log('Stop polling, drop existing ClickToCall, reset CallDurationTimer and enable ClickToCall')
            that.guiClearVideoView()
        },

        // eslint-disable-next-line no-unused-vars
        callConfirmed: function (call, message, cause) {
            that.ac_log('phone>>> callConfirmed')
            // Show or hide video controls, according call property 'video'

            let hasVideo = call.hasVideo()
            that.guiToggleLocalVideo() // set local video according current check box setting.
            if (hasVideo) {
                EventBus.$emit("showVideo");
                EventBus.$emit("hideAudioImmediately");
            } else {
                EventBus.$emit("callMessage", "time");
            }
        },
        callShowStreams: function (call, localStream, remoteStream) {
            console.log('phone>>> callShowStreams')
            let remoteVideo = document.getElementById('remote_video')
            remoteVideo.srcObject = remoteStream // to play audio and optional video
        },

        // eslint-disable-next-line no-unused-vars
        incomingCall: function (call, invite) {
            console.log('phone>>> incomingCall')
            call.reject()
        },

        // eslint-disable-next-line no-unused-vars
        callHoldStateChanged: function (call, isHold, isRemote) {
            console.log('phone>>> callHoldStateChanged ' + isHold ? 'hold' : 'unhold')
        }
    })
    phone.init(false)
}
```

上述的方法来自奥科文档，具体使用方式可以参考奥科官网，搜索webrtc-web-browser-client-sdk-api-reference-guide了解。本处提示几个地方：

本方和对方的音视频展示，需要有一个video标签，可以参考VideoWindow.vue中的：

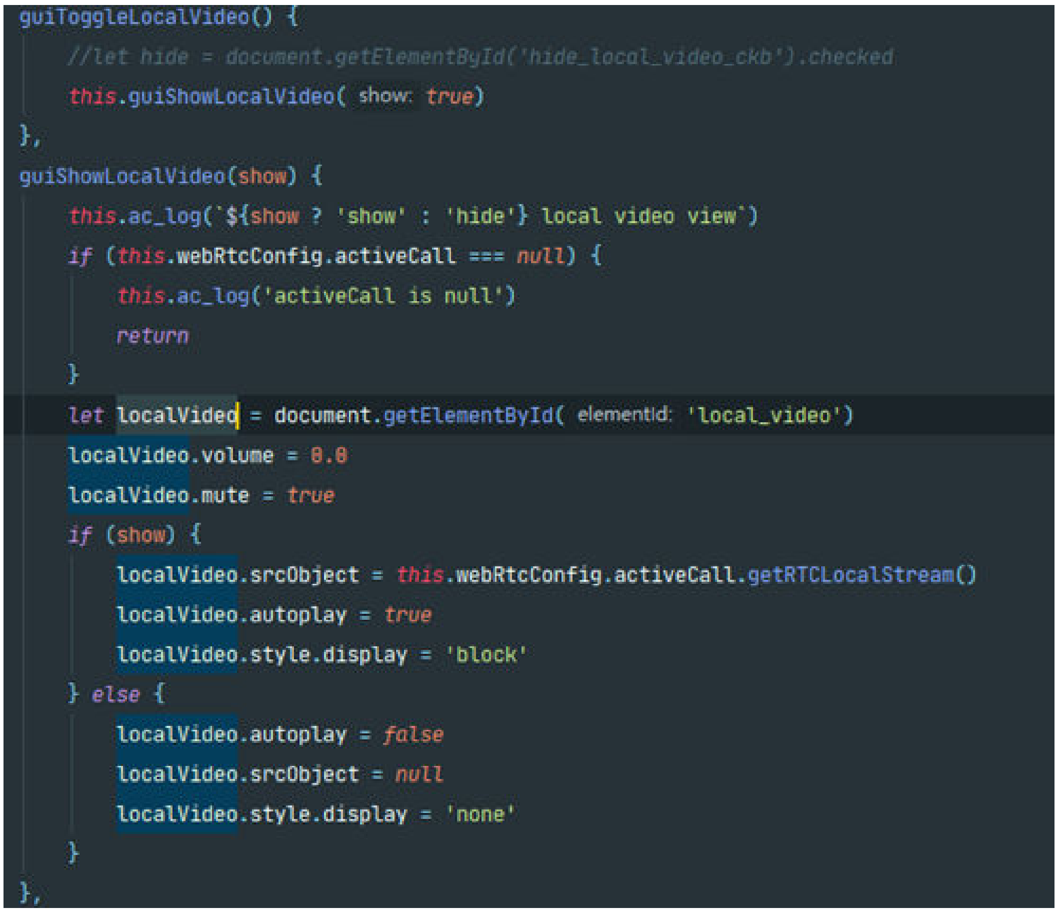


```
<template>
  <div  :class="{'video-window':!bigScreen,'video-window-big':bigScreen}" v-show="videoView">
    <div class="video-title">
      <div>视频窗口  |   {{infos}}</div>
    </div>
    <video src="" class="video1" id="remote_video" controls autoplay="autoplay"></video>
    <video src="" class="video1" id="local_video" controls></video>
    <div class="video-footer">
      <div class="video-icon">
          <span class="video-icon-span" @click="audioMuteEvent"><img v-show="!voiceIsOff" src="../assets/语音
          <span class="video-icon-span" @click="videoMuteEvent">  <img v-show="!videoIsOff" src="../assets/视
      </div>
      <el-button type="primary" size="mini" class="video-end" @click="hangUpVideoCall">结束</el-button>
    </div>
  </div>
</template>
<script>
```

在通话建立时，sdk会调用callConfirmed方法

```
callConfirmed: function (call, message, cause) {
    that.ac_log('phone>>> callConfirmed')
    // Show or hide video controls, according call property 'video'

    let hasVideo = call.hasVideo()
    that.guiToggleLocalVideo() // set local video according current check box setting.
    if (hasVideo) {
        EventBus.$emit("showVideo");
        EventBus.$emit("hideAudioImmediately");
    } else {
        EventBus.$emit("callMessage", "time");
    }
}
```

会展示本方的音视频媒体。



```
guiToggleLocalVideo() {
    //let hide = document.getElementById('hide_local_video_ckb').checked
    this.guiShowLocalVideo( show: true)
},
guiShowLocalVideo(show) {
    this.ac_log(`${show ? 'show' : 'hide'} local video view`)
    if (this.webRtcConfig.activeCall === null) {
        this.ac_log('activeCall is null')
        return
    }
    let localVideo = document.getElementById( elementId: 'local_video')
    localVideo.volume = 0.0
    localVideo.mute = true
    if (show) {
        localVideo.srcObject = this.webRtcConfig.activeCall.getRTCLocalStream()
        localVideo.autoplay = true
        localVideo.style.display = 'block'
    } else {
        localVideo.autoplay = false
        localVideo.srcObject = null
        localVideo.style.display = 'none'
    }
},
```

存在对方媒体时会调用callShowStreams展示对方媒体。

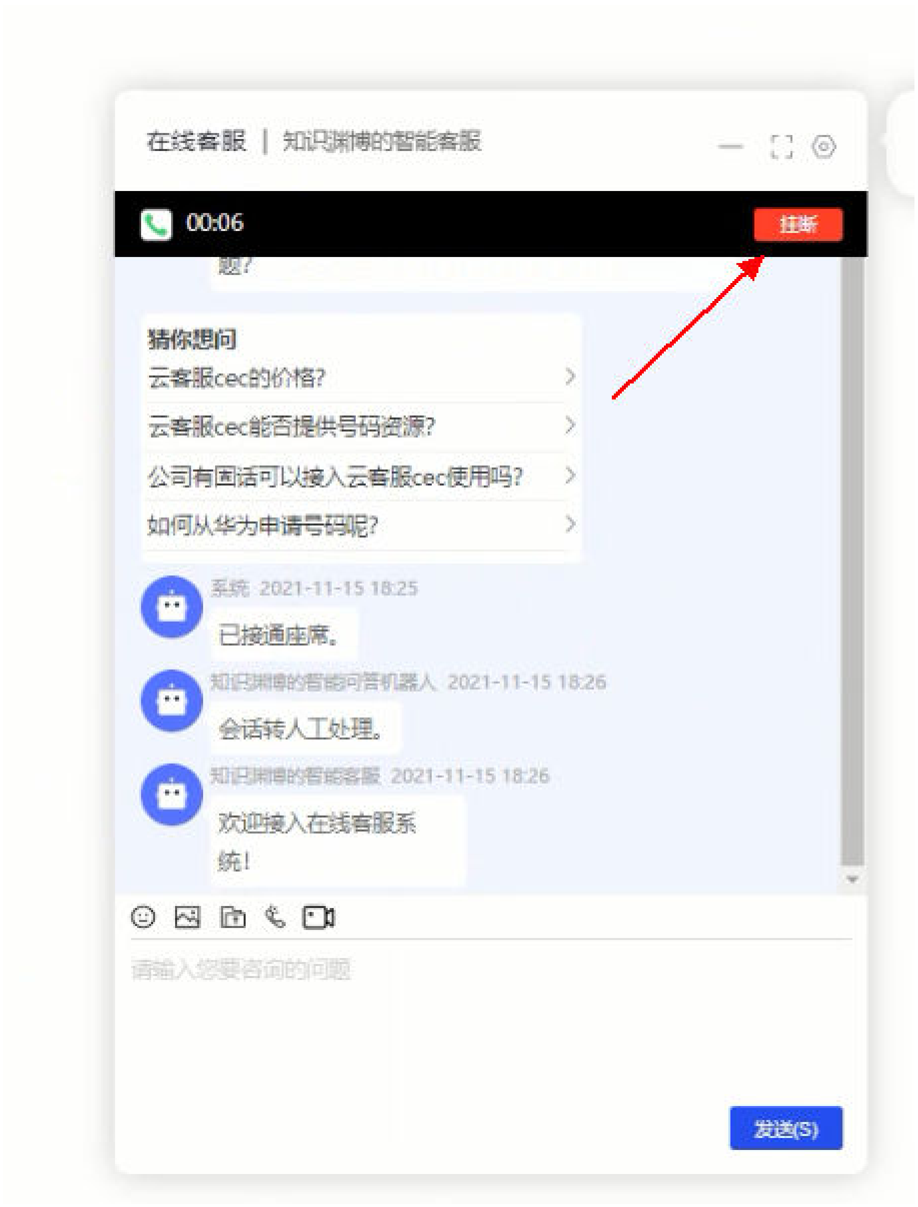
```
callShowStreams: function (call, localStream, remoteStream) {
    console.log('phone>>> callShowStreams')
    let remoteVideo = document.getElementById('remote_video')
    remoteVideo.srcObject = remoteStream // to play audio and optional video
}
```

通话中断会调用callTerminated 方法。

```
callTerminated: function (call, message, cause, redirectTo) {
    that.ac_log('phone>>> call terminated callback, cause=%o', cause)
    if (call !== that.webRtcConfig.activeCall) {
        that.ac_log('terminated no active call')
        return
    }
    that.webRtcConfig.activeCall = null
    that.ac_log('Call terminated: ', cause)
    phone.deinit() // Disconnect from SBC server.
    that.isTalking = false // 轮询结束
    console.log('Stop polling, drop existing ClickToCall, reset CallDurationTimer and enable ClickToCall')
    that.guiClearVideoView()
}
```

因其他情况结束了通话流程会触发disconnected事件。

```
case 'disconnected':
    that.ac_log('phone>>> loginStateChanged: disconnected')
    if (phone.isInitialized()) {
        that.ac_log('Cannot connect to SBC server')
    }
    if (that.callType === '1'){
        EventBus.$emit("videoDrop");
    }else {
        EventBus.$emit("audioDrop");
    }
    that.ac_log('service disconnected')
    break
```



题?

猜你想问
云客服cec的价格?
云客服cec能否提供号码资源?
公司有固话可以接入云客服cec使用吗?
如何从华为申请号码呢?

系统 2021-11-15 18:25
已接通座席。

知识渊博的智能问答机器人 2021-11-15 18:26
会话转人工处理。

知识渊博的智能客服 2021-11-15 18:26
欢迎接入在线客服系统!

语音通话的结束按键方法可以参考MainContent.vue中的HangUp方法。

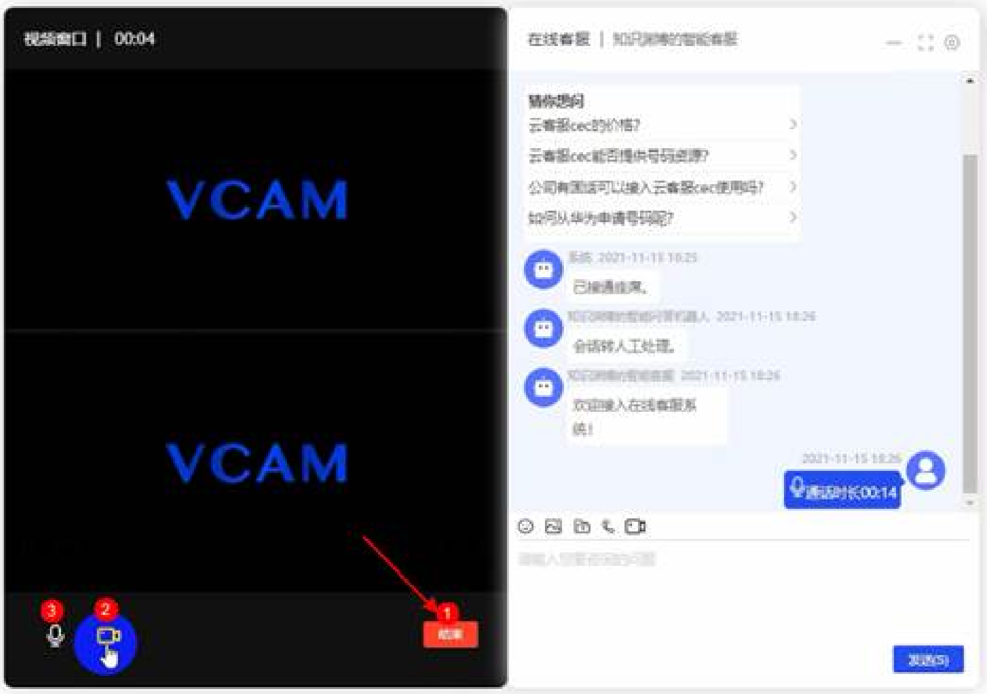
```
/**
 * 挂断点击通话
 */
hangUp() {
    if (this.hangUpButton) {
        EventBus.$emit("hangUp");
        clearInterval(this.talkTimeTask);
        this.hangUpButton = false;
        this.audioIn = "0";
        this.sec = 0;
        this.min = 0;
        let msg = {
            avatar: "zph",
            text: `<img src="` + audioPng + `" class="footer-image" alt="" style="height: 16px;margin-right:
5px;vertical-align: middle;margin-bottom: 2px">` + "通话时长" + this.infos,
            type: 0,
            time: this.$Utils.getDateString(),
            float: "right",
        };
        if (this.infos!=='呼叫中'){
            EventBus.$emit("pushInRecords", JSON.stringify(msg));
        }
        this.infos = "00:00";
    }
}
```

该方法会发送hangUp事件：EventBus.$emit("hangUp");在Footer.vue中有该事件的监听，会调用cancel方法。

```
//音视频挂断
EventBus.$on("hangUp", () => {
    this.cancel();
});
```

其中dropClickToCall会调用dropClickToCall接口。

```
cancel() {
    if (this.webRtcConfig.activeCall != null) {
        this.webRtcConfig.activeCall.terminate()
        this.webRtcConfig.activeCall = null
    }
    this.$Chat.dropClickToCall();
}
```



猜你想问
云客服cec的价格?
云客服cec能否提供号码资源?
公司有固话可以接入云客服cec使用吗?
如何从华为申请号码呢?

系统 2021-11-15 18:25
已接通座席。

知识测推的智能问答机器人 2021-11-15 18:26
会话转人工处理。

知识测推的智能客服 2021-11-15 18:26
欢迎接入在线客服系统！

2021-11-15 18:26
通话时长00:14

在视频通话中

1. 挂断方法在VideoWindow.vue中的hangUpVideoCall。

```
2.      hangUpVideoCall(){
    if (this.videoView){
      EventBus.$emit("hangUp");
      this.videoView = false;
      let msg = {
        avatar: "zph",
        text: `<img src="`+videoPng+`" class="footer-image" alt="" style="height: 16px;margin-right:
5px;vertical-align: middle;margin-bottom: 2px">`+"通话时长"+this.infos,
        type: 0,
        time: this.$Utils.getDateString(),
        float: "right",
      };
      if (this.infos!=='呼叫中'){
        EventBus.$emit("pushInRecords", JSON.stringify(msg));
      }
      this.infos = "00:00";
      this.sec = 0;
      this.min = 0;
      clearInterval(this.talkTimeTask);
      this.talkTimeTask = "";
    }
  }
```

该方法同样会发送hangUp事件，同音频挂断。

2. 停止发送本地视频VideoWindow.vue中的hangUpVideoCall

```
/**
 * 视频画面停止
 */
videoMuteEvent(){
  this.videoIsOff = !this.videoIsOff;
  EventBus.$emit("videoMute");
}
```

该方法会发送videoMute事件，在Footer.vue中监听。

```
EventBus.$on("videoMute", () => {
    this.videoMute()
});
videoMute() {
    let muted = this.webRtcConfig.activeCall.isVideoMuted()
    this.webRtcConfig.activeCall.muteVideo(!muted)
}
```

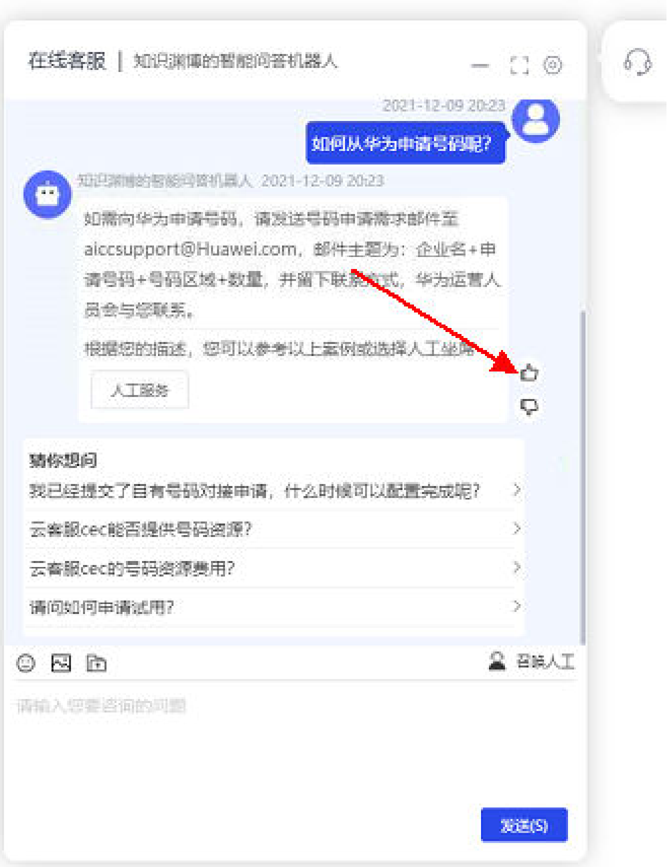
3. 视频静音方法VideoWindow.vue中的audioMuteEvent。

```
/**
 * 视频语音静音
 */
audioMuteEvent(){
  this.voiceIsOff = !this.voiceIsOff;
  EventBus.$emit("audioMute");
}
```

该方法会发送audioMute事件，在Footer.vue中监听。

```
EventBus.$on("audioMute", () => {
    this.audioMute()
})
audioMute() {
    let muted = this.webRtcConfig.activeCall.isAudioMuted()
    this.webRtcConfig.activeCall.muteAudio(!muted)
}
```

11 代码使用示例-用户评价



2021-12-09 20:23

如何从华为申请号码呢?

知识渊博的智能问答机器人 2021-12-09 20:23

如需向华为申请号码，请发送号码申请需求邮件至aiccsupport@Huawei.com，邮件主题为：企业名+申请号码+号码区域+数量，并留下联系方式，华为运营人员会与您联系。

根据您的描述，您可以参考以上案例或选择人工坐席

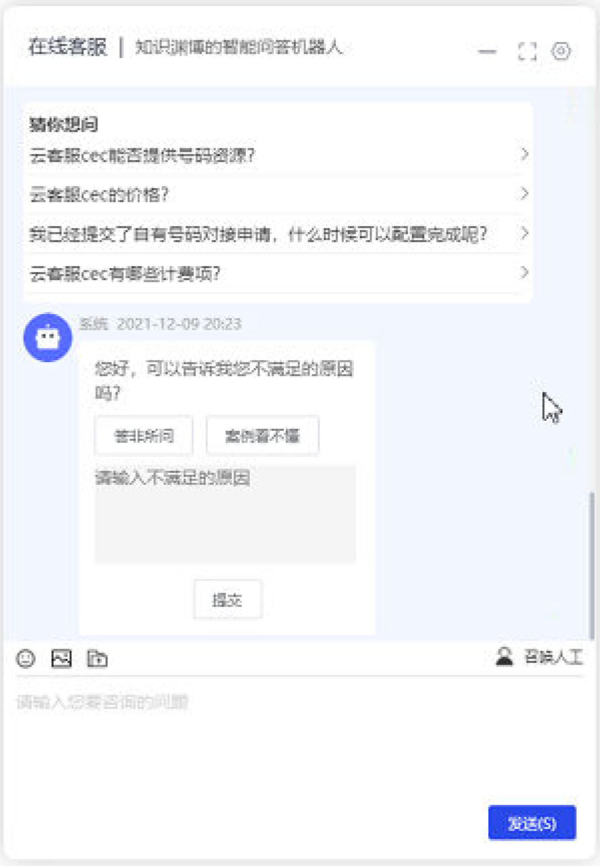
猜你想问

我已经提交了自有号码对接申请，什么时候可以配置完成呢?

云客服cec能否提供号码资源?

云客服cec的号码资源费用?

请问如何申请试用?



猜你想问
云客服cec能否提供号码资源?
云客服cec的价格?
我已经提交了自有号码对接申请，什么时候可以配置完成呢?
云客服cec有哪些计费项?

系统 2021-12-09 20:23
您好，可以告诉我您不满足的原因吗?

页面操作分为好评和差评，对应的代码在MainContent.vue的feedbackSatisfaction中。

```
<div class="satisfaction-dissatisfied" v-if="item.isRobot || item.isDrop">
    <img class="satisfaction-img" :id="'s'+item.commentImgId"
         src="../assets/满意.svg" alt="" title="满意" style="position: absolute;bottom: 35px;cursor: pointer;"
         @click="feedbackSatisfaction(item,1)">
    <img class="satisfaction-img"  :id="'dis'+item.commentImgId"
         src="../assets/不满意.svg" alt="" title="不满意" style="position: absolute;bottom: 5px;cursor: pointer;"
         @click="feedbackSatisfaction(item,0)">
    <img class="satisfaction-img"  :id="'show'+item.commentImgId"
         src="" alt="" style="position: absolute;bottom: 5px; display: none">
</div>
```

其中满意为1，不满意为0。

```
//满意度调查
feedbackSatisfaction(item, feedback,commentId=null) {
    if (feedback === 1) {
        if (item.isRobot) {
            this.$Chat.feedbacksatisfaction(item.interIdx, feedback,"ok");
        }else {
            this.$Chat.satisfactionInfo("5","ok");
        }
        this.showCommentImg(item.commentImgId,satisfied);
        let msg = {
            avatar: "zph",
            text: "感谢您的点赞，我会继续努力的~",
            type: 0,
            time: this.$Utils.getDateString(),
            float: "left",
            userName: '系统',
        };
        this.pushMessageInRecord(JSON.stringify(msg))
    } else if(feedback === 2){
        let content = document.getElementById(commentId).value;
        if (!content || content.length < 1){
            this.$message({
                message: '请输入评价内容！ ',
                type: 'warning'
            });
            return;
        }else if(content.length > 64) {
            this.$message({
                message: '评价内容不能超过64字符！ ',
                type: 'warning'
            });
            return;
        }
        if (item.robotComment) {
            this.$Chat.feedbacksatisfaction(item.interIdx, 0,content);
        }else {
            this.$Chat.satisfactionInfo("1",content);
        }
        let msg = {
            avatar: "zph",
            text: "感谢您的反馈，我会努力改进的~",
            type: 0,
            time: this.$Utils.getDateString(),
            float: "left",
            userName: '系统',
        };
        this.pushMessageInRecord(JSON.stringify(msg))
        document.getElementById('b-'+item.commentId).style.display='none';
        document.getElementById(item.commentId).readOnly = true;
    } else {
        let msg = {
            avatar: "zph",
            text: "",
            type: 5,
            time: this.$Utils.getDateString(),
            float: "left",
            userName: '系统',
            leaveMessage: false,
            interIdx:item.interIdx,
            commentId:item.interIdx+this.$Utils.uuid(8,16)
        };
        if (item.isRobot) {
            msg["robotComment"] = true;
        }
        this.showCommentImg(item.commentImgId,dissatisfied);
        this.pushMessageInRecord(JSON.stringify(msg))
    }
}
```

满意会直接在聊天栏中输入一条信息：

```
if (feedback === 1) {
    if (item.isRobot) {
        this.$Chat.feedbacksatisfaction(item.interIdx, feedback,"ok");
    }else {
        this.$Chat.satisfactionInfo("5","ok");
    }
    this.showCommentImg(item.commentImgId,satisfied);
    let msg = {
        avatar: "zph",
        text: "感谢您的点赞，我会继续努力的~",
        type: 0,
        time: this.$Utils.getDateString(),
        float: "left",
        userName: '系统',
    };
    this.pushMessageInRecord(JSON.stringify(msg))
}
```

这边的满意分为对机器人的满意和对客服人员的满意，对机器人的满意会调用chat_feedbacksatisfaction中的接口，对人员满意调用saveSatisfaction接口。

不满意则弹出弹窗，需要输入不满意的原因。

```
else {
    let msg = {
        avatar: "zph",
        text: "",
        type: 5,
        time: this.$Utils.getDateString(),
        float: "left",
        userName: '系统',
        leaveMessage: false,
        interIdx:item.interIdx,
        commentId:item.interIdx+this.$Utils.uuid(8,16)
    };
    if (item.isRobot) {
        msg["robotComment"] = true;
    }
    this.showCommentImg(item.commentImgId,dissatisfied);
    this.pushMessageInRecord(JSON.stringify(msg))
}
```

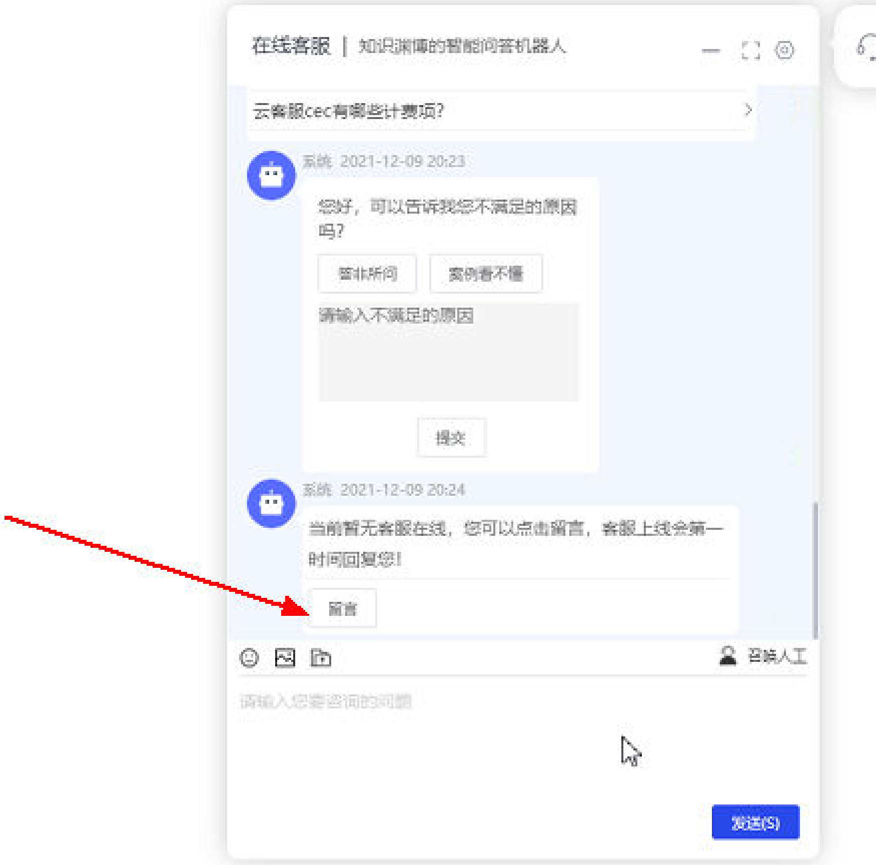
对应的样式代码。

```
<div v-if="item.type === 5" class="reason-style">
    <div>您好，可以告诉我您不满足的原因吗？</div>
    <div>
        <el-button size="small"
                   @click="pushInTextarea(item.commentId,'答非所问')">答非所问</el-button>
        <el-button size="small"
                   @click="pushInTextarea(item.commentId,'案例看不懂')">案例看不懂</el-button>
    </div>
    <div>
        <textarea :id="item.commentId" rows="4" placeholder="请输入不满足的原因"
                  class="reason-textarea"></textarea>
    </div>
    <div class="reason-submit" :id="'b-'+item.commentId">
        <el-button size="small" @click="feedbackSatisfaction(item,2,item.commentId);">提交</el-button>
    </div>
</div>
```

点击提交后再次进入到方法。

```
else if(feedback === 2){
    let content = document.getElementById(commentId).value;
    if (!content || content.length < 1){
        this.$message({
            message: '请输入评价内容！',
            type: 'warning'
        });
        return;
    }else if(content.length > 64) {
        this.$message({
            message: '评价内容不能超过64字符！',
            type: 'warning'
        });
        return;
    }
    if (item.robotComment) {
        this.$Chat.feedbacksatisfaction(item.interIdx, 0,content);
    }else {
        this.$Chat.satisfactionInfo("1",content);
    }
    let msg = {
        avatar: "zph",
        text: "感谢您的反馈，我会努力改进的~",
        type: 0,
        time: this.$Utils.getDateString(),
        float: "left",
        userName: '系统',
    };
    this.pushMessageInRecord(JSON.stringify(msg))
    document.getElementById('b-'+item.commentId).style.display='none';
    document.getElementById(item.commentId).readOnly = true;
}
```

12 代码使用示例-用户留言


在线客服 | 知识渊博的智能问答机器人

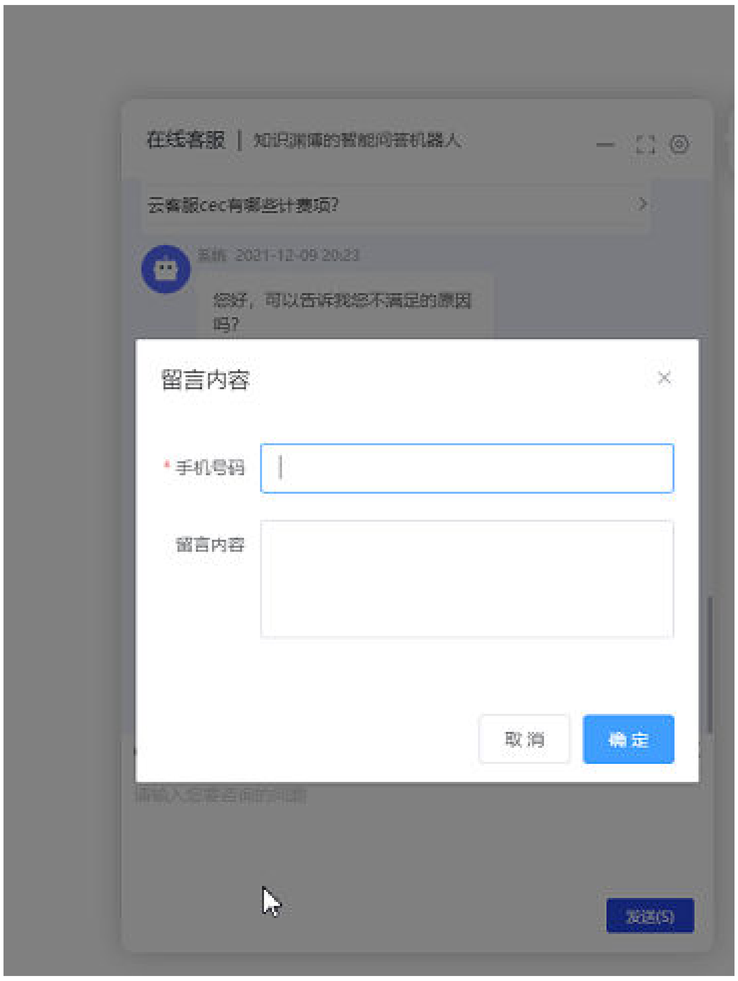
云客服cec有哪些计费项?

系统 2021-12-09 20:23

您好，可以告诉我您不满足的原因吗?

系统 2021-12-09 20:24

当前暂无客服在线，您可以点击留言，客服上线会第一时间回复您!



留言的前提条件是用户转人工失败了。参考用户转人工的方法失败时发送toAgentFailed事件。在MainContent.vue中存在监听方法：

```
//转人工失败
EventBus.$on("toAgentFailed", () => {
    this.toAgent = false;
    let msg = {
        avatar: "zph",
        text: "当前暂无客服在线，您可以点击留言，客服上线会第一时间回复您！ ",
        type: 0,
        time: this.$Utils.getDateString(),
        float: "left",
        userName: '系统',
        leaveMessage: true
    };
    this.pushMessageInRecord(JSON.stringify(msg))
});
```

该方法会向聊天框中推一条消息，满足以下代码规则，展示留言消息。

```
<div v-if="item.leaveMessage">
    <div class="line"></div>
    <el-button size="small" style="margin-top: 5px" @click="messagesVisible=true">留言
    </el-button>
</div>
```

点击留言按钮，可以修改messagesVisible属性展示的留言弹框：

```
<!-- 留言弹框 -->
<el-dialog
```

```
        title="留言内容"
        class="inner-dialog"
        :visible.sync="messagesVisible"
        :modal-append-to-body=false
        :close-on-click-modal=false
>
    <el-form :model="messageForm" ref="messageForm"   :rules='messagesRules'>
        <el-form-item label="手机号码" :label-width="formLabelWidth" required prop="phone">
            <el-input v-model="messageForm.phone" autocomplete="off"></el-input>
        </el-form-item>
        <el-form-item label="留言内容" :label-width="formLabelWidth" prop="message">
            <el-input resize="none" :rows="4" type="textarea" v-model="messageForm.message"
autocomplete="off"></el-input>
        </el-form-item>
    </el-form>
    <span slot="footer" class="dialog-footer">
        <el-button @click="messagesVisible = false">取 消</el-button>
        <el-button type="primary" @click="leaveMessage('messageForm')">确 定</el-button>
    </span>
</el-dialog>
```

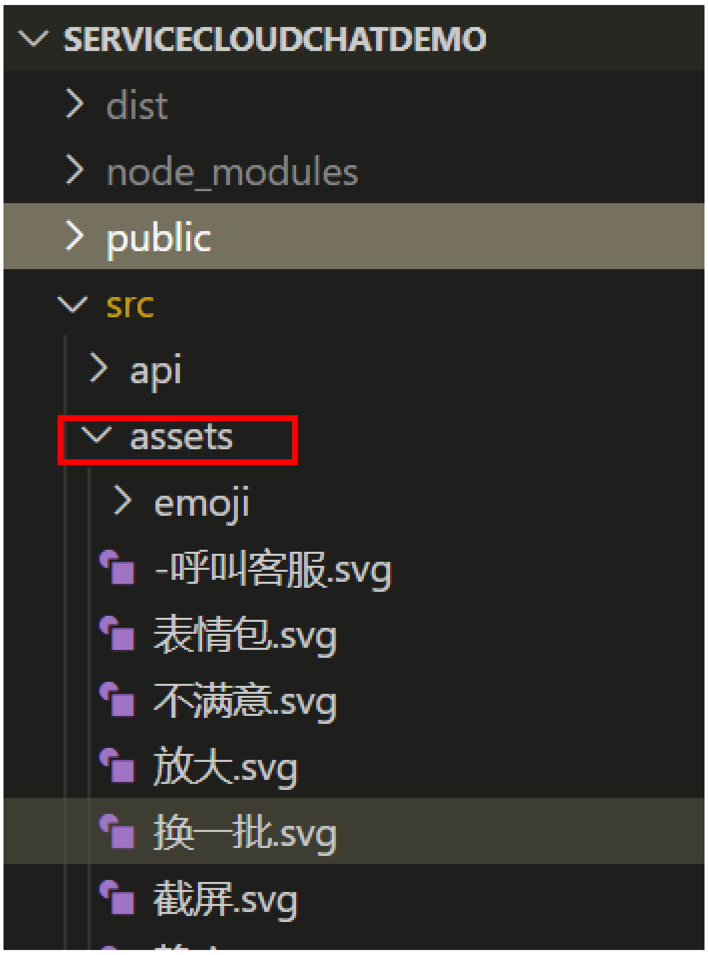
其中调用leaveMessage 方法留言，该方法会调用doLeaveMessage接口，发送留言

```
//留言
leaveMessage(messageForm) {
    this.$refs[messageForm].validate((valid) => {
        if (valid) {
            this.messagesVisible = false
            this.$Chat.doLeaveMessage(this.messageForm.phone,this.messageForm.message);
            this.messageForm.message = "";
        } else {
            return false;
        }
    });
}
```

13 定制实例

13.1 修改聊天的头像

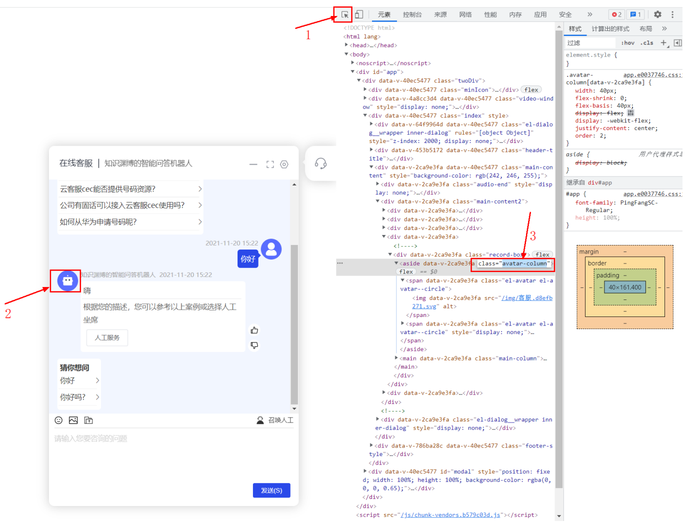
1.将打算替换的图片放到模板代码 ServiceCloudChatDemo\src\assets目录下，



```
∨ SERVICECLOUDCHATDEMO
  > dist
  > node_modules
  > public
  ∨ src
    > api
    ∨ assets
      > emoji
      -呼叫客服.svg
      表情包.svg
      不满意.svg
      放大.svg
      换一批.svg
      截屏.svg
```

2.用谷歌浏览器打开模板页面，键盘按F12打开调试页面，鼠标点击下图中的1指示的位置，然后点击2指示的地方，找到位置3所在类名"avatar-column",复制此类名到模板代码中全局搜索。

图 13-1 替换聊天头像 1

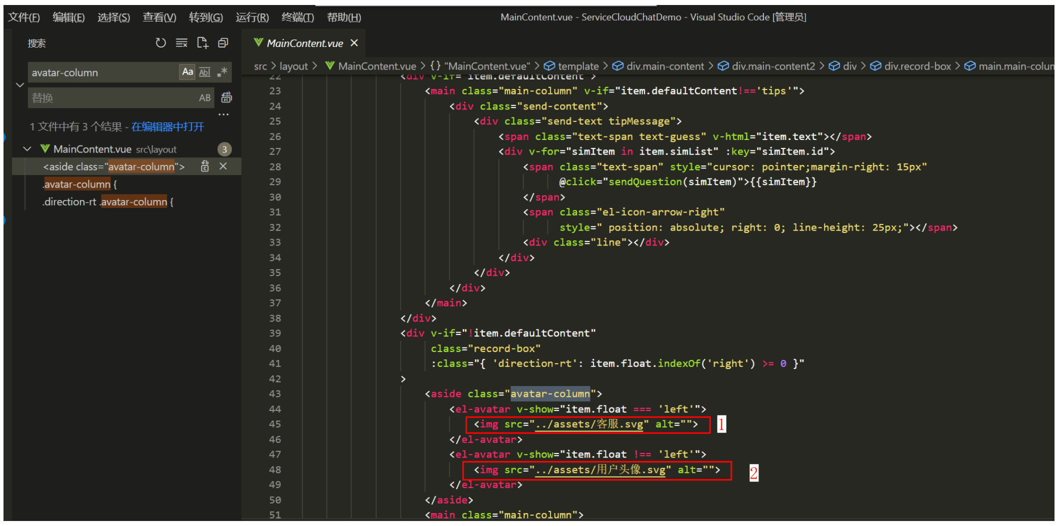


云客服cec能否提供号码资源?
公司有固话可以接入云客服cec使用吗?
如何从华为申请号码呢?
2021-11-20 15:22
你好
知识渊博的智能问答机器人 2021-11-20 15:22
嗨
根据您的描述，您可以参考以上案例或选择人工坐席
你好
你好吗?
请输入您要咨询的问题

```
<!DOCTYPE html>
<html lang>
 ▶<head>…</head>
 ▼<body>
   ▶<noscript>…</noscript>
   ▼<div id="app">
     ▼<div data-v-40ec5477 class="twoDiv">
       ▶<div data-v-40ec5477 class="minIcon">…</div> flex
       ▶<div data-v-4a8cc3d4 data-v-40ec5477 class="video-window" style="display: none;">…</div>
       ▼<div data-v-40ec5477 class="index" style>
         ▶<div data-v-64f9964d data-v-40ec5477 class="el-dialog__wrapper inner-dialog" rules="[object Object]" style="z-index: 2000; display: none;">…</div>
         ▶<div data-v-453b5172 data-v-40ec5477 class="header-title">…</div>
         ▼<div data-v-2ca9e3fa data-v-40ec5477 class="main-content" style="background-color: rgb(242, 246, 255);">
           ▶<div data-v-2ca9e3fa class="audio-end" style="display: none;">…</div>
           ▼<div data-v-2ca9e3fa class="main-content2">
             ▶<div data-v-2ca9e3fa>…</div>
             ▶<div data-v-2ca9e3fa>…</div>
             ▶<div data-v-2ca9e3fa>…</div>
             ▼<div data-v-2ca9e3fa>
                <!---->
               ▼<div data-v-2ca9e3fa class="record-box"> flex
  ...            ▼<aside data-v-2ca9e3fa class="avatar-column"> flex == $0
                   ▼<span data-v-2ca9e3fa class="el-avatar el-avatar--circle">
                      <img data-v-2ca9e3fa src="/img/客服.d8efb271.svg" alt>
                    </span>
                   ▶<span data-v-2ca9e3fa class="el-avatar el-avatar--circle" style="display: none;">…
                    </span>
                  </aside>
                 ▶<main data-v-2ca9e3fa class="main-column">…
                  </main>
                </div>
              </div>
             ▶<div data-v-2ca9e3fa>…</div>
            </div>
            <!---->
           ▶<div data-v-2ca9e3fa class="el-dialog__wrapper inner-dialog" style="display: none;">…</div>
          </div>
         ▶<div data-v-786ba28c data-v-40ec5477 class="footer-style">…</div>
        </div>
       ▶<div data-v-40ec5477 id="modal" style="position: fixed; width: 100%; height: 100%; background-color: rgba(0, 0, 0, 0.65);">…</div>
      </div>
    </div>
    <script src="/js/chunk-vendors.b579c03d.js"></script>
```

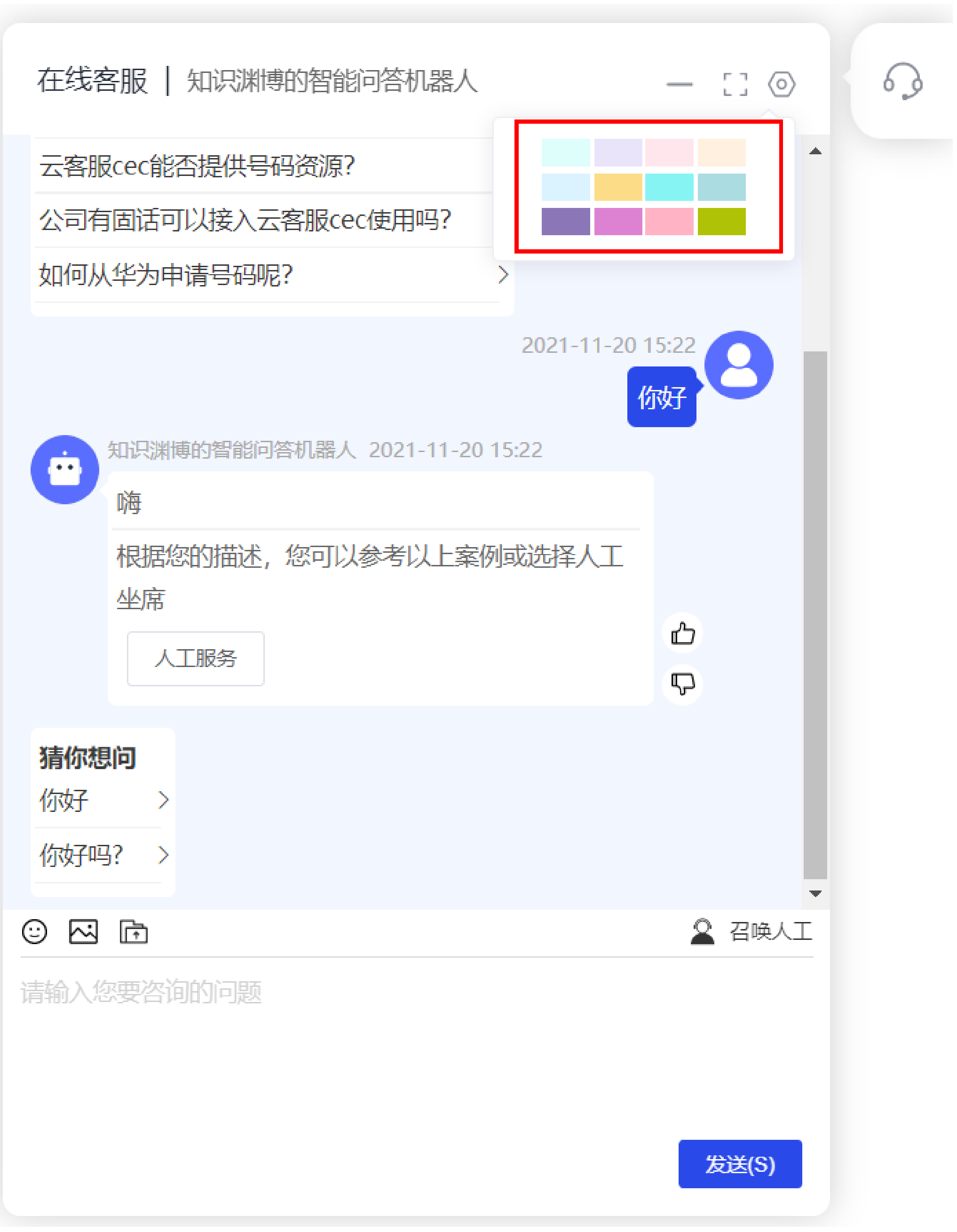
3.找到如下文件MainContent.vue，将位置1中的src=""里的图片路径替换成打算替换的图片的路径。

图 13-2 替换聊天头像 2

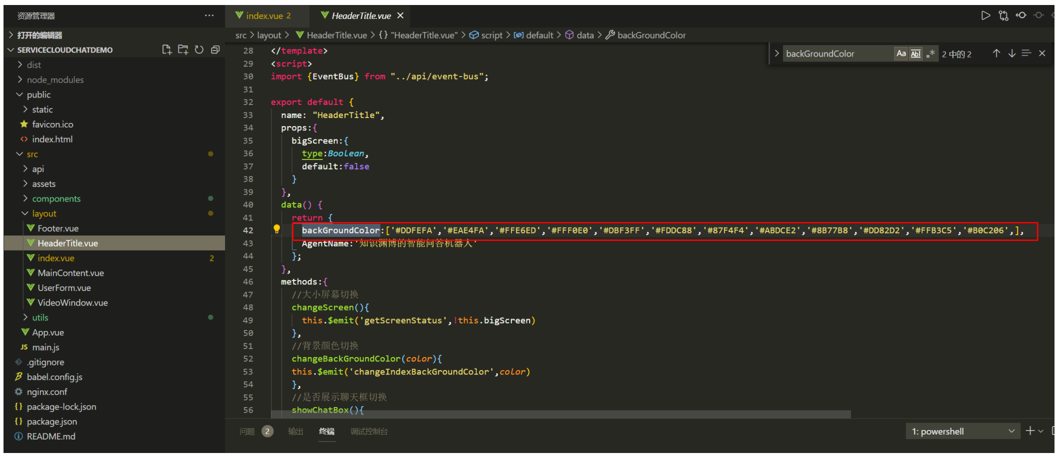

文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
MainContent.vue - ServiceCloudChatDemo - Visual Studio Code [管理员]
搜索
avatar-column
替换
1 文件中有 3 个结果 - 在编辑器中打开
MainContent.vue src\layout 3
<aside class="avatar-column">
.avatar-column {
.direction-rt .avatar-column {
MainContent.vue
src > layout > MainContent.vue > {} "MainContent.vue" > template > div.main-content > div.main-content2 > div > div.record-box > main.main-colun

```
            <main class="main-column" v-if="item.defaultContent!=='tips'">
                <div class="send-content">
                    <div class="send-text tipMessage">
                        <span class="text-span text-guess" v-html="item.text"></span>
                        <div v-for="simItem in item.simList" :key="simItem.id">
                            <span class="text-span" style="cursor: pointer;margin-right: 15px"
                                  @click="sendQuestion(simItem)">{{simItem}}
                            </span>
                            <span class="el-icon-arrow-right"
                                  style=" position: absolute; right: 0; line-height: 25px;"></span>
                            <div class="line"></div>
                        </div>
                    </div>
                </div>
            </main>
        </div>
        <div v-if="!item.defaultContent"
             class="record-box"
             :class="{ 'direction-rt': item.float.indexOf('right') >= 0 }"
        >
            <aside class="avatar-column">
                <el-avatar v-show="item.float === 'left'">
                    <img src="../assets/客服.svg" alt="">   1
                </el-avatar>
                <el-avatar v-show="item.float !== 'left'">
                    <img src="../assets/用户头像.svg" alt="">   2
                </el-avatar>
            </aside>
            <main class="main-column">
```

13.2 修改聊天背景颜色


在线客服 | 知识渊博的智能问答机器人
云客服cec能否提供号码资源?
公司有固话可以接入云客服cec使用吗?
如何从华为申请号码呢?
2021-11-20 15:22
你好
知识渊博的智能问答机器人 2021-11-20 15:22
嗨
根据您的描述，您可以参考以上案例或选择人工坐席
人工服务
猜你想问
你好
你好吗?
召唤人工
请输入您要咨询的问题
发送(S)

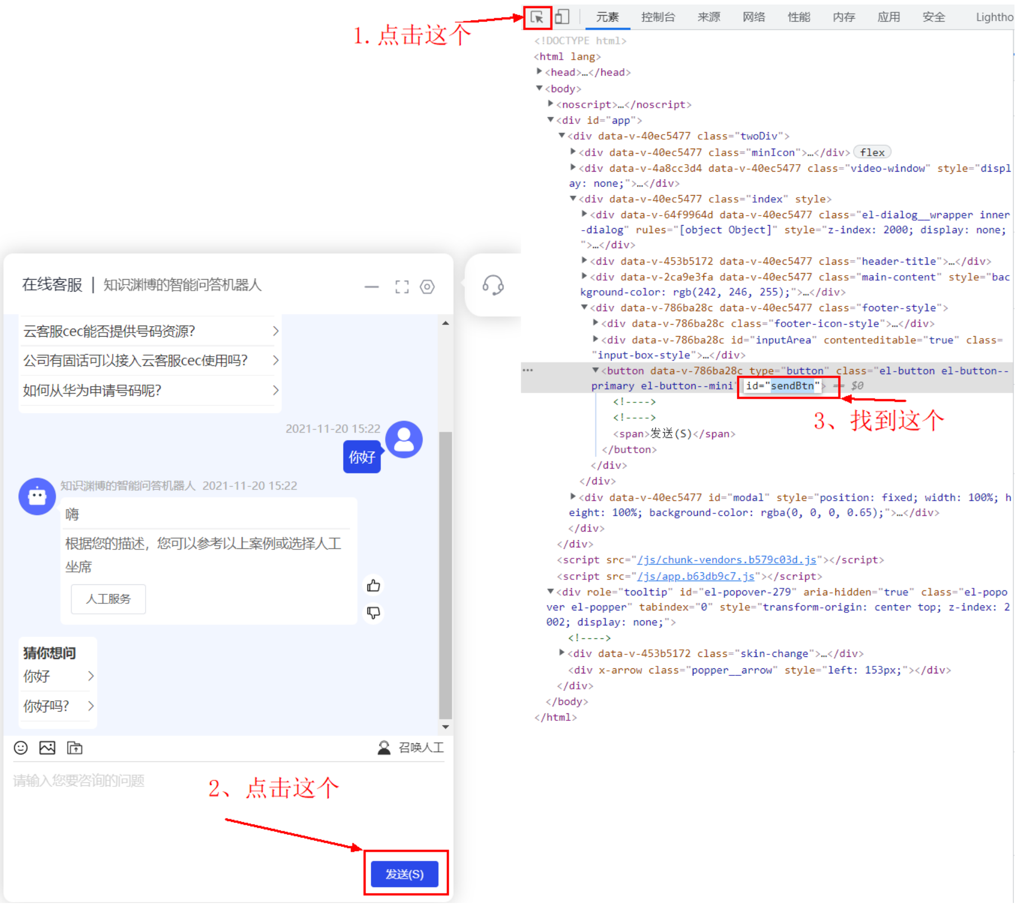
Demo已经有几个颜色，点击颜色框即可修改背景颜色，想要自定义颜色可以将下图中红框中的代码替换成想要的颜色。



```
</template>
<script>
import {EventBus} from "../api/event-bus";

export default {
  name: "HeaderTitle",
  props:{
    bigScreen:{
      type:Boolean,
      default:false
    }
  },
  data() {
    return {
      backGroundColor:['#DDFEFA','#EAE4FA','#FFE6ED','#FFF0E0','#DBF3FF','#FDDC88','#87F4F4','#ABDCE2','#8B77B8','#DD82D2','#FFB3C5','#B0C206',],
      AgentName:'知识渊博的智能问答机器人'
    };
  },
  methods:{
    //大小屏幕切换
    changeScreen(){
      this.$emit('getScreenStatus',!this.bigScreen)
    },
    //背景颜色切换
    changeBackGroundColor(color){
    this.$emit('changeIndexBackGroundColor',color)
    },
    //是否展示聊天框切换
    showChatBox(){
```

13.3 修改按钮样式

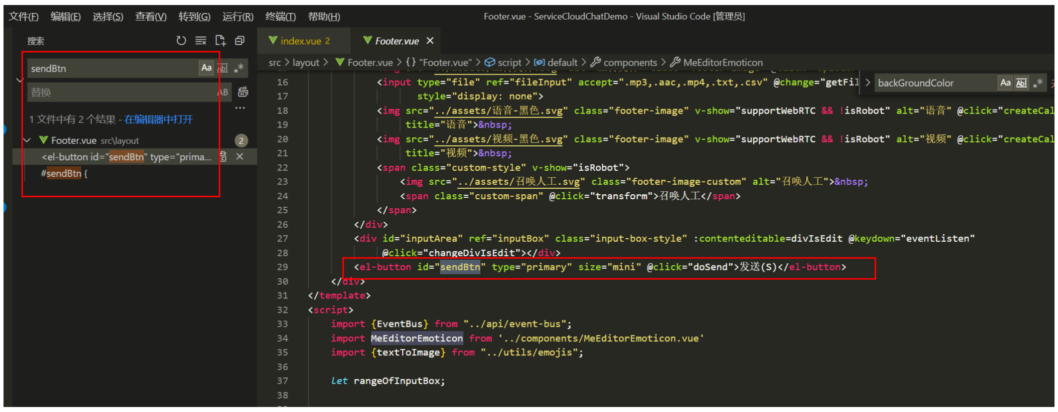
用谷歌浏览器打开模板页面，键盘按F12打开调试页面，鼠标点击下图中的1指示的位置，然后点击2指示的地方，找到3所在id "sendBtn",复制到模板代码中全局搜索，找到页面元素和样式代码，修改按钮样式代码。



```
<!DOCTYPE html>
<html lang>
▶<head>…</head>
▼<body>
  ▶<noscript>…</noscript>
  ▼<div id="app">
    ▼<div data-v-40ec5477 class="twoDiv">
      ▶<div data-v-40ec5477 class="minIcon">…</div> flex
      ▶<div data-v-4a8cc3d4 data-v-40ec5477 class="video-window" style="displ
      ay: none;">…</div>
      ▼<div data-v-40ec5477 class="index" style>
        ▶<div data-v-64f9964d data-v-40ec5477 class="el-dialog__wrapper inner
        -dialog" rules="[object Object]" style="z-index: 2000; display: none;
        ">…</div>
        ▶<div data-v-453b5172 data-v-40ec5477 class="header-title">…</div>
        ▶<div data-v-2ca9e3fa data-v-40ec5477 class="main-content" style="bac
        kground-color: rgb(242, 246, 255);">…</div>
        ▼<div data-v-786ba28c data-v-40ec5477 class="footer-style">
          ▶<div data-v-786ba28c class="footer-icon-style">…</div>
          ▶<div data-v-786ba28c id="inputArea" contenteditable="true" class=
          "input-box-style">…</div>
...       ▼<button data-v-786ba28c type="button" class="el-button el-button--
          primary el-button--mini" id="sendBtn"> == $0
              <!---->
              <!---->
              <span>发送(S)</span>
            </button>
          </div>
        </div>
      ▶<div data-v-40ec5477 id="modal" style="position: fixed; width: 100%; h
      eight: 100%; background-color: rgba(0, 0, 0, 0.65);">…</div>
      </div>
    </div>
    <script src="/js/chunk-vendors.b579c03d.js"></script>
    <script src="/js/app.b63db9c7.js"></script>
  ▼<div role="tooltip" id="el-popover-279" aria-hidden="true" class="el-popo
  ver el-popper" tabindex="0" style="transform-origin: center top; z-index: 2
  002; display: none;">
      <!---->
    ▶<div data-v-453b5172 class="skin-change">…</div>
      <div x-arrow class="popper__arrow" style="left: 153px;"></div>
    </div>
  </body>
</html>
```

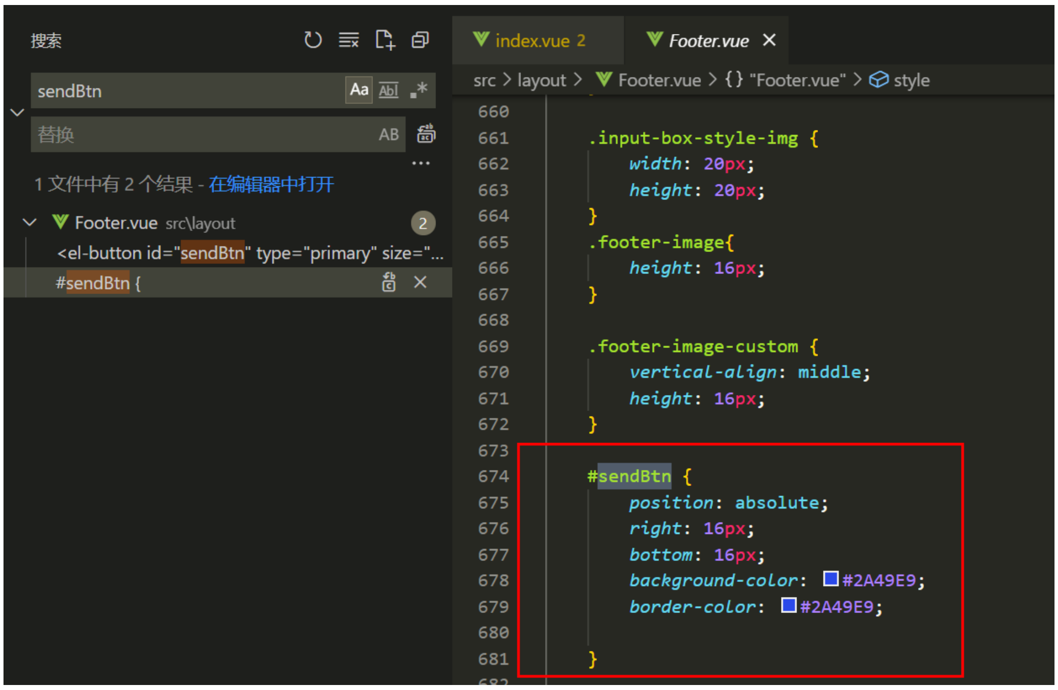
云客服cec能否提供号码资源?
公司有固话可以接入云客服cec使用吗?
如何从华为申请号码呢?
2021-11-20 15:22
你好
知识渊博的智能问答机器人 2021-11-20 15:22
嗨
根据您的描述，您可以参考以上案例或选择人工坐席
你好
你好吗?

图 13-3 页面元素代码


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
Footer.vue - ServiceCloudChatDemo - Visual Studio Code [管理员]
搜索
sendBtn
替换
1 文件中有 2 个结果 - 在编辑器中打开
Footer.vue src\layout
<el-button id="sendBtn" type="prima...
#sendBtn {
index.vue 2
Footer.vue
src > layout > Footer.vue > {} "Footer.vue" > script > default > components > MeEditorEmoticon
backGroundColor

```
        <input type="file" ref="fileInput" accept=".mp3,.aac,.mp4,.txt,.csv" @change="getFil
              style="display: none">
        <img src="../assets/语音-黑色.svg" class="footer-image" v-show="supportWebRTC && !isRobot" alt="语音" @click="createCal
            title="语音">&nbsp;
        <img src="../assets/视频-黑色.svg" class="footer-image" v-show="supportWebRTC && !isRobot" alt="视频" @click="createCal
            title="视频">&nbsp;
        <span class="custom-style" v-show="isRobot">
            <img src="../assets/召唤人工.svg" class="footer-image-custom" alt="召唤人工">&nbsp;
            <span class="custom-span" @click="transform">召唤人工</span>
        </span>
      </div>
      <div id="inputArea" ref="inputBox" class="input-box-style" :contenteditable=divIsEdit @keydown="eventListen"
          @click="changeDivIsEdit"></div>
      <el-button id="sendBtn" type="primary" size="mini" @click="doSend">发送(S)</el-button>
    </div>
</template>
<script>
    import {EventBus} from "../api/event-bus";
    import MeEditorEmoticon from '../components/MeEditorEmoticon.vue'
    import {textToImage} from "../utils/emojis";

    let rangeOfInputBox;

```

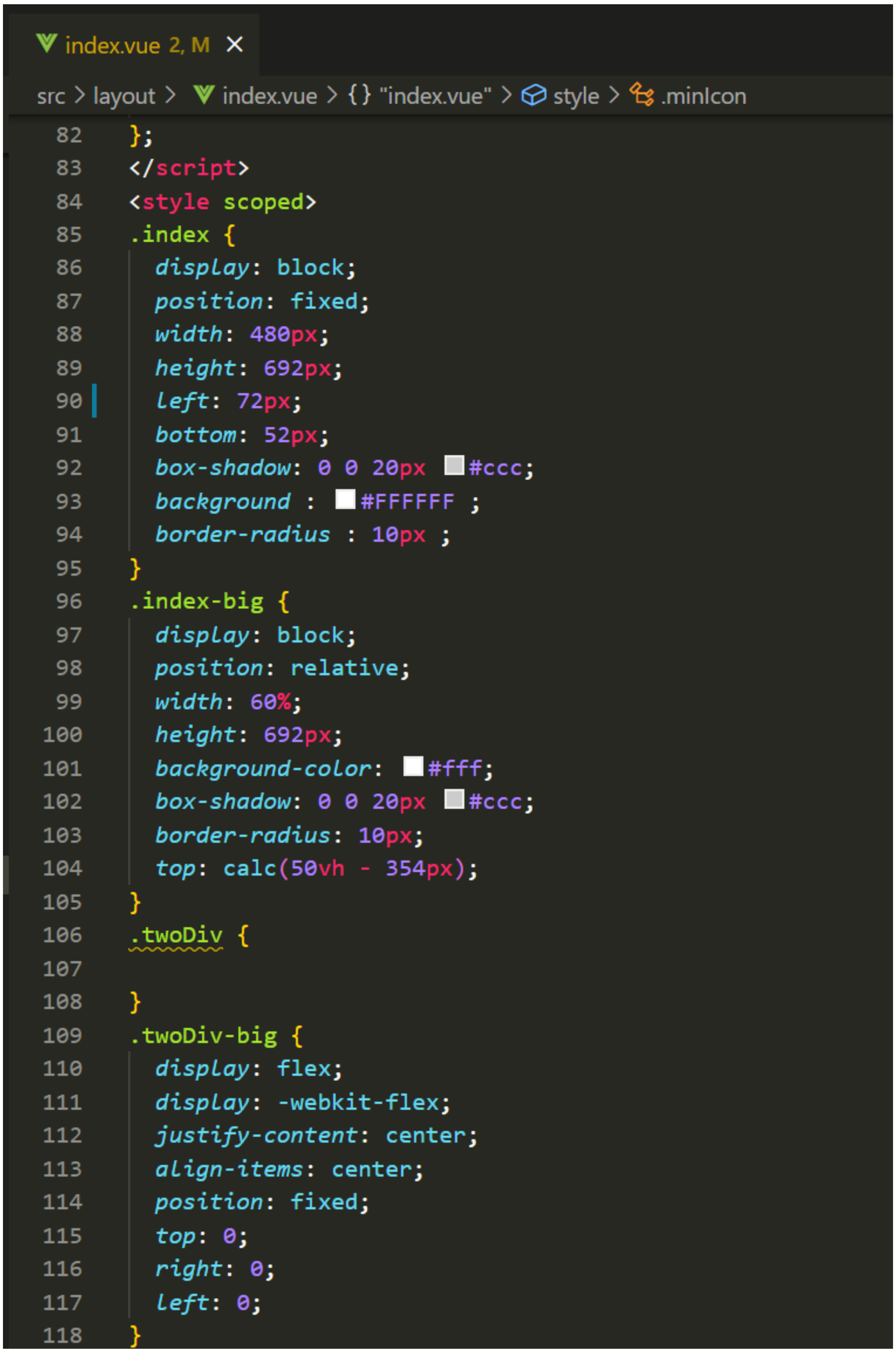
图 13-4 按钮样式代码


搜索
sendBtn
替换
1 文件中有 2 个结果 - 在编辑器中打开
Footer.vue src\layout
<el-button id="sendBtn" type="primary" size="...
#sendBtn {
index.vue 2
Footer.vue
src > layout > Footer.vue > {} "Footer.vue" > style

```

    .input-box-style-img {
        width: 20px;
        height: 20px;
    }
    .footer-image{
        height: 16px;
    }

    .footer-image-custom {
        vertical-align: middle;
        height: 16px;
    }

    #sendBtn {
        position: absolute;
        right: 16px;
        bottom: 16px;
        background-color: #2A49E9;
        border-color: #2A49E9;

    }
```

13.4 修改页面位置

聊天窗口使用绝对定位，修改位置的代码在ServiceCloudChatDemo\src\layout \index.vue文件的<style scoped>标签里。

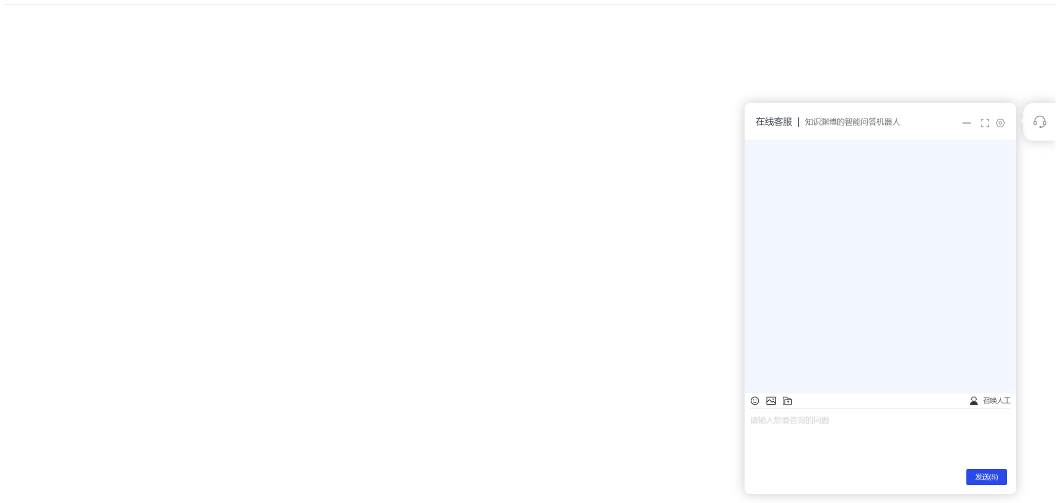

index.vue 2, M
src > layout > index.vue > {} "index.vue" > style > .minIcon

```
};
</script>
<style scoped>
.index {
  display: block;
  position: fixed;
  width: 480px;
  height: 692px;
  left: 72px;
  bottom: 52px;
  box-shadow: 0 0 20px #ccc;
  background : #FFFFFF ;
  border-radius : 10px ;
}
.index-big {
  display: block;
  position: relative;
  width: 60%;
  height: 692px;
  background-color: #fff;
  box-shadow: 0 0 20px #ccc;
  border-radius: 10px;
  top: calc(50vh - 354px);
}
.twoDiv {

}
.twoDiv-big {
  display: flex;
  display: -webkit-flex;
  justify-content: center;
  align-items: center;
  position: fixed;
  top: 0;
  right: 0;
  left: 0;
}
```

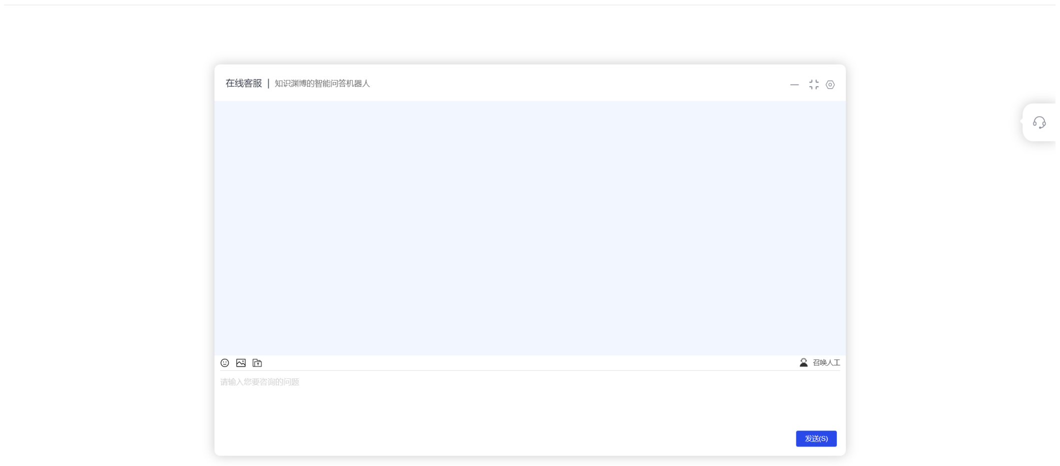
聊条窗口有三个形态1.最小化时不显示。2.小窗口居于右下角显示。3.最大化时居中显示。

- 最小化

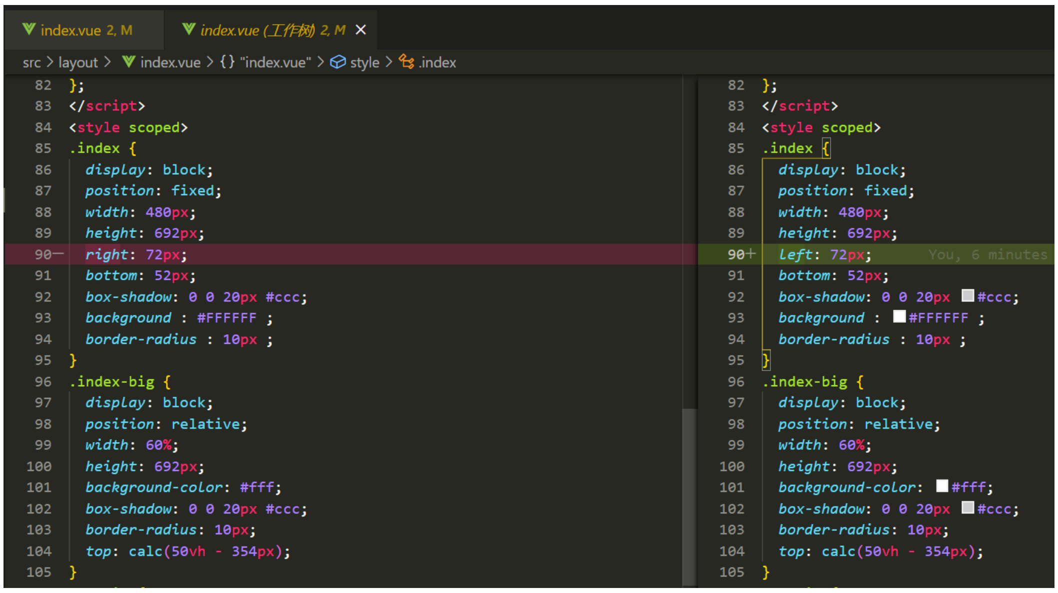
- 小窗口


在线客服 | 知识渊博的智能问答机器人
召唤人工
请输入您要咨询的问题
发送(S)

- 最大化


在线客服 | 知识渊博的智能问答机器人
召唤人工
请输入您要咨询的问题
发送(S)

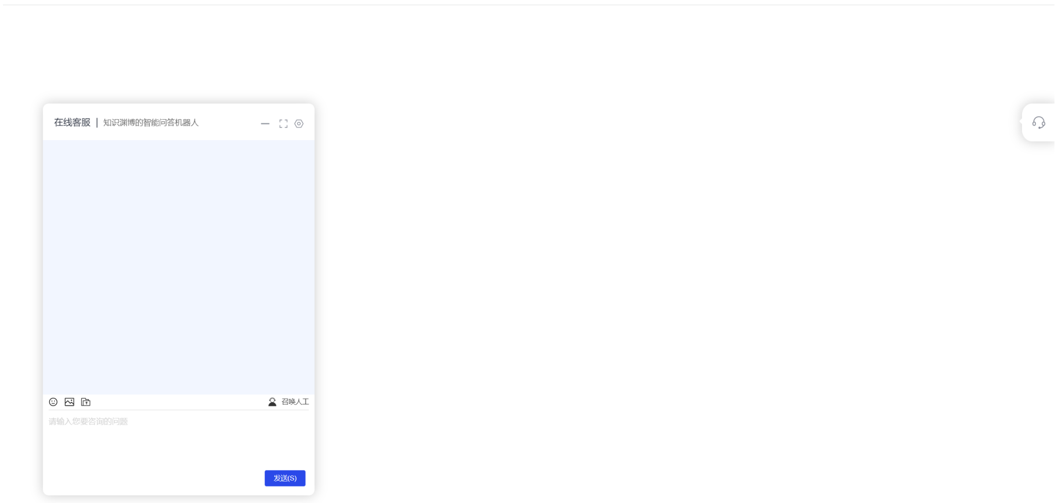
其中 .index 里写的小窗口在右下角时的样式，.index-big 和 .twoDiv-big里写的最大化居中显示的样式。如果想要窗口在左下角显示，按照下面修改



```
};
</script>
<style scoped>
.index {
  display: block;
  position: fixed;
  width: 480px;
  height: 692px;
  right: 72px;
  bottom: 52px;
  box-shadow: 0 0 20px #ccc;
  background : #FFFFFF ;
  border-radius : 10px ;
}
.index-big {
  display: block;
  position: relative;
  width: 60%;
  height: 692px;
  background-color: #fff;
  box-shadow: 0 0 20px #ccc;
  border-radius: 10px;
  top: calc(50vh - 354px);
}
```

```
};
</script>
<style scoped>
.index {
  display: block;
  position: fixed;
  width: 480px;
  height: 692px;
  left: 72px;           You, 6 minutes
  bottom: 52px;
  box-shadow: 0 0 20px #ccc;
  background : #FFFFFF ;
  border-radius : 10px ;
}
.index-big {
  display: block;
  position: relative;
  width: 60%;
  height: 692px;
  background-color: #fff;
  box-shadow: 0 0 20px #ccc;
  border-radius: 10px;
  top: calc(50vh - 354px);
}
```

效果如下：



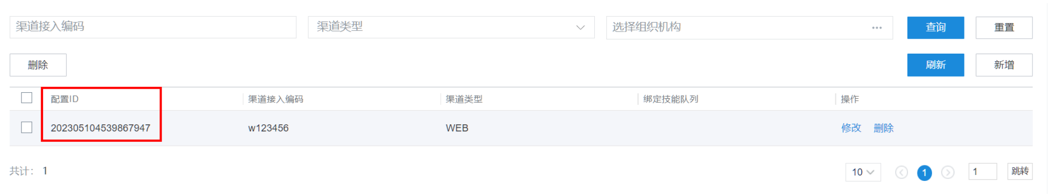
14 测试与验证

步骤1 在未修改Demo的情况下，请在/src/api/config.js中配置。

```
let chatConfig = {
    appKey:'xxxxx',
    appSecret:'xxxxx',
    channelId:'xxxxx',
    lang:'zh'
}
```

appKey和appSecret 对应apifabric接口的aksk，channelId是需要对接的渠道ID。之前需要完成渠道配置，确定渠道配置了机器人：

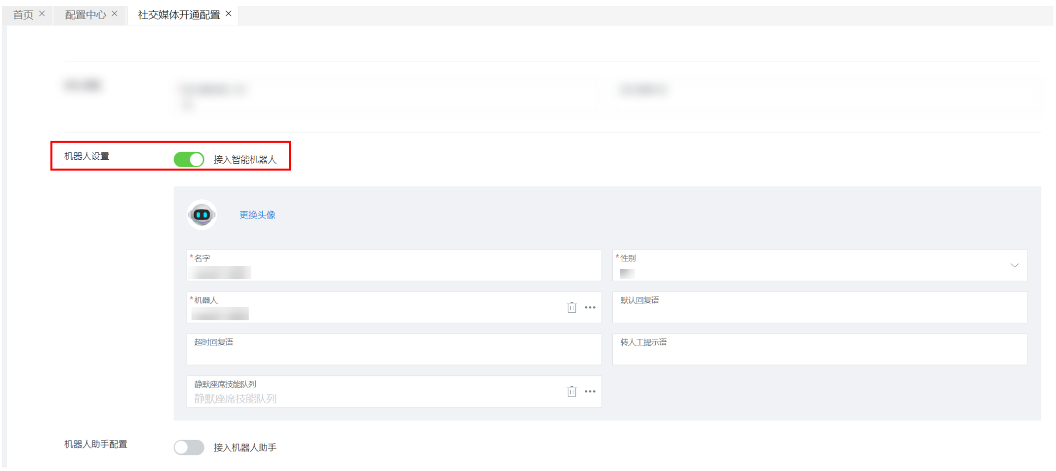
渠道ID获取如下：



	配置ID	渠道接入编码	渠道类型	绑定技能队列	操作
☐	202305104539867947	w123456	WEB		修改 删除

共计： 1

点击“修改”，确认该渠道是否配置了智能机器人。



机器人设置 接入智能机器人

*名字

*性别

*机器人

默认回复语

超时回复语

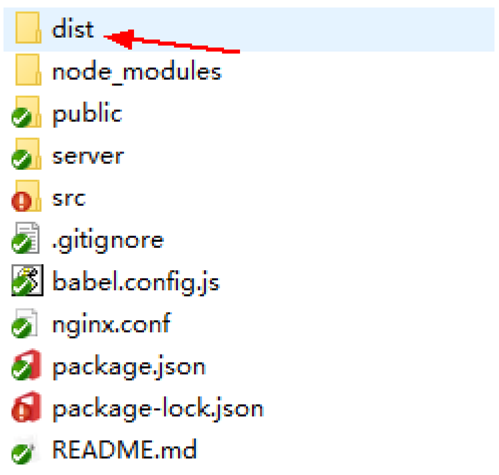
转人工提示语

静默座席技能队列 静默座席技能队列

机器人助手配置 接入机器人助手

步骤2 安装node.js，完成安装后到package.json的同级目录下，先执行npm install；完成后执行npm run build。

执行完成后，会生成一个dist目录。注意该目录的路径。



步骤3 下载nginx，windows版本的即可，下载完成后找到nginx的目录，在其目录中找到conf/nginx.conf，编辑修改。

```
worker_processes  1;

events {
    worker_connections  1024;
}

http {
    include       mime.types;
    default_type text/plain;
    charset UTF-8;
    sendfile        on;

    keepalive_timeout  65;

    server {
        listen       18082 ssl;
        server_name  localhost;

        ssl_certificate      D:/nginx/keys/server.crt;#证书路径
        ssl_certificate_key      D:/nginx/keys/server.key;#key路径
        ssl_session_cache    shared:SSL:60m; #s储存SSL会话的缓存类型和大小
        ssl_session_timeout 60m; #会话过期时间

        location / {
            root   D:/servicecloudDevelop/servicecloud/aicc-tool/ServiceCloudChatDemo/dist; #构建工程的dist目
录
            index  index.html index.htm;
        }

        location /apigovernance {
            client_max_body_size 200m;
            proxy_pass   https://10.21.119.148:28090/apigovernance;  #配置https://aicc服务地址/apigovernance
        }

        location /apiaccess {
            client_max_body_size 200m;
            proxy_pass   https://10.21.119.148:28090/apiaccess; #配置https://aicc服务地址/apigovernance
        }
    }
}
```

其中root的目录，修改成npm run build 生成的dist目录路径。

```
        location / {
            root   D:/servicecloudDevelop/servicecloud/aicc-tool/ServiceCloudChatDemo/dist; #构建工程的dist目
```

```
录
            index  index.html index.htm;
        }
```

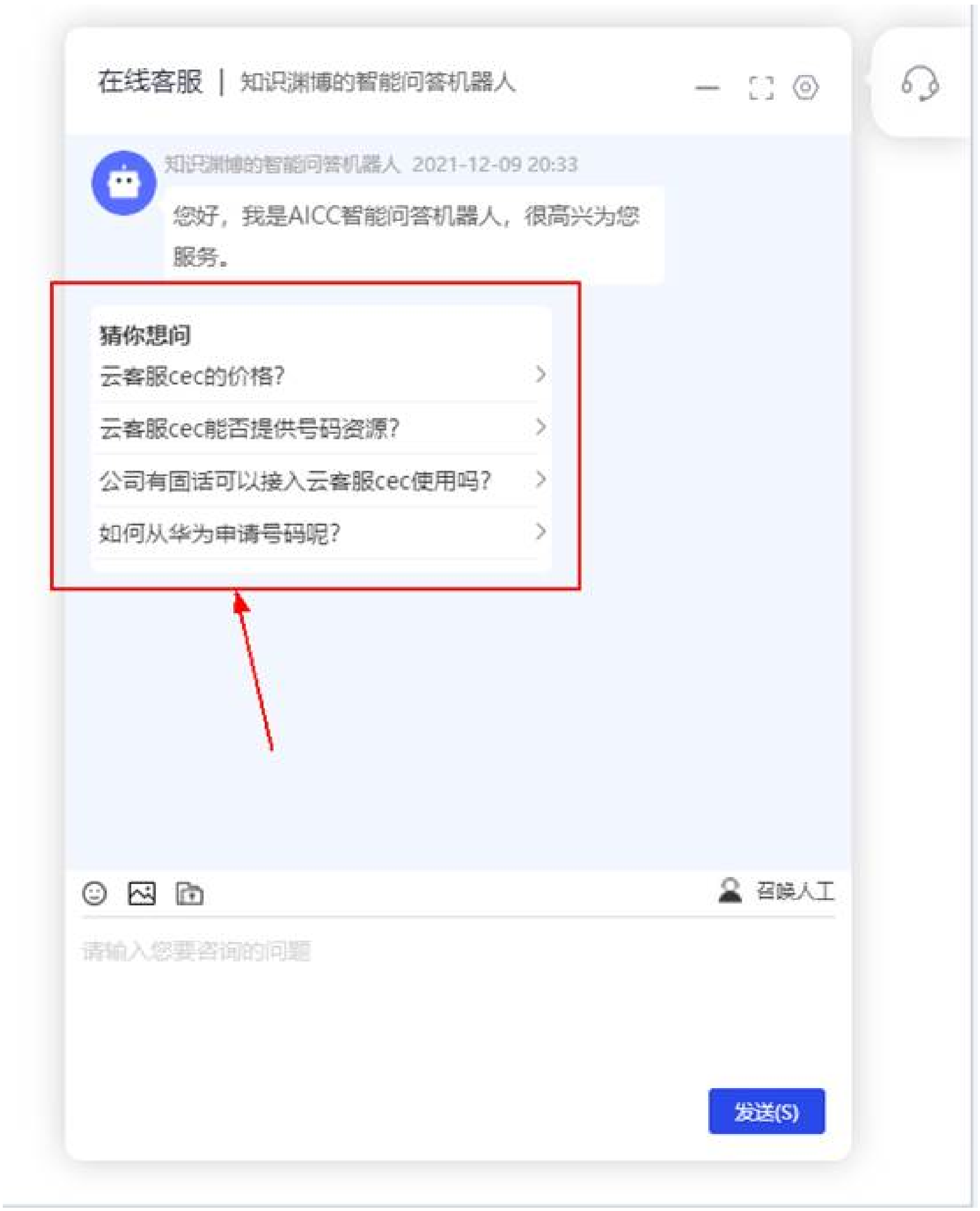
加上ssl证书路径。可以使用自签名生成的证书，生成方式可以自行搜索“自签名证书”。

```
ssl_certificate      D:/nginx/keys/server.crt;#证书路径
        ssl_certificate_key      D:/nginx/keys/server.key;#key路径
```

----结束

15 注意事项

由于是纯前台的Demo，该demo中存在很多硬编码的提示语句，用户需要修改的话可根据自身业务自行修改。例如用户接入的“猜你想问”。


在线客服 | 知识渊博的智能问答机器人

知识渊博的智能问答机器人 2021-12-09 20:33

您好，我是AICC智能问答机器人，很高兴为您服务。

猜你想问

云客服cec的价格?

云客服cec能否提供号码资源?

公司有固话可以接入云客服cec使用吗?

如何从华为申请号码呢?

在代码MainContent.vue中，同样的，还存在对于机器人回复的一些语句，如果不需要评价，可以放入在相关List中。

```
//接入时聊天框默认展示的问题
questionDefaultList: [
    "云客服cec的价格？", "云客服cec能否提供号码资源？", "公司有固话可以接入云客服cec使用吗？", "如何从华为申请号码呢？"
],
//不需要进行评价的内容
defaultMessageList: ["会话转人工处理。", "会话超时结束。", "您好，我是AICC智能问答机器人，很高兴为您服务。"]
```