# 尚云超融合JS-SDK

目录

[尚云超融合JS-SDK 1](#_Toc504411140)

[概述 3](#_Toc504411141)

[使用说明 3](#_Toc504411142)

[步骤一：引入JS文件 3](#_Toc504411143)

[步骤二：通过config接口注入权限验证配置 3](#_Toc504411144)

[步骤三：通过ready接口处理成功验证 3](#_Toc504411145)

[步骤四：通过error接口处理失败验证 3](#_Toc504411146)

[接口调用说明 4](#_Toc504411147)

[基础接口 4](#_Toc504411148)

[判断当前客户端版本是否支持指定JS接口 4](#_Toc504411149)

[分享接口 4](#_Toc504411150)

[获取“转发”按钮点击状态及自定义分享内容接口 4](#_Toc504411151)

[获取“微信”按钮点击状态及自定义分享内容接口 4](#_Toc504411152)

[获取“QQ”按钮点击状态及自定义分享内容接口 5](#_Toc504411153)

[获取“微博”按钮点击状态及自定义分享内容接口 5](#_Toc504411154)

[获取“收藏”按钮点击状态及自定义收藏内容接口 5](#_Toc504411155)

[获取“邮件”按钮点击状态及自定义邮件内容接口 5](#_Toc504411156)

[获取“浏览器打开”按钮点击状态及自定义浏览器打开内容接口 6](#_Toc504411157)

[音频接口 6](#_Toc504411158)

[开始录音接口 6](#_Toc504411159)

[停止录音接口 6](#_Toc504411160)

[监听录音自动停止接口 6](#_Toc504411161)

[播放语音接口 6](#_Toc504411162)

[暂停播放接口 6](#_Toc504411163)

[停止播放接口 6](#_Toc504411164)

[监听语音播放完毕接口 6](#_Toc504411165)

[上传语音接口 7](#_Toc504411166)

[下载语音接口 7](#_Toc504411167)

[语音识别接口 7](#_Toc504411168)

[图像接口 7](#_Toc504411169)

[拍照或从手机相册中选图接口 7](#_Toc504411170)

[预览图片接口 7](#_Toc504411171)

[上传图片接口 8](#_Toc504411172)

[下载图片接口 8](#_Toc504411173)

[文件接口 8](#_Toc504411174)

[选择文件接口 8](#_Toc504411175)

[预览文件接口 8](#_Toc504411176)

[上传文件接口 8](#_Toc504411177)

[下载文件接口 8](#_Toc504411178)

[设备信息 9](#_Toc504411179)

[获取网络状态接口 9](#_Toc504411180)

[地理位置 9](#_Toc504411181)

[使用融合客户端内置地图查看位置接口 9](#_Toc504411182)

[获取地理位置接口 9](#_Toc504411183)

[界面操作 9](#_Toc504411184)

[监听页面返回事件 9](#_Toc504411185)

[隐藏右上角菜单接口 9](#_Toc504411186)

[显示右上角菜单接口 9](#_Toc504411187)

[关闭当前网页窗口接口 10](#_Toc504411188)

[批量隐藏功能按钮接口 10](#_Toc504411189)

[批量显示功能按钮接口 10](#_Toc504411190)

[隐藏所有非基础按钮接口 10](#_Toc504411191)

[显示所有功能按钮接口 10](#_Toc504411192)

[应用内跳转 10](#_Toc504411193)

[融合客户端扫一扫 10](#_Toc504411194)

[调起扫一扫接口 10](#_Toc504411195)

[通讯录接口 11](#_Toc504411196)

[通讯录选人接口 11](#_Toc504411197)

[获取用户信息接口 11](#_Toc504411198)

[会话接口 11](#_Toc504411199)

[打开用户会话接口 11](#_Toc504411200)

[打开群会话接口 11](#_Toc504411201)

[创建并打开群会话接口 12](#_Toc504411202)

[打开应用会话接口 12](#_Toc504411203)

[优先实现的接口 12](#_Toc504411204)

[附录1-JS-SDK使用权限签名算法(未实现，可暂时忽略) 12](#_Toc504411205)

[附录2-所有JS接口列表 13](#_Toc504411206)

[附录3-所有菜单项列表 14](#_Toc504411207)

[附录4-常见错误及解决方法 14](#_Toc504411208)

## 概述

尚云超融合JS-SDK是尚云超融合平台面向网页开发者提供的基于融合客户端内的网页开发工具包。

通过使用超融合JS-SDK，网页开发者可借助融合客户端高效地使用拍照、选图、语音、位置等手机或PC系统的能力，同时可以直接使用分享、扫一扫等特有的能力，为用户提供更优质的网页体验。

此文档面向网页开发者介绍超融合JS-SDK如何使用及相关注意事项。

## 使用说明

### 步骤一：引入JS文件

在需要调用JS接口的页面引入sr.js文件，获取通过<script src=”sr.js文件的网络地址”>方式使用。

### 步骤二：通过config接口注入权限验证配置

**所有需要使用JS-SDK的页面必须先注入配置信息，否则将无法调用。**

1. sr.config({
2. beta: true,// 必须这么写
3. debug: true, // 开启调试模式,调用的所有api的返回值会在客户端alert出来
4. appId: '', // 必填，企业应用管理平台的应用ID
5. timestamp: , // 必填，可填随机值，目前暂时不做校验
6. nonceStr: '', // 必填，可填随机值，目前暂时不做校验
7. signature: '',// 必填，可填随机值，目前暂时不做校验
8. jsApiList: [] // 必填，需要使用的JS接口列表，所有JS接口列表见附录2
9. });

### 步骤三：通过ready接口处理成功验证

1. sr.ready(function(){
2. // config信息验证后会执行ready方法，所有接口调用都必须在config接口获得结果之后，config是一个客户端的异步操作，所以如果需要在页面加载时就调用相关接口，则须把相关接口放在ready函数中调用来确保正确执行。对于用户触发时才调用的接口，则可以直接调用，不需要放在ready函数中。
3. });

### 步骤四：通过error接口处理失败验证

1. sr.error(function(res){
2. // config信息验证失败会执行error函数，如签名过期导致验证失败，具体错误信息可以打开config的debug模式查看，也可以在返回的res参数中查看，对于SPA可以在这里更新签名。
3. });

### ­接口调用说明

所有接口通过sr对象来调用，参数是一个对象，除了每个接口本身需要传的参数之外，还有以下通用参数：

1. success：接口调用成功时执行的回调函数。
2. fail：接口调用失败时执行的回调函数。
3. complete：接口调用完成时执行的回调函数，无论成功或失败都会执行。
4. cancel：用户点击取消时的回调函数，仅部分有用户取消操作的api才会用到。
5. trigger: 监听Menu中的按钮点击时触发的方法，该方法仅支持Menu中的相关接口。

**注意：不要尝试在trigger中使用ajax异步请求修改本次分享的内容，因为客户端分享操作是一个同步操作，这时候使用ajax的回包会还没有返回**。

以上几个函数都带有一个参数，类型为对象，其中除了每个接口本身返回的数据之外，还有一个通用属性errMsg，其值格式如下：

1. 调用成功时：”xxx:ok” ，其中xxx为调用的接口名
2. 用户取消时：”xxx:cancel”，其中xxx为调用的接口名
3. 调用失败时：其值为具体错误信息

## 基础接口

### 判断当前客户端版本是否支持指定JS接口

1. sr.checkJsApi({
2. jsApiList: ['chooseImage'], // 需要检测的JS接口列表，所有JS接口列表见附录2,
3. success: function(res) {
4. // 以键值对的形式返回，可用的api值true，不可用为false
5. // 如：{"checkResult":{"chooseImage":true},"errMsg":"checkJsApi:ok"}
6. }

## 分享接口

### 获取“转发”按钮点击状态及自定义分享内容接口

1. sr.onMenuShareAppMessage({
2. title: '', // 分享标题
3. desc: '', // 分享描述
4. link: '', // 分享链接
5. imgUrl: '', // 分享图标
6. success: function () {
7. // 用户确认分享后执行的回调函数
8. },
9. cancel: function () {
10. // 用户取消分享后执行的回调函数
11. }
12. });

### 获取“微信”按钮点击状态及自定义分享内容接口

1. sr.onMenuShareWechat({
2. title: '', // 分享标题
3. desc: '', // 分享描述
4. link: '', // 分享链接
5. imgUrl: '', // 分享图标
6. success: function () {
7. // 用户确认分享后执行的回调函数
8. },
9. cancel: function () {
10. // 用户取消分享后执行的回调函数
11. }

### 获取“QQ”按钮点击状态及自定义分享内容接口

1. sr.onMenuShareQQ({
2. title: '', // 分享标题
3. desc: '', // 分享描述
4. link: '', // 分享链接
5. imgUrl: '', // 分享图标
6. success: function () {
7. // 用户确认分享后执行的回调函数
8. },
9. cancel: function () {
10. // 用户取消分享后执行的回调函数

### 获取“微博”按钮点击状态及自定义分享内容接口

1. sr.onMenuShareWeibo({
2. title: '', // 分享标题
3. desc: '', // 分享描述
4. link: '', // 分享链接
5. imgUrl: '', // 分享图标
6. success: function () {
7. // 用户确认分享后执行的回调函数
8. },
9. cancel: function () {
10. // 用户取消分享后执行的回调函数
11. }

### 获取“收藏”按钮点击状态及自定义收藏内容接口

1. sr.onMenuFavorite({
2. title: '', // 收藏标题
3. desc: '', // 收藏描述
4. link: '', // 收藏链接
5. imgUrl: '', // 图标
6. success: function () {
7. // 用户确认收藏后执行的回调函数
8. },
9. cancel: function () {
10. // 用户取消收藏后执行的回调函数

### 获取“邮件”按钮点击状态及自定义邮件内容接口

1. sr.onMenuMail({
2. title: '', //邮件标题
3. link: '', // 邮件链接
4. success: function () {
5. // 用户确认发送邮件后执行的回调函数
6. },
7. cancel: function () {
8. // 用户取消发送邮件后执行的回调函数

### 获取“浏览器打开”按钮点击状态及自定义浏览器打开内容接口

1. sr.onMenuOpenByBrowser({
2. link: '', // 收藏链接
3. success: function () {
4. // 用户确认浏览器打开后执行的回调函数
5. },
6. cancel: function () {
7. // 用户取消浏览器打开后执行的回调函数

## 音频接口

### 开始录音接口

1. sr.startRecord();

### 停止录音接口

1. sr.stopRecord({
2. success: function (res) {
3. var localId = res.localId;
4. }
5. });

### 监听录音自动停止接口

1. sr.onVoiceRecordEnd({
2. // 录音时间超过一分钟没有停止的时候会执行 complete 回调
3. complete: function (res) {
4. var localId = res.localId;
5. }
6. });

### 播放语音接口

1. sr.playVoice({
2. localId: '' // 需要播放的音频的本地ID，由stopRecord接口获得
3. });

### 暂停播放接口

1. sr.pauseVoice({
2. localId: '' // 需要暂停的音频的本地ID，由stopRecord接口获得
3. });

### 停止播放接口

1. sr.stopVoice({
2. localId: '' // 需要停止的音频的本地ID，由stopRecord接口获得
3. });

### 监听语音播放完毕接口

1. sr.onVoicePlayEnd({
2. success: function (res) {
3. var localId = res.localId; // 返回音频的本地ID
4. }
5. });

### 上传语音接口

1. sr.uploadVoice({
2. localId: '', // 需要上传的音频的本地ID，由stopRecord接口获得
3. isShowProgressTips: 1, // 默认为1，显示进度提示
4. success: function (res) {
5. var serverId = res.serverId; // 返回音频的服务器端ID
6. }
7. });

### 下载语音接口

1. sr.downloadVoice({
2. serverId: '', // 需要下载的音频的服务器端ID，由uploadVoice接口获得
3. isShowProgressTips: 1, // 默认为1，显示进度提示
4. success: function (res) {
5. var localId = res.localId; // 返回音频的本地ID
6. }

### 语音识别接口

1. sr.downloadVoice({
2. localId: voice.localId,
3. complete: function (res) {
4. if (res.hasOwnProperty('translateResult')) {
5. alert('识别结果：' + res.translateResult);
6. } else {
7. alert('无法识别');
8. }
9. }
10. });

## ­图像接口

### 拍照或从手机相册中选图接口

1. sr.chooseImage({
2. count: 1, // 默认9
3. sizeType: ['original', 'compressed'], // 可以指定是原图还是压缩图，默认二者都有
4. sourceType: ['album', 'camera'], // 可以指定来源是相册还是相机，默认二者都有
5. success: function (res) {
6. var localIds = res.localIds; // 返回选定照片的本地ID列表，localId可以作为img标签的src属性显示图片
7. }
8. });

### 预览图片接口

1. sr.previewImage({
2. current: '', // 当前显示图片的http链接
3. urls: [] // 需要预览的图片http链接列表
4. });

### 上传图片接口

1. sr.uploadImage({
2. localId: '', // 需要上传的图片的本地ID，由chooseImage接口获得
3. isShowProgressTips: 1, // 默认为1，显示进度提示
4. success: function (res) {
5. var serverId = res.serverId; // 返回图片的服务器端ID
6. }
7. });

### 下载图片接口

1. sr.downloadImage({
2. serverId: '', // 需要下载的图片的服务器端ID，由uploadImage接口获得
3. isShowProgressTips: 1, // 默认为1，显示进度提示
4. success: function (res) {
5. var localId = res.localId; // 返回图片下载后的本地ID
6. }

## 文件接口

### 选择文件接口

1. sr.chooseFile
2. ({
3. success: function (res) {
4. files.localId = res.localIds;
5. alert('已选择 ' + res.localIds.length + ' 个文件');
6. }
7. });

### 预览文件接口

1. sr.previewFile({
2. url: '', // 需要预览文件的地址(必填，可以使用相对路径)
3. name: '', // 需要预览文件的文件名(不填的话取url的最后部分)
4. size: 1048576 // 需要预览文件的字节大小(必填)
5. });

### 上传文件接口

1. sr.uploadFile({
2. localId: '', // 需要上传的文件的本地ID，由chooseFile接口获得
3. isShowProgressTips: 1, // 默认为1，显示进度提示
4. success: function (res) {
5. var serverId = res.serverId; // 返回文件的服务器端ID
6. }
7. });

### 下载文件接口

1. sr.downloadFile({
2. serverId: '', // 需要下载的文件的服务器端ID，由uploadFile接口获得
3. isShowProgressTips: 1, // 默认为1，显示进度提示
4. success: function (res) {
5. var localId = res.localId; // 返回文件下载后的本地ID
6. }

## 设备信息

### 获取网络状态接口

1. sr.getNetworkType({
2. success: function (res) {
3. var networkType = res.networkType; // 返回网络类型2g，3g，4g，wifi
4. }
5. });

## 地理位置

### 使用融合客户端内置地图查看位置接口

1. sr.openLocation({
2. latitude: 0, // 纬度，浮点数，范围为90 ~ -90
3. longitude: 0, // 经度，浮点数，范围为180 ~ -180。
4. name: '', // 位置名
5. address: '', // 地址详情说明
6. scale: 1, // 地图缩放级别,整形值,范围从1~28。默认为16
7. });

### 获取地理位置接口

1. sr.getLocation({
2. success: function (res) {
3. var latitude = res.latitude; // 纬度，浮点数，范围为90 ~ -90
4. var longitude = res.longitude; // 经度，浮点数，范围为180 ~ -180。
5. var speed = res.speed; // 速度，以米/每秒计
6. var accuracy = res.accuracy; // 位置精度
7. }
8. });

## 界面操作

### 监听页面返回事件

**接口说明：**该接口可在用户返回上个页面时，回调开发者注册的函数，处理业务需要的逻辑（如确认或重定向到指定的页面）

1. sr.onHistoryBack(function(){
2. return confirm('确定要放弃当前页面的修改？')
3. });

响应函数须返回布尔值false或true。false表示中断此次返回操作，否则继续执行返回操作。当页面左上角没有关闭按钮（即已是顶级页面），不产生该事件，强制执行返回。OS系统使用手势返回时，不产生该事件，强制执行返回

### 隐藏右上角菜单接口

1. sr.hideOptionMenu();

### 显示右上角菜单接口

1. sr.showOptionMenu();

### 关闭当前网页窗口接口

1. sr.closeWindow();

### 批量隐藏功能按钮接口

1. sr.hideMenuItems({
2. menuList: [] // 要隐藏的菜单项，所有menu项见附录3
3. });

### 批量显示功能按钮接口

1. sr.showMenuItems({
2. menuList: [] // 要显示的菜单项，所有menu项见附录3
3. });

### 隐藏所有非基础按钮接口

1. sr.hideAllNonBaseMenuItem();

### 显示所有功能按钮接口

1. sr.showAllNonBaseMenuItem();

### 应用内跳转

1. sr.redirectInApp({
2. schema:”dfs://\*\*\*\*\*”//必填
3. });

## 融合客户端扫一扫

### 调起扫一扫接口

1. sr.scanQRCode({
2. desc: 'scanQRCode desc',
3. needResult: 0, // 默认为0，扫描结果由融合客户端处理，1则直接返回扫描结果，
4. scanType: ["qrCode", "barCode"], // 可以指定扫二维码还是一维码，默认二者都有
5. success: function(res) {
6. // 回调
7. },
8. error: function(res) {
9. if (res.errMsg.indexOf('function\_not\_exist') > 0) {
10. alert('版本过低请升级')
11. }
12. }
13. });

## 通讯录接口

### 通讯录选人接口

1. sr.invoke("selectContact", {
2. "mode": "single",// 必填，选择模式，single表示单选，multi表示多选
3. "selectedUserIds": ["user1","user2"]// 非必填，已选用户ID列表。用于多次选人时可重入
4. },function(res){
5. if (res.err\_msg == "selectContact:ok")
6. {
7. var selectedUserList = res.result.userList; // 已选的成员列表
8. for (var i = 0; i < selectedUserList.length; i++)
9. {
10. var user = selectedUserList[i];
11. var userId = user.id; // 已选的单个成员ID
12. var userName = user.name;// 已选的单个成员名称
13. var userAvatar= user.avatar;// 已选的单个成员头像
14. }
15. }
16. }
17. );

### 获取用户信息接口

1. sr.invoke("getUserInfo", {
2. "userId": "要获取的用户Id",
3. },function(res){
4. if (res.err\_msg == "selectContact:ok")
5. {
6. var userInfo = res.result.userInfo;
7. var userId = userInfo.id; // 用户ID
8. var userName = userInfo.name;//用户名称
9. var userAvatar= userInfo.avatar;// 用户头像
10. }
11. }
12. }
13. );

## 会话接口

### 打开用户会话接口

1. sr.openUserChat({
2. userId: '要聊天的用户ID', // 必填
3. success: function(res) {
4. // 回调
5. },
6. fail: function(res) {
7. // 回调
8. }
9. });

### 打开群会话接口

1. sr.openGroupChat({
2. groupId: '要聊天的群ID', // 必填
3. success: function(res) {
4. // 回调
5. },
6. fail: function(res) {
7. // 回调
8. }
9. });

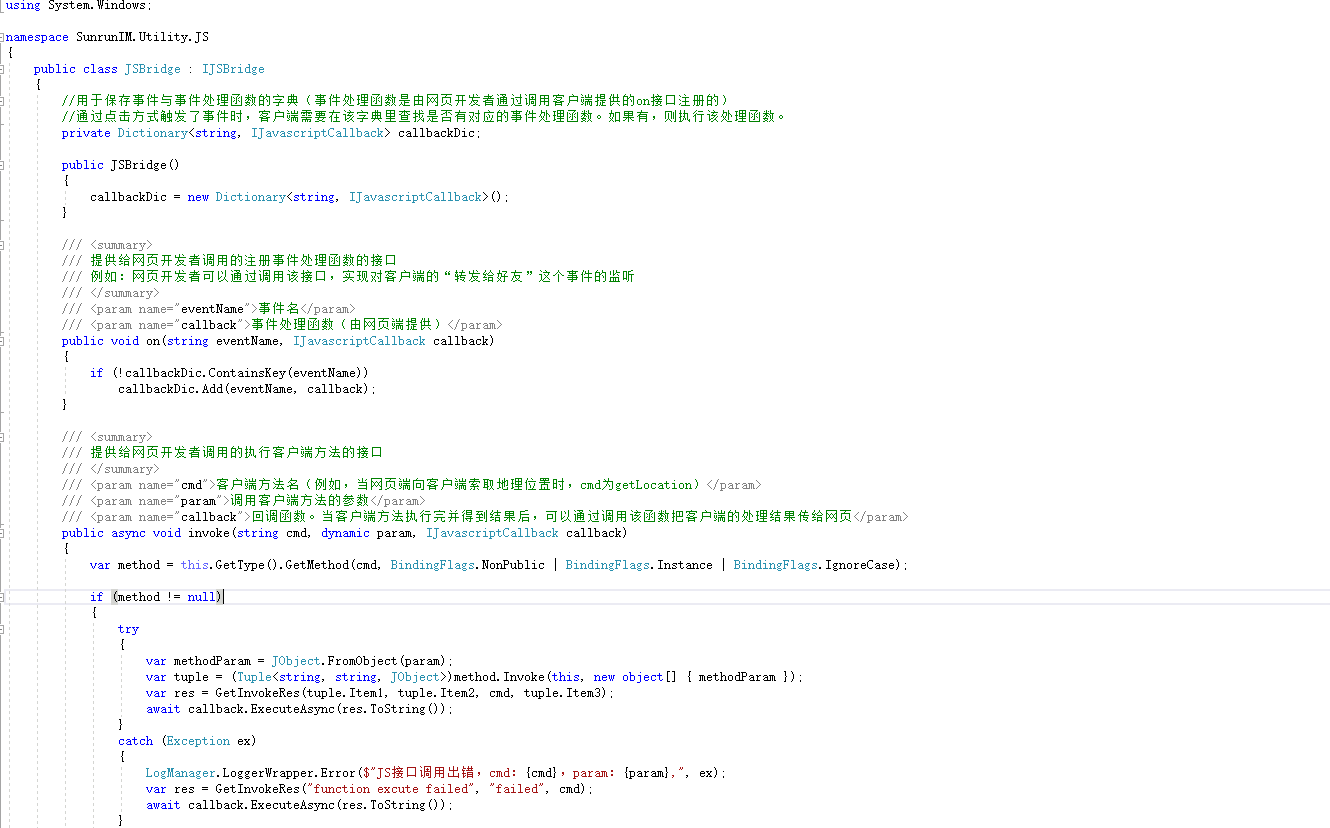
### 创建并打开群会话接口

1. sr.buildAndOpenGroupChat({
2. groupName: '群主题’,
3. UserIds:[‘czq’,’yss’,’cdl’,’hxc’],
4. success: function(res) {
5. // 回调
6. },
7. fail: function(res) {
8. // 回调
9. }

### 打开应用会话接口

1. sr.openAppChat({
2. appId: '要聊天的应用ID', // 必填
3. success: function(res) {
4. // 回调
5. },
6. fail: function(res) {
7. // 回调
8. }
9. });

## 优先实现的接口

1. 上传文件接口
2. 获取地理位置接口
3. 通讯录选人接口
4. 获取用户信息接口
5. 打开用户会话接口
6. PC客户端实现代码（仅供参考）：

## 附录1-JS-SDK使用权限签名算法(未实现，可暂时忽略)

签名算法

签名生成规则如下：参与签名的字段包括noncestr（随机字符串）, 有效的jsapi\_ticket, timestamp（时间戳）, url（当前网页的URL，不包含#及其后面部分） 。对所有待签名参数按照字段名的ASCII 码从小到大排序（字典序）后，使用URL键值对的格式 （即 key1=value1&key2=value2…）拼接成字符串string1。这里需要注意的是所有参数名均为小写字符。对string1作sha1加密，字段名和字段值都采用原始值，不进行URL 转义。

即signature=sha1(string1)。 示例

1. noncestr=Wm3WZYTPz0wzccnW
2. jsapi\_ticket=sM4AOVdWfPE4DxkXGEs8VMCPGGVi4C3VM0P37wVUCFvkVAy\_90u5h9nbSlYy3-Sl-HhTdfl2fzFy1AOcHKP7qg
3. timestamp=1414587457
4. url=http://mp.weixin.qq.com?params=value

步骤1. 对所有待签名参数按照字段名的ASCII 码从小到大排序（字典序）后，使用URL键值对的格式（即key1=value1&key2=value2…）拼接成字符串string1：

1. jsapi\_ticket=sM4AOVdWfPE4DxkXGEs8VMCPGGVi4C3VM0P37wVUCFvkVAy\_90u5h9nbSlYy3-Sl-HhTdfl2fzFy1AOcHKP7qg&noncestr=Wm3WZYTPz0wzccnW&timestamp=1414587457&url=http://mp.weixin.qq.com?params=value

步骤2. 对string1进行sha1签名，得到signature：

1. 0f9de62fce790f9a083d5c99e95740ceb90c27ed

注意事项

1. 签名用的noncestr和timestamp必须与sr.config中的nonceStr和timestamp相同。
2. 签名用的url必须是调用JS接口页面的完整URL。
3. 出于安全考虑，开发者必须在服务器端实现签名的逻辑。

如出现invalid signature 等错误详见附录4常见错误及解决办法。

## 附录2-所有JS接口列表

* onMenuShareAppMessage
* onMenuShareWechat
* onMenuShareQQ
* onMenuShareWeibo
* onMenuFavorite
* onMenuMail
* onMenuOpenByBrowser
* startRecord
* stopRecord
* onVoiceRecordEnd
* playVoice
* pauseVoice
* stopVoice
* onVoicePlayEnd
* uploadVoice
* downloadVoice
* translateVoice
* chooseImage
* previewImage
* uploadImage
* downloadImage
* chooseFile
* previewFile
* uploadFile
* downloadFile
* getNetworkType
* openLocation
* getLocation
* onHistoryBack
* hideOptionMenu
* showOptionMenu
* hideMenuItems
* showMenuItems
* hideAllNonBaseMenuItem
* showAllNonBaseMenuItem
* redirectInApp
* closeWindow
* scanQRCode
* selectContact
* openUserChat
* openGroupChat
* buildAndOpenGroupChat
* openAppChat

## 附录3-所有菜单项列表

基本类

* 调整字体: “menuItem:setFont”
* 刷新: “menuItem:refresh”

扩展类

* 转发: “menuItem:share:appMessage”
* 微信: “menuItem:share:wechat”
* QQ : “menuItem:share:qq”
* 微博: “menuItem:share:weibo”
* 收藏: “menuItem:favorite”
* 复制链接: “menuItem:copyUrl”
* 邮件: “menuItem:share:email”
* 浏览器打开: “menuItem:openByBrowser”

## 附录4-常见错误及解决方法

调用config接的时候传入参数debug: true 可以开启debug模式，页面会alert出错误信息。以下为常见错误及解决方法：。

1. the permission value is offline verifying：这个错误是因为config没有正确执行，或者是调用的JSAPI没有传入config的jsApiList参数中。建议按如下顺序检查：
   1. 确认config正确通过。
   2. 如果是在页面加载好时就调用了JSAPI，则必须写在sr.ready的回调中。
   3. 确认config的jsApiList参数包含了这个JSAPI。
2. permission denied：该应用没有权限使用这个接口。
3. function not exist：当前客户端版本不支持该接口，请升级到新版体验。
4. uploadImage怎么传多图：目前只支持一次上传一张，多张图片需等前一张图片上传之后再调用该接口。
5. 没法对本地选择的图片进行预览：chooseImage接口本身就支持预览，不需要额外支持。
6. 出现config:fail错误：这是由于传入的config参数不全导致，请确保传入正确的appId、timestamp、nonceStr、signature和需要使用的jsApiList。