Web 开发文档

ChangeLog

Version 1.3

1、SDK 支持 ie7、8、9
2、支持自定义消息类型
3、支持自定义群通知

Version 1.2

1、支持播放其他终端上发的语音消息，下载文件
2、支持显示群提示消息
3、支持群系统消息通知

Version 1.1

1、demo 登录支持 TLS（托管模式）
2、增加群组功能
3、支持发图

Version 1.0

1、用户下线
2、C2C 消息收发（文本、表情）
3、用户关系链管理（好友，黑名单管理）
4、用户资料管理（昵称、性别，加好友设置）

1. 概述

1.1. ImSDK 集成

本节主要介绍如何集成 ImSDK。
更多集成文档请参考官方 wiki 文档，地址如下：
http://avc.qcloud.com/wiki2.0/im/%E5%AE%A2%E6%88%B7%E7%AB%AF%E9%9B%86%E6%88%90/Web%E5%AE%A2%E6%88%B7%E7%AB%AF%E9%9B%86%E6%88%90/1%20%E6%A6%82%E8%BF%B0/1%20%E6%A6%82%E8%BF%B0.html

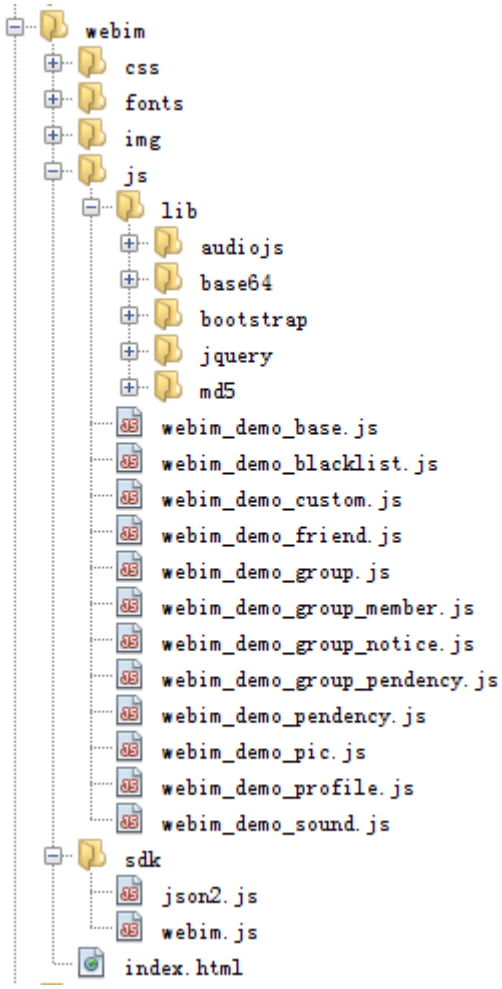
1.1.1. 下载 ImSDK

从官网下载 ImSDK：包含以下库文件：
sdk/webim.js
sdk/json2.js
其中 json2.js 提供了 json 的序列化和反序列化方法，可以将一个 json 对象转换成 json 字符串，也可以将一个 json 字符串转换成一个 json 对象。webim.js 就是 webim sdk 库，提供了聊天，群组管理，资料管理，关系链（好友，黑名单）管理功能。
此外，上传图片时，需要先获取图片的 md5，用到了下面的第三方 js 库：

```
<!--用于获取文件 MD5，上传图片需要先获取文件的 MD5-->
<script type="text/javascript" src="js/lib/md5/spark-md5.js"></script>
```

1.1.2. 集成 ImSDK

将步骤 1.1.1 下载得到的库文件引入到前台页面中，这里拿 demo 举例，webim sdk 和 demo 目录结构如下，



1. demo 中的 index.html 引入以下 js 文件：

   ```
   <script type="text/javascript" src="sdk/json2.js"></script>
   <!--web im sdk-->
   <script type="text/javascript" src="sdk/webim.js"></script>
   <!--用于获取文件 MD5，上传图片需要先获取文件的 MD5-->
   <script type="text/javascript" src="js/lib/md5/spark-md5.js"></script>
   ```

2. 如果你的帐号采用的是**托管模式**，则需要引入腾讯登录服务（Tencent Login Service，TLS）

   js 文件，如下：

   ```
   <!--引入腾讯登录服务 TLS web sdk-->
   <script type="text/javascript" src="https://tls.qcloud.com/libs/api.min.js"></script>
   ```

   详细介绍请参考第 2.1 小节。

3. 在 index.html 修改 TLS 登录成功之后的回调地址 callBackUrl，假设第三方应用的首页地址 http://webim.server.com/demo/index.html，则 callBackUrl 地址需要改成：

   ```
   //TLS 登录成功回调地址，也就是第三方应用的首页地址,需要业务自己定义
   var callBackUrl='http://webim.server.com/demo/index.html';
   ```

说明：如果帐号采用的是*独立模式*，请忽略以上第 2、3 点介绍，开发者需要在自己的服务器调用 TLS API 生成用户票据，然后调用 ImSdk 提供的接口进行相关操作。

1.1.3. 功能开发

在工程中引入上述 1.1.2 提及的库文件，根据后续章节的开发指引进行功能的开发。其中函数调用顺序可参见（1.2.2 调用顺序介绍）。

1.1.4. 支持版本

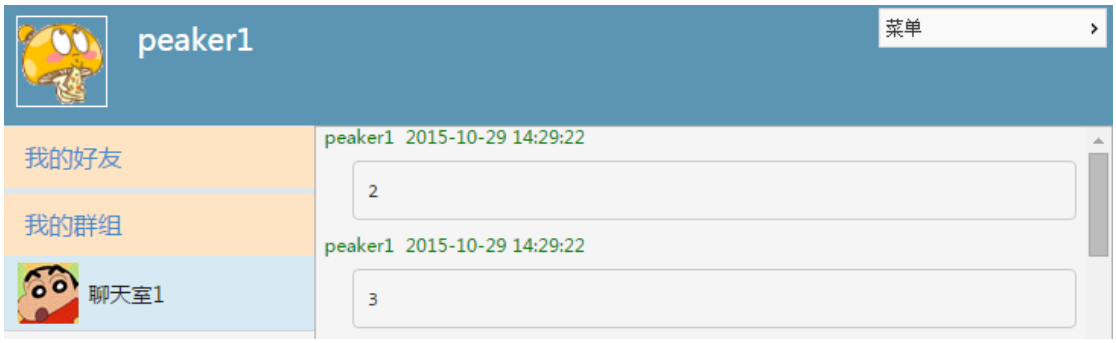
ImSDK 支持 IE 7+，Chrome 7+，FireFox 3.6+，Opera 12+和 Safari 6+。
Demo 支持 IE 8+，Chrome 7+，FireFox 3.6+，Opera 12+和 Safari 6+。

1.2. ImSDK 基本概念

会话：ImSDK 中会话(Session)分为两种，一种是 C2C 会话，表示单聊情况，自己与对方建立的对话；另一种是群会话，表示群聊情况下，群内成员组成的会话。

如下图所示，一个会话表示与一个好友的对话：

下图为群聊天会话：



消息：ImSDK 中消息(webim.Msg)表示要发送给对方的内容，消息包括若干属性，如自己是否为发送者，发送人帐号，消息产生时间等；一条消息由若干 Elem 组合而成，每种 Elem 可以是文本、表情，图片等，消息支持多种 Elem 组合发送。

1.2.1. ImSDK 对象简介

ImSDK 对象主要分为消息，会话，消息存储对象，自带表情对象，工具对象，具体的含义参见下表：

对象	介绍	功能
webim.Msg	一条消息的描述类,	消息发送、接收的 API 中都会涉及此类型的对象
webim.Session	一个会话的描述类	包括获取会话类型，会话 ID，会话中的未读消息数，会话中的总消息数等功能
webim.MsgStore	消息数据的 Model 对象(参考 MVC 概念)	提供接口访问当前存储的会话和消息数据，包括获取当前会话的个数，根据会话类型和会话 ID 取得相应会话等功能
webim.EmotionPicData	表情数据	表情数据，格式是 base64 编码的
webim.EmotionPicDataIndex	表情数据索引	表情数据对应的索引
webim.Tool	工具对象	提供了一些公用的函数。比如格式化时间戳函数 formatTimeStamp(),获取字符串（utf-8 编码）所占字节数 getStrBytes()等等。

1.2.2. 消息对象 Msg

webim.Msg:一条消息的描述类, 消息发送、接收的 API 中都会涉及此类型的对象

/* class webim.Msg
* 一条消息的描述类, 消息发送、接收的 API 中都会涉及此类型的对象
* properties:
* sess - Session object-ref, 消息所属的会话(e.g:我与好友 A 的 C2C 会话，我与群组 G 的 GROUP 会话)
* isSend - Boolean, true 表示是我发出消息, false 表示是发给我的消息)
* seq - Integer, 消息序列号, 用于判断消息是否同一条
* random - Integer, 消息随机数,用于判断消息是否同一条
* time - Integer, 消息时间戳, 为 unix timestamp
* fromAccount -String, 消息发送者帐号
* elems - Array of webim.Msg.Elem, 描述消息内容的元素列表
* constructor:
* Msg(sess, isSend, seq,random time,fromAccount) - 构造函数, 参数定义同上面 properties 中定义
* methods:
* addText(text) - 向 elems 中添加一个 TEXT 元素
* addFace(face) - 向 elems 中添加一个 FACE 元素
* toHtml() - 转成可展示的 html String

```
*/
```

webim.Msg.Elem:消息中一个组成元素的描述类，一条消息的内容被抽象描述为 N 个元素的有序列表

```
/* sub-class webim.Msg.Elem
    *    消息中一个组成元素的描述类，一条消息的内容被抽象描述为 N 个元素的有序列表
    * properties:
    *    type      - 元素类型，目前有 TEXT(文本)、FACE(表情)等
    *    content-  元素内容体，当 TEXT 时为 String
    * constructor:
    *    Elem(type, content) -  构造函数，参数定义同上面 properties 中定义
    *
    * sub-class webim.Msg.Elem.Text
    *    文本
    * properties:
    *    text    - String  内容
    * constructor:
    *    Text(text) -  构造函数，参数定义同上面 properties 中定义
         *
    * sub-class webim.Msg.Elem.Face
    *    表情
    * properties:
    *    index    - Integer  表情索引，用户自定义
    *    data     - String  额外数据，用户自定义
    * constructor:
    *    Face(index,data) -  构造函数，参数定义同上面 properties 中定义
         *
    */
```

1.2.3. 会话对象 Session

一个 Session 对象描述一个会话，会话可简单理解为最近会话列表的一个条目

```
 /* class Session
        *        type   - String,  会话类型(如"C2C", "GROUP", ...)
        *        id     -  String,  会话  ID(如  C2C  类型中为对方帐号,"C2C"时为好友 ID,"GROUP"时为群 ID)
        * properties:
        *    (Session 对象未对外暴露任何属性字段，所有访问通过下面的 getter 方法进行)
        * methods:
```

```
     *    type()        -  String,  返回会话类型,"C2C"表示与好友私聊，"GROUP"
表示群聊
     *    id()      - String,  返回会话 ID
     *    name()        - String,  返回会话标题(如 C2C 类型中为对方的昵称)
     *    icon()        - String,  返回会话图标(对 C2C 类型中为对方的头像 URL)
     *    unread()              - Integer,  返回会话未读条数
     *    time()        - Integer,  返回会话最后活跃时间,  为 unix timestamp
     *    curMaxMsgSeq()  - Integer,  返回会话最大消息序列号
     *    msgCount()   - Integer,  返回会话中所有消息条数
     *    msg(index)   - webim.Msg,  返回会话中第 index 条消息
     */
```

1.2.4. 消息存储对象 MsgStore

webim.MsgStore 是消息数据的 Model 对象(参考 MVC 概念), 它提供接口访问当前存储的会话和消息数据

```
MsgStore: {
        /* function sessMap
         *    获取所有会话
         * return:
         *    所有会话对象
         */
        sessMap: function() {return {/*Object*/};},
        /* function sessCount
         *    获取当前会话的个数
         * return:
         *    Integer,  会话个数
         */
        sessCount: function() {return 0;},
        /* function sessByTypeId
         *    根据会话类型和会话 ID 取得相应会话
         * params:
         *    type      - String,  会话类型(如"C2C", "GROUP", ...)
         *    id        - String,  会话 ID(如对方 ID)
         * return:
         *    Session,  会话对象(说明见上面)
         */
        sessByTypeId: function(type, id) {return {/*Session Object*/};},
        /* function delSessByTypeId
         *    根据会话类型和会话 ID 删除相应会话
         * params:
         *    type      - String,  会话类型(如"C2C", "GROUP", ...)
```

```
     *     id          - String,  会话 ID(如对方 ID)
     * return:
     *     Boolean,  布尔类型
     */
    delSessByTypeId: function(type, id) {return true;},

    /* function resetCookieAndSyncFlag
     *     重置上一次读取新 c2c 消息 Cookie 和是否继续拉取标记
     * return:
     *
     */
    resetCookieAndSyncFlag: function() {}
}
```

1.2.5. 自带表情对象 EmotionPicData

webim.EmotionPicData 是表情数据，所有的表情是经过 base64 编码的。

```
var emotionPicData = {
        "[微笑]":"
        "[撇嘴]":"
        "[色]":"
        "[发呆]":"
        "[得意]":"
        "[流泪]":"
        "[害羞]":"
        "[闭嘴]":"-
        "[睡]":"
        "[大哭]":"
        "[尴尬]":"
        "[发怒]":"
        "[调皮]":"-
        "[呲牙]":"
        "[惊讶]":"
```

webim.EmotionPicDataIndex 为表情数据索引。

```
var emotionPicDataIndex = {
        "[微笑]":1,
        "[撇嘴]":2,
        "[色]":3,
        "[发呆]":4,
        "[得意]":5,
        "[流泪]":6,
        "[害羞]":7,
        "[闭嘴]":8,
        "[睡]":9,
        "[大哭]":10,
        "[尴尬]":11,
        "[发怒]":12,
        "[调皮]":13,
        "[呲牙]":14,
        "[惊讶]":15,
```

1.2.6. 工具对象 Tool

webim.Tool 提供了一些公用的函数。比如格式化时间戳函数 formatTimeStamp(),获取字符串所占字节数 getStrBytes()。

1.2.7. 调用顺序介绍

当帐号为**独立模式**时，请忽略【TLS 登录】这个步骤，需要开发者在自己的服务器上调用 TLS API 生成用户票据，然后进行初始化等后续步骤，帐号为**托管模式**时，ImSDK 调用 API 需要遵循以下顺序流程：

步骤	对应函数	说明
TLS 登录（托管模式下才有）	TLSHelper.getQuery('tmpsig')	获取浏览器 URL 地址的参数为 tmpsig 的值
	TLSHelper.goLogin({ sdkappid: loginInfo.sdkAppID, acctype: loginInfo.accountType, url: callBackUrl });	TLS 登录函数，需要传入 appid、accountType 和回调地址，登录成功后会跳转到 callBackUrl
	TLSHelper.fetchUserSig();	获取用户身份凭证，成功之后会回调 tlsGetUserSig(res)函数
初始化	webim.init	初始化 SDK，需要传入当前用

		户信息，新消息通知回调函数
消息收发	webim.syncMsgs	获取未读 c2c 消息
	webim.setAutoRead	设置会话自动已读标记
	webim.sendMsg	发送消息(私聊和群聊)
	webim.syncGroupMsgs	获取群历史消息
资料管理	webim.getProfilePortrait webim.setProfilePortrait	获取/设置个人资料
关系链管理	webim.applyAddFriend webim.getAllFriend 等	申请添加好友， 获取我的好友等
群组管理	webim. createGroup webim. applyJoinGroup 等	创建群，申请加群等
登出	webim.offline	用户登出

2. 登录

2.1. TLS 登录（托管模式）

Demo 集成了**托管模式**下的腾讯登录服务（Tencent Login Service，TLS），当帐号为**独立模式**时，请跳过这一小节，关于 TLS 账号集成（托管模式和独立模式）更多详细介绍，请参考链接：**http://www.qcloud.com/wiki/音视频云通信帐号登录集成**，这里只介绍在 demo 中如何集成**托管模式**下的 web 版 TLS SDK。

（1）在 index.html 引入 web 版 TLS sdk，如：

```
<script type="text/javascript" src="https://tls.qcloud.com/libs/api.min.js"></script>
```

（2）然后在页面中调用 TLSHelper.getQuery('tmpsig')，判断是否获取到了临时身份凭证，没有，则调用 TLSHelper.goLogin({sdkappid: loginInfo.sdkAppID,acctype: loginInfo.accountType,url: callBackUrl})，跳转到 tls 登录页面，登录成功会跳转到回调地址 callBackUrl；

示例：

```
//判断是否已经拿到临时身份凭证
if (TLSHelper.getQuery('tmpsig')) {
    if (loginInfo.identifier == null) {
        console.info('start fetchUserSig');
        //获取正式身份凭证，成功后会回调 tlsGetUserSig(res)函数
        TLSHelper.fetchUserSig();
    }
} else {//未登录
    if (loginInfo.identifier == null) {
        //弹出选择应用类型对话框
        $('#select_app_dialog').modal('show');
        $("body").css("background-color", 'white');
    }
}
```

```
//tls 登录
function tlsLogin() {
    //跳转到 TLS 登录页面
    TLSHelper.goLogin({
        sdkappid: loginInfo.sdkAppID,
        acctype: loginInfo.accountType,
        url: callBackUrl
    });
}
```

（3）如果已经拿到了临时凭证，则继续调用 TLSHelper.fetchUserSig()获取正式身份凭证，成功之后会回调 tlsGetUserSig(res)函数。

示例：

```
//第三方应用需要实现这个函数，并在这里拿到 UserSig
function tlsGetUserSig(res) {
    //成功拿到凭证
    if (res.ErrorCode == TlsErrorCode.OK) {
        //从当前 URL 中获取参数为 identifier 的值
        loginInfo.identifier = TLSHelper.getQuery("identifier");
        //拿到正式身份凭证
        loginInfo.userSig = res.UserSig;
        //从当前 URL 中获取参数为 sdkappid 的值
        loginInfo.sdkAppID = loginInfo.appIDAt3rd = Number(TLSHelper.getQuery("sdkappid"));
        //从 cookie 获取 accountType
        var accountType = webim.Tool.getCookie('accountType');
        if (accountType) {
            loginInfo.accountType = accountType;
            initDemoApp();
        } else {
            alert('accountType 非法');
        }
    } else {
        //签名过期，需要重新登录
        if (res.ErrorCode == TlsErrorCode.SIGNATURE_EXPIRATION) {
            tlsLogin();
        } else {
            alert("[" + res.ErrorCode + "]" + res.ErrorInfo);
        }
    }
}
```

2.2. 登出

如用户主动注销或需要进行用户的切换，则需要调用注销操作：

```
/* function offline
 *    用户下线
 * params:
 *    cbOk    - function()类型，成功时回调函数
 *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
 * return:
 *    (无)
 */
offline: function(cbOk, cbErr) {},
```

示例：

```
//退出
function quitClick() {
    if (loginInfo.identifier) {
        document.getElementById("webim_demo").style.display = "none";
        //离线
        webim.offline(
                function (resp) {
                    loginInfo.identifier = null;
                    loginInfo.userSig = null;
                    window.location.href = callBackUrl;
                }
        );
    } else {
        alert('未登录');
    }
}
```

3. 初始化

3.1. 初始化 init

webim.init 表示初始化 web sdk,需要传入当前用户信息，新消息通知回调函数。

```
/* function init
 *    初始化 SDK
 * params:
 *    loginInfo            - Object，登录身份相关参数集合，详见下面
 *    {
```

```
        *       sdkAppID        - String, 用户标识接入 SDK 的应用 ID
        *       appIDAt3rd      - String, App 用户使用 OAuth 授权体系分配的 Appid，和 sdkAppID 一样
        *       accountType     - int, 账号类型
        *       identifier      - String, 用户帐号
        *       userSig         - String, 鉴权 Token
        *   }
        *   listeners       - Object, 事件回调函数集合, 详见下面
        *   {
        *       onConnNotify - function(connInfo), 用于收到连接状态相关通知的回调函数, 目前未使用
        *       jsonpCallback -function(rspData),//IE9(含)以下浏览器用到的 jsonp 回调函数
        *       onMsgNotify     - function(notifyInfo), 用于收到消息通知的回调函数, notifyInfo 为[{msg: Msg 对象}]
        *                         使用方有两种处理回调: 1)直接访问 webim.MsgStore 获取最新的消息 2)处理 notifyInfo 中的增量消息
        *       onGroupInfoChangeNotify     - function(notifyInfo), 用于监听群组资料变更的回调函数, notifyInfo 为新的群组资料信息
        *       groupSystemNotifys - Object, 用于监听（多终端同步）群系统消息的回调函数对象
        *
        *   }
        *   options         - Object, 其它选项, 目前未使用
        * return:
        *   (无)
        */
       init: function(loginInfo, listeners, options) {},
```

Demo 中使用例子：

```
       //当前用户身份
var loginInfo = {
    sdkAppID: sdkAppID, //用户所属应用 id
    appIDAt3rd: sdkAppID, //用户所属应用 id
    accountType: accountType, //用户所属应用帐号类型
    identifier: null, //当前用户 ID
    userSig: null, //当前用户身份凭证
    headurl: 'img/2016.gif'//当前用户默认头像
};

//监听新消息事件
function onMsgNotify(newMsg) {
    //获取所有聊天会话
    var sessMap = webim.MsgStore.sessMap();
    for (var i in sessMap) {
```

```
            var sess = sessMap[i];
            if (selToID == sess.id()) {//处于当前聊天界面
                selSess = sess;
                //获取当前会话消息数
                var msgCount = sess.msgCount();
                // add new msgs
                if (msgCount > curMsgCount) {

                    for (var j = curMsgCount; j < msgCount; j++) {
                        var msg = sess.msg(j);
                        //在聊天窗体中新增一条消息
                        addMsg(msg);
                        curMsgCount++;
                    }
                    //消息已读上报，以及设置会话自动已读标记
                    webim.setAutoRead(selSess, true, true);
                }
            } else {
                //更新其他聊天对象的未读消息数
                updateSessDiv(sess.id(), sess.unread());
            }

        }
    }
    //
    //监听（多终端同步）群系统消息方法，方法都定义在 webim_demo_group_notice.js 文件中
    //注意每个数字代表的含义，比如，
    //1 表示监听申请加群消息，2 表示监听申请加群被同意消息，3 表示监听申请加群被拒绝消息
    var groupSystemNotifys = {
        "1": onApplyJoinGroupRequestNotify, //申请加群请求（只有管理员会收到）
        "2": onApplyJoinGroupAcceptNotify, //申请加群被同意（只有申请人能够收到）
        "3": onApplyJoinGroupRefuseNotify, //申请加群被拒绝（只有申请人能够收到）
        "4": onKickedGroupNotify, //被管理员踢出群(只有被踢者接收到)
        "5": onDestoryGroupNotify, //群被解散(全员接收)
        "6": onCreateGroupNotify, //创建群(创建者接收)
        "7": onInvitedJoinGroupNotify, //邀请加群(被邀请者接收)
        "8": onQuitGroupNotify, //主动退群(主动退出者接收)
        "9": onSetedGroupAdminNotify, //设置管理员(被设置者接收)
        "10": onCanceledGroupAdminNotify, //取消管理员(被取消者接收)
        "11": onRevokeGroupNotify, //群已被回收(全员接收)
        "255": onCustomGroupNotify//用户自定义通知(默认全员接收)
    };
```

```
//IE9(含)以下浏览器用到的 jsonp 回调函数
function jsonpCallback(rspData) {
    webim.setJsonpLastRspData(rspData);
}

//监听事件
var listeners = {
    "onConnNotify": null,
    "jsonpCallback": jsonpCallback, //IE9(含)以下浏览器用到的 jsonp 回调函数
    "onMsgNotify": onMsgNotify, //监听新消息(私聊，群聊，群提示消息)事件
    "onGroupInfoChangeNotify": onGroupInfoChangeNotify, //监听群资料变化事件
    "groupSystemNotifys": groupSystemNotifys//监听（多终端同步）群系统消息事件
};

//初始化 demo
function initDemoApp() {
    $("body").css("background-color", '#2f2f2f');
    document.getElementById("webim_demo").style.display = "block";//展开聊天界面
    document.getElementById("p_my_face").src = loginInfo.headurl;
    document.getElementById("t_my_name").innerHTML = loginInfo.identifier;
    //菜单
    $("#t_my_menu").menu();

    //web sdk  初始化
    webim.init(loginInfo, listeners, null);
    //读取我的好友列表
    getAllFriend(getAllFriendsCallbackOK);
    //读取我的群组列表
    getJoinedGroupListHigh();
    $("#send_msg_text").focus();
    //初始化我的加群申请表格
    initGetApplyJoinGroupPendency([]);
    //初始化我的群组系统消息表格
    initGetMyGroupSystemMsgs([]);
}
```

4. 消息收发

4.1. 发送消息（文本和表情）

```
    /* function sendMsg
```

```
     *     发送一条消息
     * params:
     *     msg      - webim.Msg 类型，要发送的消息对象
     *     cbOk     - function()类型，当发送消息成功时的回调函数
     *     cbErr    - function(err)类型，当发送消息失败时的回调函数, err 为错误对
象
     * return:
     *     (无)
     */
    sendMsg: function(msg, cbOk, cbErr) {},
```

示例：

```
    //发送消息
    function onSendMsg() {
        if (!selToID) {
            alert("您还没有好友，暂不能聊天");
            $("#send_msg_text").val('');
            return;
        }
        //获取消息内容
        var msgtosend = document.getElementsByClassName("msgedit")[0].value;
        var msgLen = webim.Tool.getStrBytes(msgtosend);

        if (msgtosend.length < 1) {
            alert("发送的消息不能为空!");
            $("#send_msg_text").val('');
            return;
        }
        var maxLen,errInfo;
        if(selType==SessionType.C2C){
            maxLen=MaxMsgLen.C2C;
            errInfo="消息长度超出限制(最多"+Math.round(maxLen/3)+"汉字)";
        }else{
            maxLen=MaxMsgLen.GROUP;
            errInfo="消息长度超出限制(最多"+Math.round(maxLen/3)+"汉字)";
        }
        if (msgLen > maxLen) {
            alert(errInfo);
            return;
        }
        if (!selSess) {
            selSess = new webim.Session(selType, selToID, selToID, friendHeadUrl,
Math.round(new Date().getTime() / 1000));
```

```
                }
                var msg = new webim.Msg(selSess, true);

                //解析文本和表情
                var expr = /\[[^[\]]{1,3}\]/mg;
                var emotions = msgtosend.match(expr);
                if (!emotions || emotions.length < 1) {
                    var text_obj = new webim.Msg.Elem.Text(msgtosend);
                    msg.addText(text_obj);
                } else {

                    for (var i = 0; i < emotions.length; i++) {
                        var tmsg = msgtosend.substring(0, msgtosend.indexOf(emotions[i]));
                        if (tmsg) {
                            var text_obj = new webim.Msg.Elem.Text(tmsg);
                            msg.addText(text_obj);
                        }
                        var emotion = webim.EmotionPicData[emotions[i]];
                        if (emotion) {

                            var face_obj = new webim.Msg.Elem.Face(webim.EmotionPicDataIndex[emotions[i]], emotions[i]);
                            msg.addFace(face_obj);

                        } else {
                            var text_obj = new webim.Msg.Elem.Text(emotions[i]);
                            msg.addText(text_obj);
                        }
                        var restMsgIndex = msgtosend.indexOf(emotions[i]) + emotions[i].length;
                        msgtosend = msgtosend.substring(restMsgIndex);
                    }
                    if (msgtosend) {
                        var text_obj = new webim.Msg.Elem.Text(msgtosend);
                        msg.addText(text_obj);
                    }
                }

                webim.sendMsg(msg, function (resp) {
                    addMsg(msg);
                    curMsgCount++;
                    $("#send_msg_text").val('');
                    turnoffFaces_box();
```

```
        }, function (err) {
            alert(err.ErrorInfo);
            $("#send_msg_text").val('');
        });
    }
```

4.2. 上传图片

目前 demo 采用了 H5 FileAPI 读取图片，并将图片二进制数据转换成 base64 编码进行分片上传，所以 ie9（含）以下暂不支持上传图片。

```
/* function uploadPic
 *     上传图片
 * params:
 *     cbOk     - function()类型，成功时回调函数
 *     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
 * return:
 *     (无)
 */
uploadPic: function(options,cbOk, cbErr) {},
```

示例：

```
//上传图片
function uploadPic() {
    var uploadFiles = document.getElementById('upd_pic');
    var file = uploadFiles.files[0];

    var businessType;//业务类型，1-发群图片，2-向好友发图片
    if (selType == SessionType.C2C) {//向好友发图片
        businessType = UploadPicBussinessType.C2C_MSG;
    } else if (selType == SessionType.GROUP) {//发群图片
        businessType = UploadPicBussinessType.GROUP_MSG;
    }
    //封装上传图片请求
    var opt = {
        'file': file, //图片对象
        'onProgressCallBack': onProgressCallBack, //上传图片进度条回调函数
        //'abortButton': document.getElementById('upd_abort'), //停止上传图片按钮
        'From_Account': loginInfo.identifier, //发送者帐号
        'To_Account': selToID, //接收者
        'businessType': businessType//业务类型
    };
    //上传图片
    webim.uploadPic(opt,
```

```
                function (resp) {
                    //上传成功发送图片
                    sendPic(resp);
                    $('#upload_pic_dialog').modal('hide');
                },
                function (err) {
                    alert(err.ErrorInfo);
                }
            );
        }
```

4.3. 发送消息（图片）

在 IE9（含）以下浏览器，sdk 采用了 jsonp 方法解决 ajax 跨域问题，由于 jsonp 是采用 get 方法传递数据的，且 get 存在数据大小限制（不同浏览器不一样），所以暂不支持异步发送图片。

```
/* function sendMsg
     *    发送一条消息
     * params:
     *    msg      - webim.Msg 类型, 要发送的消息对象
     *    cbOk     - function()类型, 当发送消息成功时的回调函数
     *    cbErr    - function(err)类型, 当发送消息失败时的回调函数, err 为错误对象
     * return:
     *    (无)
     */
    sendMsg: function(msg, cbOk, cbErr) {},
```

示例：

```
//发送图片
function sendPic(images) {
    if (!selToID) {
        alert("您还没有好友，暂不能聊天");
        return;
    }

    if (!selSess) {
        selSess = new webim.Session(selType, selToID, selToID, friendHeadUrl, Math.round(new Date().getTime() / 1000));
    }
    var msg = new webim.Msg(selSess, true);
```

```
    var images_obj = new webim.Msg.Elem.Images(images.File_UUID);
    for (var i in images.URL_INFO) {
        var img = images.URL_INFO[i];
        var newImg;
        var type;
        switch (img.PIC_TYPE) {
            case 1://原图
                type = 1;//原图
                break;
            case 2://小图（缩略图）
                type = 3;//小图
                break;
            case 4://大图
                type = 2;//大图
                break;
        }
        newImg = new webim.Msg.Elem.Images.Image(type, img.PIC_Size, img.PIC_Width,
img.PIC_Height, img.DownUrl);
        images_obj.addImage(newImg);
    }
    msg.addImage(images_obj);
    //调用发送图片接口
    webim.sendMsg(msg, function (resp) {
        addMsg(msg);
        curMsgCount++;
    }, function (err) {
        alert(err.ErrorInfo);
    });
}
```

4.4. 播放语音

目前 web 端只支持显示并播放 android 或 ios im demo 发的语音消息，暂不支持上传并发送语音消息。使用 audio 控件来播放语音，注意，确保其他终端上传的语音格式是 mp3 格式（所有主流浏览器下的 audio 控件都兼容 mp3，除了 IE8 下不支持使用 audio 标签播放语音）。

语音消息对象如下：

```
// class Msg.Elem.Sound
Msg.Elem.Sound = function (uuid, second, size, senderId, downUrl) {
    this.uuid = uuid;//语音 id
    this.second = second;//时长，单位：秒
    this.size = size;//大小，单位：字节
    this.senderId = senderId;//发送者 id
    this.downUrl = downUrl;//下载 url
};
```

```
Msg.Elem.Sound.prototype.getUUID = function () {
    return this.uuid;
};
Msg.Elem.Sound.prototype.getSecond = function () {
    return this.second;
};
Msg.Elem.Sound.prototype.getSize = function () {
    return this.size;
};
Msg.Elem.Sound.prototype.getSenderId = function () {
    return this.senderId;
};
Msg.Elem.Sound.prototype.getDownUrl = function () {
    return this.downUrl;
};
Msg.Elem.Sound.prototype.toHtml = function () {
    if (browserInfo.type == 'ie' && parseInt(browserInfo.ver) <= 8) {
        return '[这是一条语音消息]demo 暂不支持 ie8(含)以下浏览器播放语音,语音 URL:' + this.downUrl;
    }
    return '<audio src="' + this.downUrl + '" controls="controls" onplay="onChangePlayAudio(this)" preload="none"></audio>';
};
```

4.5. 下载文件

目前 web 端只支持显示并下载 android 或 ios im demo 发的文件消息，暂不支持上传并发送文件消息。

文件消息对象如下：

```
// class Msg.Elem.File
Msg.Elem.File = function(uuid,name,size,senderId,downUrl) {
    this.uuid = uuid;//文件 id
    this.name = name;//文件名
    this.size = size;//大小，单位：字节
    this.senderId = senderId;//发送者
        this.downUrl = downUrl;//下载地址
};
    Msg.Elem.File.prototype.getUUID = function() {
        return this.uuid;
};
    Msg.Elem.File.prototype.getName = function() {
        return this.name;
};
    Msg.Elem.File.prototype.getSize = function() {
```

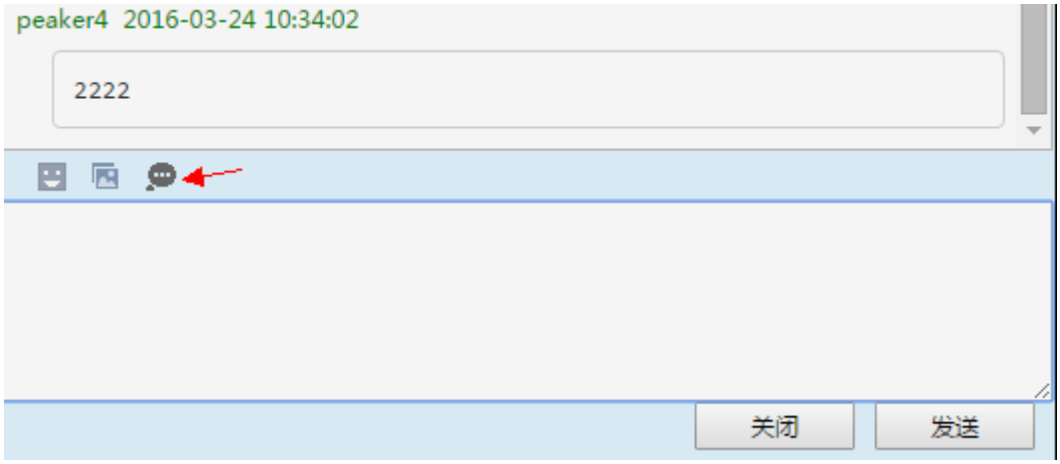
```
                return this.size;
        };
                Msg.Elem.File.prototype.getSenderId = function() {
                        return this.senderId;
        };
                Msg.Elem.File.prototype.getDownUrl = function() {
                        return this.downUrl;
        };
                Msg.Elem.File.prototype.toHtml = function() {
                        var fileSize=Math.round(this.size/1024);
                        return '<a href="'+this.downUrl+'" title="点击下载文件" ><i
    class="glyphicon glyphicon-file"> '+this.name+'('+fileSize+'KB)</i></a>';
        };
```

4.6. 发送消息（自定义）

```
    /* function sendMsg
        *     发送一条消息
        * params:
        *     msg      - webim.Msg 类型，要发送的消息对象
        *     cbOk     - function()类型，当发送消息成功时的回调函数
        *     cbErr    - function(err)类型，当发送消息失败时的回调函数, err 为错误对
    象
        * return:
        *     (无)
        */
        sendMsg: function(msg, cbOk, cbErr) {},
```

示例：

发送自定义消息入口




```
//发送自定义消息
```

```
function sendCustomMsg() {
    if (!selToID) {
        alert("您还没有好友或群组，暂不能聊天");
        return;
    }
    var data = $("#ecm_data").val();
    var desc = $("#ecm_desc").val();
    var ext = $("#ecm_ext").val();

    var msgLen = webim.Tool.getStrBytes(data);

    if (data.length < 1) {
        alert("发送的消息不能为空!");
        return;
    }
    var maxLen, errInfo;
    if (selType == SessionType.C2C) {
        maxLen = MaxMsgLen.C2C;
        errInfo = "消息长度超出限制(最多" + Math.round(maxLen / 3) + "汉字)";
    } else {
        maxLen = MaxMsgLen.GROUP;
        errInfo = "消息长度超出限制(最多" + Math.round(maxLen / 3) + "汉字)";
    }
    if (msgLen > maxLen) {
        alert(errInfo);
        return;
    }

    if (!selSess) {
        selSess = new webim.Session(selType, selToID, selToID, friendHeadUrl, Math.round(new
Date().getTime() / 1000));
    }
    var msg = new webim.Msg(selSess, true);
    var custom_obj = new webim.Msg.Elem.Custom(data, desc, ext);
    msg.addCustom(custom_obj);
    //调用发送消息接口
    webim.sendMsg(msg, function (resp) {
        addMsg(msg);
        curMsgCount++;
        $('#edit_custom_msg_dialog').modal('hide');
    }, function (err) {
        alert(err.ErrorInfo);
    });
}
```

4.7. 获取未读 c2c 消息

```
/* function syncMsgs
        *   拉取最新 C2C 消息
        *   一般不需要使用方直接调用，SDK 底层会自动同步最新消息并通知使用方，一种有用的调用场景是用户手动触发刷新消息
        * params:
        *   cbOk   - function(notifyInfo)类型，当同步消息成功时的回调函数，notifyInfo 同上面 cbNotify 中的说明，
        *                 如果此参数为 null 或 undefined 则同步消息成功后会像自动同步那样回调 cbNotify
        *   cbErr   - function(err)类型，当同步消息失败时的回调函数，err 为错误对象
        * return:
        *   (无)
        */
       syncMsgs: function(cbOk, cbErr) {},
```

示例:

```
webim.syncMsgs(syncMsgsCallbackOK);
//获取 C2C 最新消息成功回调函数
        function syncMsgsCallbackOK() {

            if (webim.MsgStore.sessCount() > 0) {

                var sessMap = webim.MsgStore.sessMap();
                for (var i in sessMap) {
                    //console.info("sessMap[i]=%O",sessMap[i]);
                    var sess = sessMap[i];
                    if (selToID == sess.id()) {//处于当前聊天界面
                        selSess = sess;

                        var msgCount = sess.msgCount();
                        // add new msgs
                        if (msgCount > curMsgCount) {

                            for (var mj = curMsgCount; mj < msgCount; mj++) {
                                var msg = sess.msg(mj);
                                addMsg(msg);
                                curMsgCount++;
                            }
```

```
                        //消息已读上报，以及设置会话自动已读标记
                        webim.setAutoRead(selSess, true, true);
                    }
                } else {
                    //更新其他聊天对象的未读消息数
                    updateSessDiv(sess.id(), sess.unread());
                }

            }
        }
    }
```

4.8. 获取群漫游消息

```
    /* function syncGroupMsgs
     *  拉取群漫游消息
     * params:
     *     options  -  请求参数
     *     cbOk     - function()类型，成功时回调函数
     *     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *     (无)
     */
    syncGroupMsgs: function(options,cbOk, cbErr) {},
```

示例:

```
    //获取我的群组列表回调函数
    function getGroupsCallbackOK() {

        selType == SessionType.GROUP && getGroupInfo(selToID, function (resp) {
            //拉取最新消息
            var opts = {
                'GroupId': selToID,
                'ReqMsgSeq': resp.GroupInfo[0].NextMsgSeq - 1,
                'ReqMsgNumber': 100
            };
            if (opts.ReqMsgSeq == null || opts.ReqMsgSeq == undefined) {
                alert('群消息序列号非法');
                return;
            }
            webim.syncGroupMsgs(opts, syncGroupMsgsCallbackOK);
        });
    }
```

4.9. 获取所有会话

Session 对象,简单理解为最近会话列表的一个条目

```
/* class Session
*       type  - String, 会话类型(如"C2C", "GROUP", ...)
*       id    - String, 会话 ID(如 C2C 类型中为对方帐号,"C2C"时为好友 ID,"GROUP"时为群 ID)
* properties:
*    (Session 对象未对外暴露任何属性字段, 所有访问通过下面的 getter 方法进行)
* methods:
*    type()        - String, 返回会话类型,"C2C"表示与好友私聊，"GROUP"表示群聊
*    id()      - String, 返回会话 ID
*    name()        - String, 返回会话标题(如 C2C 类型中为对方的昵称)
*    icon()        - String, 返回会话图标(对 C2C 类型中为对方的头像 URL)
*    unread()              - Integer, 返回会话未读条数
*    time()        - Integer, 返回会话最后活跃时间, 为 unix timestamp
*    curMaxMsgSeq()  - Integer, 返回会话最大消息序列号
*    msgCount()  - Integer, 返回会话中所有消息条数
*    msg(index)  - webim.Msg, 返回会话中第 index 条消息
*/
```

webim.MsgStore 是消息数据的 Model 对象,它提供接口访问当前存储的会话和消息数据。

```
MsgStore: {
        /* function sessMap
         *    获取所有会话
         * return:
         *    所有会话对象
         */
        sessMap: function() {return {/*Object*/};},
        /* function sessCount
         *    获取当前会话的个数
         * return:
         *    Integer, 会话个数
         */
        sessCount: function() {return 0;},
```

```
        /* function sessByTypeId
         *     根据会话类型和会话 ID 取得相应会话
         * params:
         *     type      - String,  会话类型(如"C2C", "GROUP", ...)
         *     id        - String,  会话 ID(如对方 ID)
         * return:
         *     Session,  会话对象(说明见上面)
         */
        sessByTypeId: function(type, id) {return {/*Session Object*/};},
        /* function delSessByTypeId
         *     根据会话类型和会话 ID 删除相应会话
         * params:
         *     type      - String,  会话类型(如"C2C", "GROUP", ...)
         *     id        - String,  会话 ID(如对方 ID)
         * return:
         *     Boolean,  布尔类型
         */
        delSessByTypeId: function(type, id) {return true;},

        /* function resetCookieAndSyncFlag
         *     重置上一次读取新 c2c 消息 Cookie 和是否继续拉取标记
         * return:
         *
         */
        resetCookieAndSyncFlag: function() {}
    }
```

示例：

```
//获取所有聊天会话
var sessMap = webim.MsgStore.sessMap();

//遍历会话列表
for (var i in sessMap) {
    var sess = sessMap[i];
    if (selToID == sess.id()) {//处于当前聊天界面
        selSess = sess;
        //获取当前会话消息数
        var msgCount = sess.msgCount();
        // add new msgs
        if (msgCount > curMsgCount) {
```

```
                for (var j = curMsgCount; j < msgCount; j++) {
                    var msg = sess.msg(j);
                    //在聊天窗体中新增一条消息
                    addMsg(msg);
                    curMsgCount++;
                }
                //消息已读上报，以及设置会话自动已读标记
                webim.setAutoRead(selSess, true, true);
                }
            } else {
                //更新其他聊天对象的未读消息数
                updateSessDiv(sess.id(), sess.unread());
            }
        }
```

4.10. 获取会话

可以根据会话类型和会话 ID 取得相应会话。

```
        /* function sessByTypeId
                 *     根据会话类型和会话 ID 取得相应会话
                 * params:
                 *     type      - String,  会话类型(如"C2C", "GROUP", ...)
                 *     id        - String,  会话 ID(如对方 ID)
                 * return:
                 *     Session,  会话对象(说明见上面)
                 */
                sessByTypeId: function(type, id) {return {/*Session Object*/};},
```

示例：

```
selSess = webim.MsgStore.sessByTypeId(selType, selToID);
```

5. 未读计数

5.1. 获取当前会话未读消息数

可以根据 Session 对象定义的 unread()方法获取未读消息数

示例：

```
//更新其他聊天对象的未读消息数
updateSessDiv(sess.id(), sess.unread());
```

5.2. 设置会话自动已读标记

当用户阅读某个会话的数据后，需要进行会话消息的已读上报，SDK 根据会话中最后一条阅读的消息，设置会话中之前所有消息为已读。

```
/* function setAutoRead
        *     //设置会设置会话自动已读标记话自动已读上报标志
        * params:
        *     selSess   - webim.Session 类型, 当前会话
        *     isOn      - boolean, 将 selSess 的自动已读消息标志改为 isOn，同时是否上报当前会话已读消息
        *     isResetAll      - boolean，是否重置所有会话的自动已读标志
        * return:
        *     (无)
        */
       setAutoRead: function(selSess, isOn, isResetAll) {},
```

示例：

```
//消息已读上报，以及设置会话自动已读标记
webim.setAutoRead(selSess, true, true);
```

5.3. 重置上一次读取新 c2c 消息 Cookie 和是否继续拉取标记

当切换聊天对象时，需要调用 resetCookieAndSyncFlag 方法。

```
/* function resetCookieAndSyncFlag
        *     重置上一次读取新 c2c 消息 Cookie 和是否继续拉取标记
        * return:
        *
        */
       resetCookieAndSyncFlag: function() {}
```

示例：

```
webim.MsgStore.resetCookieAndSyncFlag();
```

6. 用户资料

IM 云提供的帐号分为两种，一种是独立帐号体系，由用户自己保存并更新用户资料和关系链信息，另一种是托管模式，此模式下用户可完全不用搭建自己的后台服务，把用户的资料和关系链托管在 IM 云服务。

如果开发者选择资料和关系链托管，需要调用本章的接口进行资料和关系链的操作：

6.1. 设置个人资料

```
/* function setProfilePortrait
```

```
*     设置个人资料
* params:
*     cbOk     - function()类型，成功时回调函数
*     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
* return:
*     (无)
*/
setProfilePortrait: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-资料管理-REST 协议文档》

示例：
请参考 demo 中的 demo/js/webim_demo_profile.js 的示例代码

6.2. 获取个人资料

```
/* function getProfilePortrait
*     拉取资料（搜索用户）
* params:
*     cbOk     - function()类型，成功时回调函数
*     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
* return:
*     (无)
*/
getProfilePortrait: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-资料管理-REST 协议文档》

示例：
请参考 demo 中的 demo/js/webim_demo_profile.js 的示例代码。

7. 关系链

7.1. 申请增加好友

```
/* function applyAddFriend
*     申请添加好友
* params:
*     cbOk     - function()类型，成功时回调函数
*     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
* return:
*     (无)
*/
applyAddFriend: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》
示例：
请参考 demo 中的 demo/js/webim_demo_friend.js 的示例代码。

7.2. 拉取好友申请

```
/* function getPendency
    *    拉取好友申请
    * params:
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   getPendency: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_pendency.js 的示例代码。

7.3. 响应好友申请

```
/* function responseFriend
    *    响应好友申请
    * params:
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   responseFriend: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_pendency.js 的示例代码。

7.4. 删除好友申请

```
/* function deletePendency
    *    删除好友申请
    * params:
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   deletePendency: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_pendency.js 的示例代码。

7.5. 我的好友列表

```
/* function getAllFriend
    *    拉取我的好友
    * params:
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   getAllFriend: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_friend.js 的示例代码。

7.6. 删除好友

```
/* function deleteFriend
    *    删除好友
    * params:
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   deleteFriend: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_friend.js 的示例代码。

7.7. 增加黑名单

```
/* function addBlackList
 *    增加黑名单
 * params:
 *    cbOk    - function()类型，成功时回调函数
 *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
 * return:
 *    (无)
 */
addBlackList: function(options, cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_blacklist.js 的示例代码。

7.8. 我的黑名单

```
/* function getBlackList
    *    删除黑名单
    * params:
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   getBlackList: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_blacklist.js 的示例代码。

7.9. 删除黑名单

```
/* function deleteBlackList
    *    我的黑名单
    * params:
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   deleteBlackList: function(options,cbOk, cbErr) {},
```

具体参数请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

示例：

请参考 demo 中的 demo/js/webim_demo_blacklist.js 的示例代码。

8. 群组管理

8.1. 创建群

```
/* function createGroup
    *    创建群
    * params:
    *    options - 请求参数，详见 api 文档
    *    cbOk    - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
   createGroup: function(options,cbOk, cbErr) {},
```

示例：

```
//创建群组
var createGroup = function () {
    var sel_friends = $('#select_friends').val();
    if (!sel_friends) {
        alert('请先选择好友，再建群');
        return;
    }
    var member_list = [];
    var members = sel_friends.split(";"); //字符分割
    for (var i = 0; i < members.length; i++)
    {
        if (members[i] && members[i].length > 0) {
            member_list.push(members[i]);
        }
    }
    if (member_list.length <= 0) {
        alert('请先选择好友，再建群');
        return;
    }
    if ($("#cg_name").val().length == 0) {
        alert('请输入群组名称');
        return;
    }
    if (webim.Tool.trimStr($("#cg_name").val()).length == 0) {
        alert('您输入的群组名称全是空格,请重新输入');
        return;
    }
    if (webim.Tool.getStrBytes($("#cg_name").val()) > 30) {
        alert('您输入的群组名称超出限制(最长 10 个汉字)');
        return;
    }
    if (webim.Tool.getStrBytes($("#cg_notification").val()) > 150) {
        alert('您输入的群组公告超出限制(最长 50 个汉字)');
        return;
    }
    if (webim.Tool.getStrBytes($("#cg_introduction").val()) > 120) {
        alert('您输入的群组简介超出限制(最长 40 个汉字)');
        return;
    }
    var options = {
        'Owner_Account': loginInfo.identifier,
        'Type':                    $('input[name="cg_type_radio"]:checked').val(),
//Private/Public/ChatRoom
        'Name': $("#cg_name").val(),
```

```
            'FaceUrl': '',
            'Notification': $("#cg_notification").val(),
            'Introduction': $('#cg_introduction').val(),
            'MemberList': member_list
        };
        webim.createGroup(
                options,
                function (resp) {
                    $('#create_group_dialog').modal('hide');
                    alert('创建群成功');
                    //读取我的群组列表
                    getJoinedGroupListHigh(getGroupsCallbackOK);
                },
                function (err) {
                    alert(err.ErrorInfo);
                }
        );
};
```

8.2. 搜索群

```
/* function getGroupPublicInfo
     *    读取群公开资料-高级接口
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *    (无)
     */
    getGroupPublicInfo: function(options,cbOk, cbErr) {},
```

示例：

```
//读取群组公开资料-高级接口（可用于搜索群）
var getGroupPublicInfo = function () {
    if ($("#sg_group_id").val().length == 0) {
        alert('请输入群组 ID');
        return;
    }
    if (webim.Tool.trimStr($("#sg_group_id").val()).length == 0) {
        alert('您输入的群组 ID 全是空格,请重新输入');
        return;
    }
    var options = {
```

```
            'GroupIdList': [
                $("#sg_group_id").val()
            ],
            'GroupBasePublicInfoFilter': [
                'Type',
                'Name',
                'Introduction',
                'Notification',
                'FaceUrl',
                'CreateTime',
                'Owner_Account',
                'LastInfoTime',
                'LastMsgTime',
                'NextMsgSeq',
                'MemberNum',
                'MaxMemberNum',
                'ApplyJoinOption'
            ]
        };
        webim.getGroupPublicInfo(
                options,
                function (resp) {
                    var data = [];
                    if (resp.GroupInfo && resp.GroupInfo.length > 0) {
                        for (var i in resp.GroupInfo) {
                            if (resp.GroupInfo[i].ErrorCode > 0) {
                                alert(resp.GroupInfo[i].ErrorInfo);
                                return;
                            }
                            var group_id = resp.GroupInfo[i].GroupId;
                            var name = webim.Tool.formatText2Html(resp.GroupInfo[i].Name);
                            var type_zh = webim.Tool.groupTypeEn2Ch(resp.GroupInfo[i].Type);
                            var type = resp.GroupInfo[i].Type;
                            var owner_account = resp.GroupInfo[i].Owner_Account;
                            var create_time = webim.Tool.formatTimeStamp(resp.GroupInfo[i].CreateTime);
                            var member_num = resp.GroupInfo[i].MemberNum;
                            var notification = webim.Tool.formatText2Html(resp.GroupInfo[i].Notification);
                            var introduction = webim.Tool.formatText2Html(resp.GroupInfo[i].Introduction);
                            data.push({
```

```
                                    'GroupId': group_id,
                                    'Name': name,
                                    'TypeZh': type_zh,
                                    'Type': type,
                                    'Owner_Account': owner_account,
                                    'MemberNum': member_num,
                                    'Notification': notification,
                                    'Introduction': introduction,
                                    'CreateTime': create_time
                                });
                            }
                        }
                        $('#search_group_table').bootstrapTable('load', data);
                    },
                    function (err) {
                        alert(err.ErrorInfo);
                    }
            );
        };
```

8.3. 申请加群

```
        /* function applyJoinGroup
             *    申请加群
             * params:
             *    options - 请求参数，详见 api 文档
             *    cbOk    - function()类型，成功时回调函数
             *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
             * return:
             *    (无)
             */
            applyJoinGroup: function(options,cbOk, cbErr) {},
        示例：
        //申请加群
        var applyJoinGroup = function () {
            if (webim.Tool.getStrBytes($("#ajg_apply_msg").val()) > 300) {
                alert('您输入的附言超出限制(最长 100 个汉字)');
                return;
            }
            var options = {
                'GroupId': $("#ajg_group_id").val(),
                'ApplyMsg': $("#ajg_apply_msg").val(),
                'UserDefinedField': ''
            };
```

```
        webim.applyJoinGroup(
                options,
                function (resp) {
                    $('#apply_join_group_dialog').modal('hide');

                    if ($("#ajg_group_type").val() == 'ChatRoom') {
                        //刷新我的群组列表
                        getJoinedGroupListHigh(getGroupsCallbackOK);
                        alert('成功加入该聊天室');
                    } else {
                        alert('申请成功，请等待群主处理');
                    }
                },
                function (err) {
                    alert(err.ErrorInfo);
                }
        );
    };
```

8.4. 处理加群申请（同意或拒绝）

```
/* function handleApplyJoinGroup
 *    处理申请加群(同意或拒绝)
 * params:
 *    options - 请求参数，详见 api 文档
 *    cbOk    - function()类型，成功时回调函数
 *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
 * return:
 *    (无)
 */
handleApplyJoinGroup: function(options, cbOk, cbErr) {},
```

其中 options 定义如下：

```
'GroupId': //群 id
'Applicant_Account': //申请人 id
'HandleMsg': //是否同意,Agree-同意  Reject-拒绝
'Authentication': //申请凭证（包含在管理员收到的加群申请系统消息中）
'MsgKey': /消息 key（包含在管理员收到的加群申请系统消息中）
'ApprovalMsg': //处理附言
'UserDefinedField': //用户自定义字段（包含在管理员收到的加群申请系统消息中）
```

示例：

```
//处理加群申请
var handleApplyJoinGroup = function () {
```

```
        var options = {
            'GroupId': $("#hajg_group_id").val(), //群 id
            'Applicant_Account': $("#hajg_to_account").val(), //申请人 id
            'HandleMsg':  $('input[name="hajg_action_radio"]:checked').val(),  //是否同
意,Agree-同意  Reject-拒绝
            'Authentication': $("#hajg_authentication").val(), //申请凭证
            'MsgKey': $("#hajg_msg_key").val(),//消息可以
            'ApprovalMsg': $("#hajg_approval_msg").val(), //处理附言
            'UserDefinedField': $("#hajg_group_id").val()//用户自定义字段
        };
        webim.handleApplyJoinGroup(
                options,
                function (resp) {
                    //在表格中删除对应的行

$('#get_apply_join_group_pendency_table').bootstrapTable('remove', {
                            field: 'Authentication',
                            values: [$("#hajg_authentication").val()]
                        });
                        $('#handle_ajg_dialog').modal('hide');
                        alert('处理加群申请成功');
                },
                function (err) {
                        alert(err.ErrorInfo);
                }
        );
    };
```

8.5. 删除加群申请

在处理完加群申请之后，需要删除对应的加群申请

```
/* function deleteApplyJoinGroupPendency
 *     删除加群申请
 * params:
 *     options - 请求参数，详见 api 文档
 *     cbOk    - function()类型，成功时回调函数
 *     cbErr   - function(err)类型，失败时回调函数, err 为错误对象
 * return:
 *     (无)
 */
deleteApplyJoinGroupPendency: function(options, cbOk, cbErr) {},
```

其中 options 定义如下：

```
//要删除的群未决消息(支持批量删除)
```

```
var options = {
    //需要删除的消息列表
    'DelMsgList': [
        {
            "From_Account":“@TIM#SYSTEM”,//消息发送者
            "MsgSeq": 345,//消息序列号
            "MsgRandom": 1234//消息随机数
        }
    ]
};
```

示例代码：

```
//删除已处理的加群未决消息
var deleteApplyJoinGroupPendency = function (opts) {

    webim.deleteApplyJoinGroupPendency(opts,
            function (resp) {
                console.info('delete group pendency msg success');
            },
            function (err) {
                alert(err.ErrorInfo);
                console.error('delete group pendency msg failed');
            }
    );
    return;
};
```

8.6. 主动退群

```
/* function quitGroup
     *    主动退群
     * params:
     *     options  -  请求参数，详见 api 文档
     *     cbOk     - function()类型，成功时回调函数
     *     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *     (无)
     */
    quitGroup: function(options,cbOk, cbErr) {},
```

示例：

```
//退群
var quitGroup = function (group_id) {
```

```
    var options = null;
    if (group_id) {
        options = {
            'GroupId': group_id
        };
    }
    if (options == null) {
        alert('退群时，群组 ID 非法');
        return;
    }
    webim.quitGroup(
            options,
            function (resp) {
                //在表格中删除对应的行
                $('#get_my_group_table').bootstrapTable('remove', {
                    field: 'GroupId',
                    values: [group_id]
                });
                //刷新我的群组列表
                getJoinedGroupListHigh(getGroupsCallbackOK);
            },
            function (err) {
                alert(err.ErrorInfo);
            }
    );
};
```

8.7. 解散群

```
/* function destroyGroup
     *    解散群
     * params:
     *     options  -  请求参数，详见 api 文档
     *     cbOk     - function()类型，成功时回调函数
     *     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *     (无)
     */
    destroyGroup: function(options,cbOk, cbErr) {},
```

示例：

```
//解散群组
var destroyGroup = function (group_id) {
    var options = null;
```

```
    if (group_id) {
        options = {
            'GroupId': group_id
        };
    }
    if (options == null) {
        alert('解散群时，群组 ID 非法');
        return;
    }
    webim.destroyGroup(
            options,
            function (resp) {
                //在表格中删除对应的行
                $('#get_my_group_table').bootstrapTable('remove', {
                    field: 'GroupId',
                    values: [group_id]
                });
                //读取我的群组列表
                getJoinedGroupListHigh(getGroupsCallbackOK);
            },
            function (err) {
                alert(err.ErrorInfo);
            }
    );
};
```

8.8. 我的群组列表

```
/* function getJoinedGroupListHigh
     *     获取我的群组-高级接口
     * params:
     *     options  -  请求参数，详见 api 文档
     *     cbOk     - function()类型，成功时回调函数
     *     cbErr    - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *     (无)
     */
    getJoinedGroupListHigh: function(options,cbOk, cbErr) {},
```

示例：

```
//获取我的群组
var getMyGroup = function () {
    initGetMyGroupTable([]);
    var options = {
```

```
            'Member_Account': loginInfo.identifier,
            'Limit': totalCount,
            'Offset': 0,
            //'GroupType':'',
            'GroupBaseInfoFilter': [
                'Type',
                'Name',
                'Introduction',
                'Notification',
                'FaceUrl',
                'CreateTime',
                'Owner_Account',
                'LastInfoTime',
                'LastMsgTime',
                'NextMsgSeq',
                'MemberNum',
                'MaxMemberNum',
                'ApplyJoinOption'
            ],
            'SelfInfoFilter': [
                'Role',
                'JoinTime',
                'MsgFlag',
                'UnreadMsgNum'
            ]
        };
        webim.getJoinedGroupListHigh(
                options,
                function (resp) {
                    if (!resp.GroupIdList || resp.GroupIdList.length == 0) {
                        alert('你目前还没有加入任何群组');
                        return;
                    }
                    var data = [];
                    for (var i = 0; i < resp.GroupIdList.length; i++) {

                        var group_id = resp.GroupIdList[i].GroupId;
                        var name =
webim.Tool.formatText2Html(resp.GroupIdList[i].Name);
                        var type_en = resp.GroupIdList[i].Type;
                        var type =
webim.Tool.groupTypeEn2Ch(resp.GroupIdList[i].Type);
                        var role_en = resp.GroupIdList[i].SelfInfo.Role;
                        var role =
```

```
webim.Tool.groupRoleEn2Ch(resp.GroupIdList[i].SelfInfo.Role);
                    var msg_flag = webim.Tool.groupMsgFlagEn2Ch(resp.GroupIdList[i].SelfInfo.MsgFlag);
                    var msg_flag_en = resp.GroupIdList[i].SelfInfo.MsgFlag;
                    var join_time = webim.Tool.formatTimeStamp(resp.GroupIdList[i].SelfInfo.JoinTime);
                    var member_num = resp.GroupIdList[i].MemberNum;
                    var notification = webim.Tool.formatText2Html(resp.GroupIdList[i].Notification);
                    var introduction = webim.Tool.formatText2Html(resp.GroupIdList[i].Introduction);
                    data.push({
                        'GroupId': group_id,
                        'Name': name,
                        'TypeEn': type_en,
                        'Type': type,
                        'RoleEn': role_en,
                        'Role': role,
                        'MsgFlagEn': msg_flag_en,
                        'MsgFlag': msg_flag,
                        'MemberNum': member_num,
                        'Notification': notification,
                        'Introduction': introduction,
                        'JoinTime': join_time
                    });
                }
                //打开我的群组列表对话框
                $('#get_my_group_table').bootstrapTable('load', data);
                $('#get_my_group_dialog').modal('show');
            },
            function (err) {
                alert(err.ErrorInfo);
            }
        );
    };
```

8.9. 读取群详细资料

```
    /* function getGroupInfo
     *    读取群详细资料-高级接口
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
```

```
     *     (无)
     */
    getGroupInfo: function(options,cbOk, cbErr) {},
```

示例：

```
//读取群组基本资料-高级接口
var getGroupInfo = function (group_id, cbOK, cbErr) {
    var options = {
        'GroupIdList': [
            group_id
        ],
        'GroupBaseInfoFilter': [
            'Type',
            'Name',
            'Introduction',
            'Notification',
            'FaceUrl',
            'CreateTime',
            'Owner_Account',
            'LastInfoTime',
            'LastMsgTime',
            'NextMsgSeq',
            'MemberNum',
            'MaxMemberNum',
            'ApplyJoinOption'
        ],
        'MemberInfoFilter': [
            'Account',
            'Role',
            'JoinTime',
            'LastSendMsgTime',
            'ShutUpUntil'
        ]
    };
    webim.getGroupInfo(
            options,
            function (resp) {
                if (cbOK) {
                    cbOK(resp);
                }
            },
            function (err) {
                alert(err.ErrorInfo);
            }
```

```
    );
};
```

8.10. 修改群基本资料

```
/* function modifyGroupBaseInfo
     *    修改群基本资料
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *    (无)
     */
    modifyGroupBaseInfo: function(options,cbOk, cbErr) {},
```

示例：

```
//修改群资料
var modifyGroup = function () {
    if ($("#mg_name").val().length == 0) {
        alert('请输入群组名称');
        return;
    }
    if (webim.Tool.trimStr($("#mg_name").val()).length == 0) {
        alert('您输入的群组名称全是空格,请重新输入');
        return;
    }
    if (webim.Tool.getStrBytes($("#fsm_name").val()) > 30) {
        alert('您输入的群组名称超出限制(最长 10 个汉字)');
        return;
    }
    if (webim.Tool.getStrBytes($("#fsm_notification").val()) > 150) {
        alert('您输入的群组公告超出限制(最长 50 个汉字)');
        return;
    }
    if (webim.Tool.getStrBytes($("#fsm_introduction").val()) > 120) {
        alert('您输入的群组简介超出限制(最长 40 个汉字)');
        return;
    }
    var options = {
        'GroupId': $('#mg_group_id').val(),
        'Name': $('#mg_name').val(),
        //'FaceUrl': $('#mg_face_url').val(),
        'Notification': $('#mg_notification').val(),
```

```
            'Introduction': $('#mg_introduction').val()
        };
        webim.modifyGroupBaseInfo(
                options,
                function (resp) {
                    //在表格中修改对应的行
                    $('#get_my_group_table').bootstrapTable('updateRow', {
                        index: $('#mg_sel_row_index').val(),
                        row: {
                            Type: $('input[name="mg_type_radio"]:checked').val(),
                            Name:
webim.Tool.formatText2Html($('#mg_name').val()),
                            Introduction:
webim.Tool.formatText2Html($('#mg_introduction').val()),
                            Notification:
webim.Tool.formatText2Html($('#mg_notification').val())
                        }
                    });
                    $('#modify_group_dialog').modal('hide');
                    alert('修改群资料成功');
                },
                function (err) {
                    alert(err.ErrorInfo);
                }
        );
    };
```

9. 群成员管理

9.1. 获取群成员列表

```
    /* function getGroupMemberInfo
     *    获取群组成员列表
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *    (无)
     */
    getGroupMemberInfo: function(options,cbOk, cbErr) {},
```

示例：

```
//读取群组成员
```

```
var getGroupMemberInfo = function (group_id) {
    initGetGroupMemberTable([]);
    var options = {
        'GroupId': group_id,
        'Offset': 0, //必须从 0 开始
        'Limit': totalCount,
        'MemberInfoFilter': [
            'Account',
            'Role',
            'JoinTime',
            'LastSendMsgTime',
            'ShutUpUntil'
        ]
    };
    webim.getGroupMemberInfo(
            options,
            function (resp) {
                if (resp.MemberNum <= 0) {
                    alert('该群组目前没有成员');
                    return;
                }
                var data = [];
                for (var i in resp.MemberList) {
                    var account = resp.MemberList[i].Member_Account;
                    var role = webim.Tool.groupRoleEn2Ch(resp.MemberList[i].Role);
                    var join_time = webim.Tool.formatTimeStamp(resp.MemberList[i].JoinTime);
                    var shut_up_until = webim.Tool.formatTimeStamp(resp.MemberList[i].ShutUpUntil);
                    if (shut_up_until == 0) {
                        shut_up_until = '-';
                    }
                    data.push({
                        GroupId: group_id,
                        Member_Account: account,
                        Role: role,
                        JoinTime: join_time,
                        ShutUpUntil: shut_up_until
                    });
                }
                $('#get_group_member_table').bootstrapTable('load', data);
                $('#get_group_member_dialog').modal('show');
            },
```

```
            function (err) {
                alert(err.ErrorInfo);
            }
    );
};
```

9.2. 邀请好友加群

```
/* function addGroupMember
     *    邀请好友加群
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *    (无)
     */
    addGroupMember: function(options,cbOk, cbErr) {},
```

示例：

```
//邀请好友加群
var addGroupMember = function () {
    var options = {
        'GroupId': $('#agm_group_id').val(),
        'MemberList': [
            {
                'Member_Account': $('#agm_account').val()
            }

        ]
    };
    webim.addGroupMember(
            options,
            function (resp) {
                //在表格中删除对应的行
                $('#get_my_friend_group_table').bootstrapTable('remove', {
                    field: 'Info_Account',
                    values: [$('#agm_account').val()]
                });
                $('#add_group_member_dialog').modal('hide');
                alert('邀请好友加群成功');
            },
            function (err) {
                alert(err.ErrorInfo);
```

```
            }
        );
    };
```

9.3. 修改群消息提示

```
    /* function modifyGroupMember
     *    修改群成员资料（角色或者群消息提类型示）
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *    (无)
     */
    modifyGroupMember: function(options,cbOk, cbErr) {},
```

示例：

```
//修改群消息提示类型
var modifyGroupMsgFlag = function () {
    var msg_flag_en = $('input[name="mgmf_msg_flag_radio"]:checked').val();
    var msg_flag_zh = webim.Tool.groupMsgFlagEn2Ch(msg_flag_en);
    var options = {
        'GroupId': $('#mgmf_group_id').val(),
        'Member_Account': loginInfo.identifier,
        'MsgFlag': msg_flag_en
    };
    webim.modifyGroupMember(
            options,
            function (resp) {
                //在表格中修改对应的行
                $('#get_my_group_table').bootstrapTable('updateRow', {
                    index: $('#mgmf_sel_row_index').val(),
                    row: {
                        MsgFlag: msg_flag_zh,
                        MsgFlagEn: msg_flag_en
                    }
                });
                $('#modify_group_msg_flag_dialog').modal('hide');
                alert('设置群消息提示类型成功');
            },
            function (err) {
                alert(err.ErrorInfo);
            }
    );
```

```
};
```

9.4. 修改群成员角色

```
/* function modifyGroupMember
     *    修改群成员资料（角色或者群消息提类型示）
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *    (无)
     */
    modifyGroupMember: function(options,cbOk, cbErr) {},
```

示例：

```
//修改群组成员角色
var modifyGroupMemberRole = function () {
    var role_en = $('input[name="mgm_role_radio"]:checked').val();
    var role_zh = webim.Tool.groupRoleEn2Ch(role_en);
    var options = {
        'GroupId': $('#mgm_group_id').val(),
        'Member_Account': $('#mgm_account').val(),
        'Role': role_en
    };
    webim.modifyGroupMember(
            options,
            function (resp) {
                //在表格中修改对应的行
                $('#get_group_member_table').bootstrapTable('updateRow', {
                    index: $('#mgm_sel_row_index').val(),
                    row: {
                        Role: role_zh
                    }
                });
                $('#modify_group_member_dialog').modal('hide');
                alert('修改群成员角色成功');
            },
            function (err) {
                alert(err.ErrorInfo);
            }
    );
};
```

9.5. 设置群成员禁言时间

```
/* function forbidSendMsg
 *     设置群成员禁言时间
 * params:
 *     options - 请求参数，详见 api 文档
 *     cbOk    - function()类型，成功时回调函数
 *     cbErr   - function(err)类型，失败时回调函数, err 为错误对象
 * return:
 *     (无)
 */
forbidSendMsg: function(options,cbOk, cbErr) {},
```

示例：

```
//设置成员禁言时间
var forbidSendMsg = function () {
    if (!webim.Tool.validNumber($('#fsm_shut_up_time').val())) {
        alert('您输入的禁言时间非法,只能是数字(0-31536000)');
        return;
    }
    var shut_up_time = parseInt($('#fsm_shut_up_time').val());
    if (shut_up_time > 31536000) {
        alert('您输入的禁言时间非法,只能是数字(0-31536000)');
        return;
    }
    var shut_up_until = '-';
    if (shut_up_time != 0) {
        //当前时间+禁言时间=禁言截至时间
        shut_up_until = webim.Tool.formatTimeStamp(Math.round(new
Date().getTime() / 1000) + shut_up_time);
    }
    var options = {
        'GroupId': $('#fsm_group_id').val(),
        'Members_Account': [$('#fsm_account').val()],
        'ShutUpTime': shut_up_time
    };
    webim.forbidSendMsg(
            options,
            function (resp) {
                //在表格中修改对应的行
                $('#get_group_member_table').bootstrapTable('updateRow', {
                    index: $('#fsm_sel_row_index').val(),
                    row: {
                        ShutUpUntil: shut_up_until
                    }
```

```
                    });
                    $('#forbid_send_msg_dialog').modal('hide');
                    alert('设置成员禁言时间成功');
                },
                function (err) {
                    alert(err.ErrorInfo);
                }
        );
    };
```

9.6. 删除群成员

```
    /* function deleteGroupMember
     *    删除群成员
     * params:
     *    options - 请求参数，详见 api 文档
     *    cbOk    - function()类型，成功时回调函数
     *    cbErr   - function(err)类型，失败时回调函数, err 为错误对象
     * return:
     *    (无)
     */
    deleteGroupMember: function(options,cbOk, cbErr) {},
```

示例：

```
//删除群组成员
var deleteGroupMember = function () {
    if (!confirm("确定移除该成员吗？")) {
        return;
    }
    var options = {
        'GroupId': $('#dgm_group_id').val(),
        //'Silence':  $('input[name="dgm_silence_radio"]:checked').val(),//只有 ROOT
用户采用权限设置该字段（是否静默移除）
        'MemberToDel_Account': [$('#dgm_account').val()]
    };
    webim.deleteGroupMember(
            options,
            function (resp) {
                //在表格中删除对应的行
                $('#get_group_member_table').bootstrapTable('remove', {
                    field: 'Member_Account',
                    values: [$('#dgm_account').val()]
                });
                $('#delete_group_member_dialog').modal('hide');
                alert('移除群成员成功');
```

```
            },
            function (err) {
                alert(err.ErrorInfo);
            }
        );
};
```

10. **群提示消息**

当有用户被邀请加入群组，或者有用户被移出群组时，群内会产生有提示消息，调用方可以根据需要展示给群组用户，或者忽略。

如下图中，展示一条用户主动退群的提示消息：

群提示消息类型定义如下：

```
//群提示消息类型
var WEB_IM_GROUP_TIP_TYPE={
    "JOIN":1,//加入群组
    "QUIT":2,//退出群组
    "KICK":3,//被踢出群组
    "SET_ADMIN":4,//被设置为管理员
    "CANCEL_ADMIN":5,//被取消管理员
    "MODIFY_GROUP_INFO":6,//修改群资料
    "MODIFY_MEMBER_INFO":7//修改群成员信息
};
```

群提示消息对象定义如下：

```
// class Msg.Elem.GroupTip  群提示消息对象
Msg.Elem.GroupTip = function (opType, opUserId, groupId, groupName, userIdList) {
```

```
    this.opType = opType;//操作类型
    this.opUserId = opUserId;//操作者 id
    this.groupId = groupId;//群 id
    this.groupName = groupName;//群名称
    this.userIdList = userIdList ? userIdList : [];//被操作的用户 id 列表
    this.groupInfoList = [];//新的群资料信息，群资料变更时才有值
    this.memberInfoList = [];//新的群成员资料信息，群成员资料变更时才有值
};
Msg.Elem.GroupTip.prototype.addGroupInfo = function (groupInfo) {
    this.groupInfoList.push(groupInfo);
};
Msg.Elem.GroupTip.prototype.addMemberInfo = function (memberInfo) {
    this.memberInfoList.push(memberInfo);
};
Msg.Elem.GroupTip.prototype.getOpType = function () {
    return this.opType;
};
Msg.Elem.GroupTip.prototype.getOpUserId = function () {
    return this.opUserId;
};
Msg.Elem.GroupTip.prototype.getGroupId = function () {
    return this.groupId;
};
Msg.Elem.GroupTip.prototype.getGroupName = function () {
    return this.groupName;
};
Msg.Elem.GroupTip.prototype.getUserIdList = function () {
    return this.userIdList;
};
Msg.Elem.GroupTip.prototype.getGroupInfoList = function () {
    return this.groupInfoList;
};
Msg.Elem.GroupTip.prototype.getMemberInfoList = function () {
    return this.memberInfoList;
};
Msg.Elem.GroupTip.prototype.toHtml = function () {
    var text = "[群提示消息]";
    var maxIndex = WEB_IM_GROUP_TIP_MAX_USER_COUNT - 1;
    switch (this.opType) {
        case WEB_IM_GROUP_TIP_TYPE.JOIN://退出群
            //代码省略
            break;
        case WEB_IM_GROUP_TIP_TYPE.QUIT://退出群
            //代码省略
```

```
                break;
            case WEB_IM_GROUP_TIP_TYPE.KICK://踢出群
                //代码省略
                break;
            case WEB_IM_GROUP_TIP_TYPE.SET_ADMIN://设置管理员
                //代码省略
                break;
            case WEB_IM_GROUP_TIP_TYPE.CANCEL_ADMIN://取消管理员
                //代码省略
                break;
            case WEB_IM_GROUP_TIP_TYPE.MODIFY_GROUP_INFO://群资料变更
                //代码省略
                break;
            case  WEB_IM_GROUP_TIP_TYPE.MODIFY_MEMBER_INFO://群成员资料变更(禁言时间)
                //代码省略
                break;
            default:
                text += "未知群提示消息类型：type=" + this.opType;
                break;
        }
        return text;
    };
```

10.1. 用户被邀请加入群组

触发时机：当有用户被邀请加入群组时，群组内会由系统发出通知，开发者可选择展示样式。可以更新群成员列表。

收到的消息 type 为 WEB_IM_GROUP_TIP_TYPE.JOIN。

Msg.Elem.GroupTip 成员方法：

getOpType： WEB_IM_GROUP_TIP_TYPE.JOIN

getOpUserId ： 邀请人 id

getGroupName ： 群名

getUserIdList： 被邀请入群的用户 id 列表

示例：

```
case WEB_IM_GROUP_TIP_TYPE.JOIN://加入群
                    text += this.opUserId + "邀请了";
                    for (var m in this.userIdList) {
                        text += this.userIdList[m] + ",";
                        if (this.userIdList.length > WEB_IM_GROUP_TIP_MAX_USER_COUNT && m == maxIndex) {
                            text += "等" + this.userIdList.length + "人";
```

```
                    break;
                }
            }
            text += "加入该群";
            break;
```

10.2. 用户主动退出群组

触发时机：当有用户主动退群时，群组内会由系统发出通知。可以选择更新群成员列表。

收到的消息 type 为 WEB_IM_GROUP_TIP_TYPE.QUIT。

Msg.Elem.GroupTip 成员方法：

getOpType： WEB_IM_GROUP_TIP_TYPE.QUIT

getOpUserId ： 退群用户 id

getGroupName ： 群名

示例：

```
case WEB_IM_GROUP_TIP_TYPE.QUIT://退出群
            text += this.opUserId + "主动退出该群";
            break;
```

10.3. 用户被踢出群组

触发时机：当有用户被踢出群组时，群组内会由系统发出通知。可以更新群成员列表。

收到的消息 type 为 WEB_IM_GROUP_TIP_TYPE.KICK。

Msg.Elem.GroupTip 成员方法：

getOpType： WEB_IM_GROUP_TIP_TYPE.KICK

getOpUserId ： 踢人 id

getGroupName ： 群名

getUserIdList： 被踢出群的用户 id 列表

示例：

```
case WEB_IM_GROUP_TIP_TYPE.KICK://踢出群
            text += this.opUserId + "将";
            for (var m in this.userIdList) {
                text += this.userIdList[m] + ",";
                if (this.userIdList.length > WEB_IM_GROUP_TIP_MAX_USER_COUNT && m == maxIndex) {
                    text += "等" + this.userIdList.length + "人";
                    break;
                }
            }
            text += "踢出该群";
```

10.4. 用户被设置成管理员

触发时机：当有用户被设置为管理员时，群组内会由系统发出通知。如果界面有显示是否管理员，此时可更新管理员标识

收到的消息 type 为 WEB_IM_GROUP_TIP_TYPE.SET_ADMIN。

Msg.Elem.GroupTip 成员方法：

getOpType： WEB_IM_GROUP_TIP_TYPE.SET_ADMIN

getOpUserId ： 设置者 id

getGroupName ： 群名

getUserIdList： 被设置成管理员的用户 id 列表

示例：

```
case WEB_IM_GROUP_TIP_TYPE.SET_ADMIN://设置管理员
                                text += this.opUserId + "将";
                                for (var m in this.userIdList) {
                                    text += this.userIdList[m] + ",";
                                    if (this.userIdList.length > WEB_IM_GROUP_TIP_MAX_USER_COUNT && m == maxIndex) {
                                        text += "等" + this.userIdList.length + "人";
                                        break;
                                    }
                                }
                                text += "设为管理员";
                                break;
```

10.5. 用户被取消管理员身份

触发时机：当有用户被被取消管理员身份时，群组内会由系统发出通知。如果界面有显示是否管理员，此时可更新管理员标识

收到的消息 type 为 WEB_IM_GROUP_TIP_TYPE.CANCEL_ADMIN。

Msg.Elem.GroupTip 成员方法：

getOpType： WEB_IM_GROUP_TIP_TYPE.CANCEL_ADMIN

getOpUserId ： 取消者 id

getGroupName ： 群名

getUserIdList： 被取消管理员身份的用户 id 列表

示例：

```
case WEB_IM_GROUP_TIP_TYPE.CANCEL_ADMIN://取消管理员
                                text += this.opUserId + "取消";
                                for (var m in this.userIdList) {
                                    text += this.userIdList[m] + ",";
```

```
                    if (this.userIdList.length > WEB_IM_GROUP_TIP_MAX_USER_COUNT && m == maxIndex) {
                        text += "等" + this.userIdList.length + "人";
                        break;
                    }
                }
                text += "的管理员资格";
                break;
```

10.6. 群组资料变更

触发时机：当群资料变更，如群名、群简介等，会有系统消息发出，可更新相关展示字段，或者选择性把消息展示给用户。

收到的消息 type 为 WEB_IM_GROUP_TIP_TYPE.MODIFY_GROUP_INFO。

Msg.Elem.GroupTip 成员方法：

getOpType： WEB_IM_GROUP_TIP_TYPE.MODIFY_GROUP_INFO

getOpUserId ： 修改群资料的用户 id

getGroupName ： 群名

getGroupInfoList： 群变更的具体资料信息，为 Msg.Elem.GroupTip.GroupInfo 对象列表

变更的群资料信息对象定义如下：

```
// class Msg.Elem.GroupTip.GroupInfo，变更的群资料信息对象
    Msg.Elem.GroupTip.GroupInfo = function(type,value) {
        this.type = type;//群资料信息类型
        this.value = value;//对应的值
    };
        Msg.Elem.GroupTip.GroupInfo.prototype.getType = function() {
                return this.type;
    };
        Msg.Elem.GroupTip.GroupInfo.prototype.getValue = function() {
                return this.value;
    };
```

群提示消息-群资料变更类型定义如下：

```
    //群提示消息-群资料变更类型
    var WEB_IM_GROUP_TIP_MODIFY_GROUP_INFO_TYPE={
        "FACE_URL":1,//修改群头像 URL
        "NAME":2,//修改群名称
        "OWNER":3,//修改群主
        "NOTIFICATION":4,//修改群公告
        "INTRODUCTION":5//修改群简介
    };
```

示例：

```
case WEB_IM_GROUP_TIP_TYPE.MODIFY_GROUP_INFO://群资料变更
                    text += this.opUserId + "修改了群资料：";
                    for (var m in this.groupInfoList) {
                        var type=this.groupInfoList[m].getType();
                        var value=this.groupInfoList[m].getValue();
                        switch(type){
                            case
WEB_IM_GROUP_TIP_MODIFY_GROUP_INFO_TYPE.FACE_URL:
                                text += "群头像为" + value + "; ";
                                break;
                            case
WEB_IM_GROUP_TIP_MODIFY_GROUP_INFO_TYPE.NAME:
                                text += "群名称为" + value + "; ";
                                break;
                            case
WEB_IM_GROUP_TIP_MODIFY_GROUP_INFO_TYPE.OWNER:
                                text += "群主为" + value + "; ";
                                break;
                            case
WEB_IM_GROUP_TIP_MODIFY_GROUP_INFO_TYPE.NOTIFICATION:
                                text += "群公告为" + value + "; ";
                                break;
                            case
WEB_IM_GROUP_TIP_MODIFY_GROUP_INFO_TYPE.INTRODUCTION:
                                text += "群简介为" + value + "; ";
                                break;
                            default:
                                text += "未知信息为:type=" +
type+",value="+value + "; ";
                                break;
                        }
                    }
                    break;
```

当收到群组资料变更提示消息时，客户端需要监听这种消息，并进行相关处理。

demo 的做法是先定义监听群组资料变更事件，在初始化的时候，传给 sdk，当 sdk 收到群组资料变更消息时进行回调。

首先定义监听群组资料变更事件方法 onGroupInfoChangeNotify(notify)，其中 notify 格式如下：

```
var notify = {
    "GroupId": group_id,//群 id
```

```
        "GroupFaceUrl": null,// //新群组图标，为空，则表示没有变化
        "GroupName": null, //新群名称，为空，则表示没有变化
        "OwnerAccount": null, //新的群主 id，为空，则表示没有变化
        "GroupNotification": null, //新的群公告，为空，则表示没有变化
        "GroupIntroduction": null//新的群简介，为空，则表示没有变化
    };
```

示例如下：

```
    //监听 群资料变化 群提示消息
    function onGroupInfoChangeNotify(notify) {
        console.info("执行 群资料变化 回调： %s", JSON.stringify(notify));
        var groupId = notify.GroupId;//群 ID
        var newFaceUrl = notify.GroupFaceUrl;//新群组图标，为空，则表示没有变化
        var newName = notify.GroupName;//新群名称，为空，则表示没有变化
        var newOwner = notify.OwnerAccount;//新的群主 id，为空，则表示没有变化
        var newNotification = notify.GroupNotification;//新的群公告，为空，则表示没有变化
        var newIntroduction = notify.GroupIntroduction;//新的群简介，为空，则表示没有变化

        if (newName) {
            //更新群组列表的群名称
            var groupNameDivId = "nameDiv_" + groupId;
            var groupNameDiv = document.getElementById(groupNameDivId);
            if (groupNameDiv) {
                if (newName.length > maxNameLen) {//帐号或昵称过长，截取一部分
                    newName = newName.substr(0, maxNameLen) + "...";
                }
                groupNameDiv.innerHTML = newName;
            } else {
                console.warn("不存在该群组 div: groupNameDivId=" + groupNameDivId);
            }
        }
    }
```

然后在初始化的时候，将 onGroupInfoChangeNotify 传给 sdk，如下：

```
    //监听事件 var listeners = { "onConnNotify": null, "onMsgNotify": onMsgNotify,//监听新消息(私聊，群聊，群提示消息)事件 "onGroupInfoChangeNotify": onGroupInfoChangeNotify,// 监听群资料变化事件 "groupSystemNotifys": groupSystemNotifys//监听（多终端同步）群系统消息事件 };
```

10.7. 群成员资料变更

触发时机：当群成员的群相关资料变更时，会有系统消息发出，可更新相关字段展示，或者选择性把消息展示给用户。（注意：这里的资料仅跟群相关资料，比如禁言时间、成员角色变更等，不包括用户昵称等本身资料，目前只支持禁言时间通知）。

收到的消息 type 为 WEB_IM_GROUP_TIP_TYPE.MODIFY_MEMBER_INFO。

Msg.Elem.GroupTip 成员方法：

getOpType： WEB_IM_GROUP_TIP_TYPE.MODIFY_MEMBER_INFO

getOpUserId ： 修改者 id

getGroupName ： 群名

getMemberInfoList ： 变更的群成员的具体资料信息，为 Msg.Elem.GroupTip.MemberInfo 对象列表

变更的群成员资料信息对象定义如下：

```
// class Msg.Elem.GroupTip.MemberInfo，变更的群成员资料信息对象
                    Msg.Elem.GroupTip.MemberInfo = function(userId,shutupTime){
                      this.userId = userId;//群成员 id
                      this.shutupTime = shutupTime;//群成员被禁言时间，0 表示取消禁言，大于 0 表示被禁言时长，单位：秒
                    };
            Msg.Elem.GroupTip.MemberInfo.prototype.getUserId = function() {
                    return this.userId;
                    };
            Msg.Elem.GroupTip.MemberInfo.prototype.getShutupTime = function() {
                    return this.shutupTime;
                    };
```

示例：

```
case WEB_IM_GROUP_TIP_TYPE.MODIFY_MEMBER_INFO://群成员资料变更(禁言时间)
                                text += this.opUserId + "修改了群成员资料:";
                                for (var m in this.memberInfoList) {
                                    var userId=this.memberInfoList[m].getUserId();
                                    var shutupTime=this.memberInfoList[m].getShutupTime();
                                    text += userId+": ";
                                    if (shutupTime != null && shutupTime !== undefined) {
                                        if (shutupTime == 0) {
                                            text += "取消禁言; ";
                                        } else {
```

```
                        text += "禁言" + shutupTime + "秒; ";
                    }
                } else{
                    text += " shutupTime 为空";
                }
                if (this.memberInfoList.length > WEB_IM_GROUP_TIP_MAX_USER_COUNT && m == maxIndex) {
                    text += "等" + this.memberInfoList.length + "人";
                    break;
                }
            }
            break;
```

11. 群系统消息

当有用户申请加群等事件发生时，管理员会收到邀请加群系统消息，相应的消息会通过群系统消息展示给用户。群系统消息类型主要有以下几种：

```
//群系统消息类型
var WEB_IM_GROUP_SYSTEM_TYPE={
    "JOIN_GROUP_REQUEST":1,//申请加群请求（只有管理员会收到）
    "JOIN_GROUP_ACCEPT":2,//申请加群被同意（只有申请人能够收到）
    "JOIN_GROUP_REFUSE":3,//申请加群被拒绝（只有申请人能够收到）
    "KICK":4,//被管理员踢出群(只有被踢者接收到)
    "DESTORY":5,//群被解散(全员接收)
    "CREATE":6,//创建群(创建者接收, 不展示)
    "INVITED_JOIN_GROUP_REQUEST":7,//邀请加群(被邀请者接收)
    "QUIT":8,//主动退群(主动退出者接收, 不展示)
    "SET_ADMIN":9,//设置管理员(被设置者接收)
    "CANCEL_ADMIN":10,//取消管理员(被取消者接收)
    "REVOKE":11,//群已被回收(全员接收, 不展示)
    "CUSTOM":255//用户自定义通知(默认全员接收)
};
```

目前是通过定义群系统消息监听事件来处理群系统消息的。

示例：

```
//监听（多终端同步）群系统消息事件
var groupSystemNotifys = {
    "1": onApplyJoinGroupRequestNotify, //申请加群请求（只有管理员会收到）
    "2": onApplyJoinGroupAcceptNotify, //申请加群被同意（只有申请人能够收到）
    "3": onApplyJoinGroupRefuseNotify, //申请加群被拒绝（只有申请人能够收到）
    "4": onKickedGroupNotify, //被管理员踢出群(只有被踢者接收到)
    "5": onDestoryGroupNotify, //群被解散(全员接收)
```

```
    "6": onCreateGroupNotify, //创建群(创建者接收)
    "7": onInvitedJoinGroupNotify, //邀请加群(被邀请者接收)
    "8": onQuitGroupNotify, //主动退群(主动退出者接收)
    "9": onSetedGroupAdminNotify, //设置管理员(被设置者接收)
    "10": onCanceledGroupAdminNotify, //取消管理员(被取消者接收)
    "11": onRevokeGroupNotify, //群已被回收(全员接收)
    "255": onCustomGroupNotify//用户自定义通知(默认全员接收)
};
```

Demo 提供了查看群系统消息通知的入口：

11.1. 申请加群

触发时机：当有用户申请加群时，群管理员会收到申请加群消息，管理员决定是否同意对方加群。

Demo 提供了查看加群申请的入口：

示例：

```
//监听  申请加群  系统消息
function onApplyJoinGroupRequestNotify(notify) {
    console.info("执行  加群申请  回调：  %s", JSON.stringify(notify));
    var data = [];
    var timestamp = notify.MsgTime;
    notify.MsgTimeStamp = timestamp;
    notify.MsgTime = webim.Tool.formatTimeStamp(notify.MsgTime);
    data.push(notify);
    $('#get_apply_join_group_pendency_table').bootstrapTable('append', data);
    $('#get_apply_join_group_pendency_dialog').modal('show');
```

```
        var reportTypeCh = "[申请加群]";
        var content = notify.Operator_Account + "申请加入你的群";
        addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, timestamp);
    }
```

11.2. 申请加群被同意

触发时机：当管理员同意加群请求时，申请人会收到同意加群的消息。

示例：

```
//监听 申请加群被同意 系统消息
function onApplyJoinGroupAcceptNotify(notify) {
    console.info("执行 申请加群被同意 回调： %s", JSON.stringify(notify));
    //刷新我的群组列表
    getJoinedGroupListHigh(getGroupsCallbackOK);

    var reportTypeCh = "[申请加群被同意]";
    var content = notify.Operator_Account + "同意你的加群申请，附言：" +
notify.RemarkInfo;
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);

}
```

11.3. 申请加群被拒绝

触发时机：当管理员拒绝时，申请人会收到拒绝加群的消息。

示例：

```
//监听 申请加群被拒绝 系统消息
function onApplyJoinGroupRefuseNotify(notify) {
    console.info("执行 申请加群被拒绝 回调： %s", JSON.stringify(notify));
    var reportTypeCh = "[申请加群被拒绝]";
    var content = notify.Operator_Account + "拒绝了你的加群申请，附言：" +
notify.RemarkInfo;
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

11.4. 被管理员踢出群

触发时机：当用户被管理员踢出群组时，用户会收到被踢出群的消息。

示例：

```
//监听 被踢出群 系统消息
function onKickedGroupNotify(notify) {
    console.info("执行 被踢出群 回调： %s", JSON.stringify(notify));
    //刷新我的群组列表
```

```
        getJoinedGroupListHigh(getGroupsCallbackOK);
        var reportTypeCh = "[被踢出群]";
        var content = "你被管理员" + notify.Operator_Account + "踢出该群";
        addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
    }
```

11.5. 解散群

触发时机：当群被解散时，全员会收到解散群消息。

示例：

```
//监听 解散群 系统消息
function onDestoryGroupNotify(notify) {
    console.info("执行 解散群 回调： %s", JSON.stringify(notify));
    //刷新我的群组列表
    getJoinedGroupListHigh(getGroupsCallbackOK);
    var reportTypeCh = "[群被解散]";
    var content = "群主" + notify.Operator_Account + "已解散该群";
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

11.6. 创建群

触发时机：当创建群成功时，创建者会收到创建群消息。

示例：

```
//监听 创建群 系统消息
function onCreateGroupNotify(notify) {
    console.info("执行 创建群 回调： %s", JSON.stringify(notify));
    var reportTypeCh = "[创建群]";
    var content = "你创建了该群";
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

11.7. 被邀请加群

触发时机：当用户被邀请加入群组时，该用户会收到邀请消息，注意：创建群组时初始成员无需邀请即可入群。

示例：

```
//监听 被邀请加群 系统消息
function onInvitedJoinGroupNotify(notify) {
    console.info("执行 被邀请加群 回调： %s", JSON.stringify(notify));
    //刷新我的群组列表
    getJoinedGroupListHigh(getGroupsCallbackOK);
    var reportTypeCh = "[被邀请加群]";
```

```
        var content = "你被管理员" + notify.Operator_Account + "邀请加入该群";
        addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
    }
```

11.8. 主动退群

触发时机：当用户主动退出群组时，该用户会收到退群消息，只有退群的用户自己可以收到。

示例：

```
//监听 主动退群 系统消息
function onQuitGroupNotify(notify) {
    console.info("执行 主动退群 回调： %s", JSON.stringify(notify));
    var reportTypeCh = "[主动退群]";
    var content = "你退出了该群";
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

11.9. 被设为管理员

触发时机：当用户被设置为管理员时，可收到被设置管理员的消息通知，可提示用户。

示例：

```
//监听 被设置为管理员 系统消息
function onSetedGroupAdminNotify(notify) {
    console.info("执行 被设置为管理员 回调： %s", JSON.stringify(notify));
    var reportTypeCh = "[被设置为管理员]";
    var content = "你被群主" + notify.Operator_Account + "设置为管理员";
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

11.10. 被取消管理员

触发时机：当用户被取消管理员时，可收到取消通知，可提示用户。

示例：

```
//监听 被取消管理员 系统消息
function onCanceledGroupAdminNotify(notify) {
    console.info("执行 被取消管理员 回调： %s", JSON.stringify(notify));
    var reportTypeCh = "[被取消管理员]";
    var content = "你被群主" + notify.Operator_Account + "取消了管理员资格";
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

11.11. 群被回收

触发时机：当群组被系统回收时，全员可收到群组被回收消息。

示例：

```
//监听  群被回收  系统消息
function onRevokeGroupNotify(notify) {
    console.info("执行  群被回收  回调：  %s", JSON.stringify(notify));
    //刷新我的群组列表
    getJoinedGroupListHigh(getGroupsCallbackOK);
    var reportTypeCh = "[群被回收]";
    var content = "该群已被回收";
    addGroupSystemMsg(notify.ReportType,  reportTypeCh,  notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

11.12. 用户自定义

触发时机：只有 app 管理员才可以发送自定义系统通知，全员可收到该消息。

发送 API：

```
/* function sendCustomGroupNotify
    *    发送自定义群通知
    * params:
    *    options  -  请求参数，详见 api 文档
    *    cbOk     - function()类型，成功时回调函数
    *    cbErr    - function(err)类型，失败时回调函数, err 为错误对象
    * return:
    *    (无)
    */
    sendCustomGroupNotify: function(options,cbOk, cbErr) {},
```

示例：

demo 增加了发送自定义通知入口。

在 demo 中点击【菜单】->【群组】->【我的群组】，发送自定义群通知如下图：

发送 demo：

```
//发送自定义群系统通知
```

```
var sendCustomGroupNotify = function () {

    var content=$("#sgsm_content").val();
    if (content.length < 1) {
        alert("发送的消息不能为空!");
        return;
    }
    if (webim.Tool.getStrBytes(content) > MaxMsgLen.GROUP) {
        alert("消息长度超出限制(最多" + Math.round(maxLen / 3) + "汉字)");
        return;
    }
    var options = {
        'GroupId': $("#sgsm_group_id").val(),
        'Content': content
    };
    webim.sendCustomGroupNotify(
            options,
            function (resp) {
                $('#send_group_system_msg_dialog').modal('hide');
                alert('发送成功');
            },
            function (err) {
                alert(err.ErrorInfo);
            }
    );
};
```

监听解析 demo：

```
//监听 用户自定义 群系统消息
function onCustomGroupNotify(notify) {
    console.info("执行 用户自定义系统消息 回调： %s", JSON.stringify(notify));
    var reportTypeCh = "[用户自定义系统消息]";
    var content = "收到了自己自定义的系统消息";
    addGroupSystemMsg(notify.ReportType, reportTypeCh, notify.GroupId,
notify.GroupName, content, notify.MsgTime);
}
```

12. 错误码说明

12.1. 收发消息错误码

命令名	错误码	错误描述

syncMsgs	**20001**	无效包
	20002	签名鉴权失败
	20003	无效的号码
	20004	网络异常
	20005	服务异常
	20006	因第三方要求，拦截发送方请求
	80001	安全打击
sendMsg	**20001**	无效包
	20002	签名鉴权失败
	20003	无效的号码
	20004	网络异常
	20005	服务异常
	20006	因第三方要求，拦截发送方请求
	80001	安全打击

12.2. 资料相关错误码

与资料相关的 api 错误码请参考《腾讯云 IM 开放-WebSDK-资料管理-REST 协议文档》

12.3. 关系链相关错误码

与关系链相关的 api 错误码请参考《腾讯云 IM 开放-WebSDK-关系链管理-REST 协议文档》

12.4. 群组相关错误码

与群组相关的 api 错误码请参考《腾讯云 IM 开放-WebSDK-群组管理-REST 协议文档》

12.5. 上传图片错误码

与上传图片相关的 api 错误码请参考《腾讯云 IM 开放-WebSDK-图片管理-REST 协议文档》