ALIBABA CLOUD

阿里云

语音服务

融合通信

文档版本：20220302

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.产品简介 05
1.1. 什么是融合通信 05
1.2. 基本概念 05
2.使用指南 07
2.1. 融合通信使用流程 07
2.2. 服务配置 08
3.服务端开发指南 12
3.1. 服务端API指南 12
3.2. 呼叫回调 15
3.3. 回执消息 19
3.3.1. 话单消息 19
3.3.2. 语音消息回执 22
4.客户端SDK开发指南 24
4.1. Android 24
4.1.1. Android开发 24
4.1.2. SDK使用说明 27
4.2. iOS 38
4.2.1. iOS开发 38
4.2.2. SDK使用说明 40
4.3. Web 45
4.3.1. Web开发 45
4.3.2. SDK接口 50

1.产品简介

1.1. 什么是融合通信

通过本文您可以了解什么是融合通信，以及融合通信的产品架构和使用场景等内容，帮助您更快地了解和使用融合通信。

产品概述

融合通信产品兼具公共电话网接入能力与互联网实时音视频媒体通话能力，提供互联网域与电话网域的双向通信能力，辅以纯互联网域的音视频通信以及纯电话网域的音频通信能力。

产品优势

融合通信产品提供Web、Android、iOS等多平台SDK接入形式，同时提供丰富的通话模式、管理控制、实时录音、组合呼叫等，使接入方不再依赖传统的电话机或通信模组，赋能各类软硬件灵活的通信能力，使任何能接入互联网的设备都能够自由地与电话、手机等通信设备相互接打电话，使“电话”这一重要而硬核的功能不再游离在各类集成硬件或系统之外，并赋予从通话到数据分析沉淀的闭环功能。

主要功能与应用场景

名称	功能描述	应用场景
同振	主叫同时呼叫多个被叫，多个被叫同时响铃。当有一个被叫接听时其他被叫停止响铃。	CRM、工作号
顺振	主叫依次呼叫多个被叫，如果当前被叫超时无人接听，则依次呼叫下一个被叫。当有一个被叫接听时不再往下呼叫。	CRM、工作号
号显	对于VOIP2PSTN通话，主叫呼叫时需传入号显号码、被叫号码两个号码，其中号显需为在数据语音平台已购买的号码，被叫将收到主叫为号显号码的来电。对于PSTN2VOIP通话，电话网中的终端通过呼叫号显号码呼入到数据语音平台，并由数据语音平台路由到对应的VOIP终端。	CRM、出境游、智能硬件
实时录音	用户可调用接口获取通话实时录音。	CRM、工作号

1.2. 基本概念

本文为您介绍了融合通信使用过程中遇到的常用名词的基本概念和简要描述。

名词解释

名称	描述
VOIP	即Voice on IP。在数据语音中，通常将通话互联网接入的终端（Web、iOS、Android等）简称为VOIP端。
PSTN	即Public Switched Telephone Network，公共电话网。在数据语音中，通常将通过公共电话网接入的终端（电话）简称为PSTN端。

名称	描述
VOIP2PSTN	主叫为VOIP、被叫为PSTN的通话类型。即从Web、Android、iOS终端呼叫公共电话网中的一个终端。有时也将这种通话模式简称为呼出。
PSTN2VoIP	主叫为PSTN、被叫为VOIP的通话类型。即从公共电话网呼叫到Web、Android、iOS终端。有时也将这种通话模式简称为呼入。
VOIP2VOIP	主被叫均为VOIP的纯互联网域的通话类型。
PSTN_TRANSFER（呼转或转呼）	主被叫均为PSTN的通话类型。即公共电话网中终端呼叫到数据语音平台后，数据语音平台将通话转呼到公共电话网中的另一终端。
PID	即PartnerId，语音平台合作伙伴ID，通常与阿里云账号一一对应。
CID	即CustomerId，数字中台用户ID，通常与阿里云账号、PID一一对应。
Module	与PID、CID一一对应。
RtcId	通过互联网域接入的终端身份标识，相当于互联网域的“手机号”。
ChannelId	一次通话的唯一ID。
UUID	通话中某个终端的一次会话接入的唯一ID。例如，终端A呼叫终端B，那么整个通话有唯一的ChannelId，并且对于A的这次会话接入有唯一的uuid-A，对于B的这次会话接入也有唯一的uuid-B。

2.使用指南

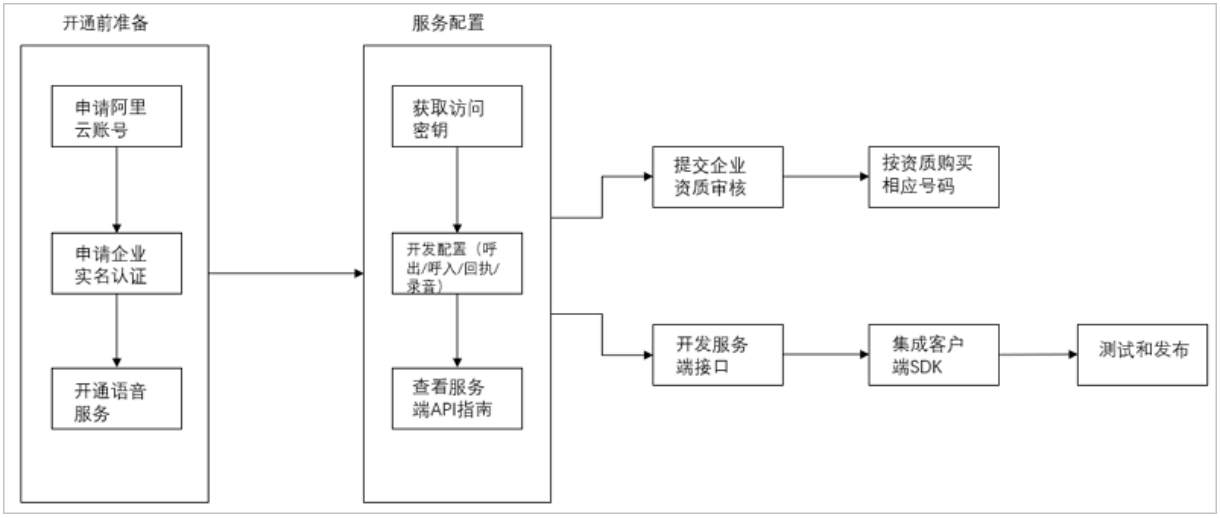
2.1. 融合通信使用流程

您可以通过本文快速了解融合通信功能的使用方式。

前提条件

- 您已经完成注册阿里云账号，并完成企业实名认证。具体操作，请参见和
- 开通语音服务。具体操作，请参见开通服务。
- 获取访问密钥。具体操作，请参见获取AccessKey。

接入流程图



操作流程

1. 开发配置。

   使用融合通信功能，必须完成控制台上相关配置。更多信息，请参见服务配置。

2. （可选）购买号码。

   > 注意
   >
   > - 如果不需要电话功能，则无需购买号码，此步骤可以跳过。另外自有线路和托管模式时也无需购买号码。
   > - 号码需要开通融合通信服务后才能使用融合通信业务。通过SIP打标后购买的号码，会在打标时开通此服务。否则需要补签开通融合通信相关号码服务。您可以在对接群中通知技术支持进行SIP打标，再购买号码。

   如果企业需要使用电话功能，需要完成以下操作。

   i. 提交提交企业资质。企业资质审核标准，请参见企业资质审核标准。

   ii. 购买购买号码。
   购买完成后需通知对接群里技术支持做相应配置、SIP打标、补签开通融合通信相关号码服务、号码呼入配置。

3. 开发接入。

   对接开发流程，更多信息，请参见服务端API指南。

2.2. 服务配置

本文为您介绍了如何在控制台进行服务配置，以方便您能快速使用服务功能。服务配置可提供针对通话发起时的控制，推送状态消息回执、通话后的话单及录音消息回执。

登录控制台

所有配置都需要登录账号后到语音服务控制台配置。请登录语音服务控制台。

呼出回调配置

当呼叫从SDK发起时，融合通信平台根据配置的地址进行回调，对是否允许建立呼叫以及通话的最大时长进行控制。若未配置，默认所有呼叫都允许且不限最大通话时长。

> **说明** 该回调配置尚未提供控制台配置入口，如有需求请联系技术支持进行配置。

呼入回调配置

若购买了语音号码且配置了回调地址，则有用户呼叫该语音号码时，融合通信平台根据配置的地址进行回调，对是否允许建立呼叫以及通话的最大时长进行控制，同时控制：

- 转呼到哪一个rtcID上，相应rtcID若在线，则端侧收到来电。
- 转呼到其他固定电话号码或手机号码上，此场景下，支持转呼到一组号码上，支持顺振或同振该组号码。同振指的是所有配置的号码同时呼叫，任一一个接通后其他呼叫立即停止，顺振指按照顺序呼叫该组号码，直到有一个接听或全部超时未接听。

若未配置，默认所有呼叫都不接通，主叫听到默认放音。

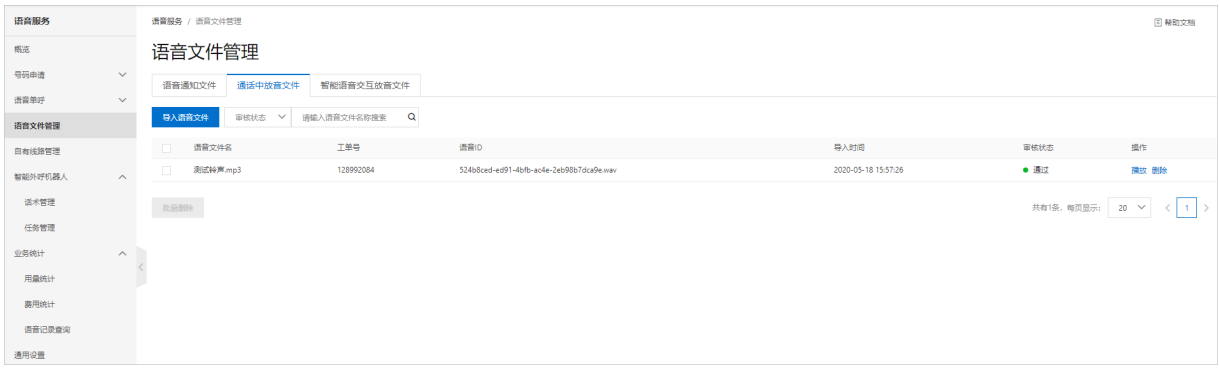
控制台中配置入口及说明如下图所示：

登录语音服务控制台，在控制台选择**通用配置 > 呼入设置 > 融合呼入设置**，配置回调接口以及拨号过程中各种情况下的放音提醒。

> **注意** 呼入回调配置完成后，需要联系客服将呼入通道切换为融合通信通道，这样呼入时才会走回调地址，否则使用的是通用呼入配置。

放音文件上传：

在控制台选择**语音文件管理 > 通话中的语音文件**，单击**导入语音文件**，完成文件上传。



> **注意** 语音文件上传后需要等待审批通过后才能使用。

话单回执消息配置

若开通话单回执消息，通话结束后融合通信平台会推送相应通话话单，采用MNS或HTTP方式，消息内容包含：呼叫ID、开始时间、结束时间、通话时长、主叫ID（或号码）、被叫ID（或号码）等信息。客户可以接收该信息做通话统计和计费统计。

在控制台选择**通用设置 > 订阅回执消息**中下拉找到融合通信相关开关。

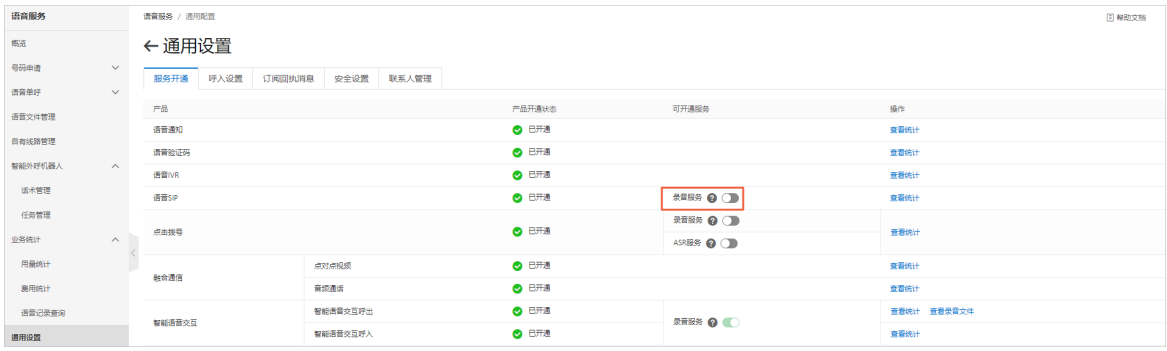
录音开启配置

开启录音后，使用SDK拨打电话可在服务端录音，采用MNS或HTTPS方式，开发者接收到录音回执消息后，可下载录音到本地。

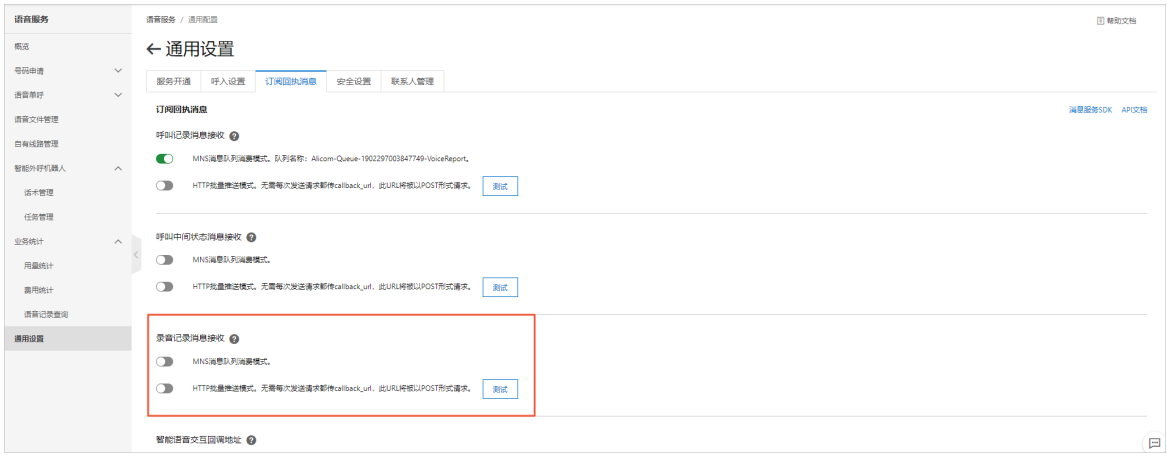
> **说明** 电话呼入及音频通话（网络）暂时不支持录音，录音不支持直接存储到客户OSS上。

控制台配置打开录音功能：

1. 在您购买到的号码上开通语音SIP服务。



2. 控制台配置接收录音回执消息。


语音服务
概览
号码申请
语音单呼
语音文件管理
自有线路管理
智能外呼机器人
话术管理
任务管理
业务统计
用量统计
费用统计
语音记录查询
通用设置
语音服务 / 通用配置
帮助文档
← 通用设置
服务开通
呼入设置
订阅回执消息
安全设置
联系人管理
订阅回执消息
消息服务SDK
API文档
呼叫记录消息接收
MNS消息队列消费模式。队列名称：Alicom-Queue-1902297003847749-VoiceReport。
HTTP批量推送模式。无需每次发送请求即传callback_url，此URL将被以POST形式请求。
测试
呼叫中间状态消息接收
MNS消息队列消费模式。
HTTP批量推送模式。无需每次发送请求即传callback_url，此URL将被以POST形式请求。
测试
录音记录消息接收
MNS消息队列消费模式。
HTTP批量推送模式。无需每次发送请求即传callback_url，此URL将被以POST形式请求。
测试
智能语音交互回调地址

3.服务端开发指南

3.1. 服务端API指南

调用AddRtcAccount会返回一个融合通信账号,这个融合通信账号会归属于您当前的阿里云账号，不会被其他开发者使用。每个开发者可以最多创建500万个融合通信账号，如果不能满足需求可以线下联系您的客户经理。

第一步：在项目中引入对应的依赖。

```
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-core</artifactId>
  <version>3.2.3</version>
</dependency>
```

```
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-dyvmsapi</artifactId>
  <version>1.2.0</version>
</dependency>
```

步骤二：调用AddRtcAccount接口创建账号。

- 请求参数

参数名称	参数类型	是否必选	示例值	描述
deviceId	String	是	abcdefg	这个融合通信账号关联的设备ID，同一个设备ID只能创建一个融合通信账号。

- 返回数据

出参名称	出参类型	示例值	描述
requestId	String	8906582E-6722	请求ID。
code	String	OK	状态码。返回OK代表请求成功，其他错误码详见错误码列表。
message	String	请求成功	状态码的描述。
module	String	200000***00001	融合通信账号ID。

- 示例程序

```
 //设置访问超时时间
System.setProperty("sun.net.client.defaultConnectTimeout", "10000");
System.setProperty("sun.net.client.defaultReadTimeout", "10000");
//融合通信服务产品名称（产品名固定，无需修改）
final String product = "Dyvmsapi";
//融合通信服务产品域名（接口地址固定，无需修改）
final String domain = "dyvmsapi.aliyuncs.com";
//需要替换成开发者的AK信息
//初始化acsClient暂时不支持多region
IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou", accessKeyId, accessKeyS
ecret);
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", product, domain);
IAcsClient acsClient = new DefaultAcsClient(profile);
//组装请求对象-具体描述见控制台-文档部分内容
AddRtcAccountRequest request = new AddRtcAccountRequest();
request.setDeviceId(UUID.randomUUID().toString());
//hint 此处可能会抛出异常，注意catch
AddRtcAccountResponse addRtcAccountResponse = acsClient.getAcsResponse(request);
```

步骤三：调用GetRtcToken获取token。

客户端在使用云通信的融合通信服务时需要token等其他参数，这些参数可以通过GetRtcToken接口从云通信平台获取。在客户端的updateToken回调中，开发者需要实现从云通信平台获取token。为了系统安全，开发者不要直接从客户端调用云通信平台POP接口，建议通过开发者的服务端访问POP接口。

> **注意** 入参userId是步骤二中的出参module。

- 请求参数

参数名称	参数类型	是否必选	示例值	描述
userId	String	是	200000***000001	用户ID。在有账号模式时就是融合通信账号，无账号模式时是用户的自定义账号。
deviceId	String	是	abcdefg	设备ID。建议使用设备的UUID，同个账号，去获取token时，如果设备ID不一样，较早登录的账号会被踢出，无法继续使用融合通信服务。
isCustomAccount	Boolean	是	false	是否是无账号模式。

- 返回数据

出参名称	出参类型	示例值	描述
requestId	String	8906582E-6722	请求ID。
code	String	OK	状态码。返回OK代表请求成功，其他错误码详见错误码列表。
message	String	请求成功	状态码的描述。
module	String	{ “cleansession” :true, ” clientId” :” GID_VOIP@@@ClientId_2000000000000009_100648480015” ,” conferenceTopic” :” cs_alicom_voip_conference” ,” host” :” mqtt-cn-4590mdhb901.mqtt.aliyuncs.com” ,” meetingEventKeepAliveInterval” :0,” phoneTopic” :” alicom_voip_phone” ,” port” :0,” reconnectTimeout” :2000,” registerTime” :0,” sdkClientPort” :8883,” serverId” :” GID_VOIP@@@MTEuMTMuMTM2LjExOA==” ,” sgwServerTopic” :” alicom_voip_server_pre” ,” tlsport” :443 ,” tokenData” :” abcdef” ,” useTLS” :false}	token对象，整个JSON都需要返回给客户端并且传给融合通信SDK。

- 示例程序

```
//设置访问超时时间
System.setProperty("sun.net.client.defaultConnectTimeout", "10000");
System.setProperty("sun.net.client.defaultReadTimeout", "10000");
//融合通信服务产品名称（产品名固定，无需修改）
final String product = "Dyvmsapi";
//融合通信服务产品域名（接口地址固定，无需修改）
final String domain = "dyvmsapi.aliyuncs.com";
//需要替换成开发者的AK信息
//初始化acsClient暂时不支持多region
IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou", accessKeyId, accessKeyS
ecret);
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", product, domain);
IAcsClient acsClient = new DefaultAcsClient(profile);
//组装请求对象-具体描述见控制台-文档部分内容
GetRtcTokenRequest request = new GetRtcTokenRequest();
request.setUserId("2000000000000009");
request.setDeviceId("abcdefg");
request.setIsCustomAccount(false);
//hint 此处可能会抛出异常，注意catch
GetRtcTokenResponse getRtcTokenResponse = acsClient.getAcsResponse(request);
```

错误码

Code	描述
OK	请求成功
isp.RAM_PERMISSION_DENY	RAM权限DENY
isv.OUT_OF_SERVICE	业务停机
isv.PRODUCT_UNSUBSCRIBE	产品未开通
isv.ACCOUNT_NOT_EXISTS	账户不存在
isv.ACCOUNT_ABNORMAL	账户异常
isv.INVALID_PARAMETERS	参数异常
isp.SYSTEM_ERROR	系统错误

3.2. 呼叫回调

当您使用融合通信服务时，在呼叫的某些环节会回调开发者的HTTP接口，用业务鉴权和呼叫控制。所有HTTp接口都是使用POST的方式，返回值是一个JSON对象。

呼出操作鉴权HTTP回调

如果您在我们平台配置了呼出操作鉴权HTTP回调地址，我们在隶属于您的融合通信账号发生呼出操作时，会通过HTTP接口回调您的服务端接口，并依据返回的结果决定是否允许当前操作继续发生。如果您没有配置对应的回调接口，我们默认这个操作是允许的。

- 请求参数。

参数名称	参数类型	必填与否	样例取值	参数说明
communicationType	String	必须	voip2voip	呼叫类型。取值： ○ voip2voip ○ voip2pstn
callerInfo	Object	必须	{“voipId”：“20000000000****”}	主叫信息。
calleeInfos	Array	必须	[{“isPstn”：“true”，“platformNumber”：“1381234****”，” phoneNumber”：“1371234****”}]	被叫信息。
channelId	String	必须	12323123123	本次通话的唯一标识。
extend	String	必须	abc	SDK发起呼叫时传入的扩展字段。

- callerInfo结构体。

参数名称	参数类型	必填与否	样例取值	参数说明
voipId	String	必须	2000000000****	主叫数据语音账号。

- calleeInfo结构体

参数名称	参数类型	必填与否	样例取值	参数说明
isPstn	Boolean	必须	true	是否为pstn。
platformNumber	String	非必须	05710000****	isPstn为true时有该字段，呼出到pstn时使用的主叫电话号码，即被叫看到的来电号码。
phoneNumber	String	非必须	1380000****	isPstn为true时有该字段，被叫电话号码。
selfLineName	String	非必须	123	isPstn为true且为自有线路用户时有该字段，自有线路名称。
voipId	String	非必须	200000000****	isPstn为false时有该字段，被叫数据语音账号。

- 返回数据。

出参名称	出参类型	样例取值	参数说明
success	Boolean	true	是否允许本次呼叫，true表示允许。
message	String	呼叫受限	呼叫不允许时透传给客户端的错误信息，可选。（即将支持）
model	Object	{“timeLimit”：200}	允许呼叫时的一些附加参数，可选。

model结构体。

出参名称	出参类型	样例取值	参数说明
timeLimit	Long	100	本次通话可允许的最大通话时长，单位为秒，可选。
needRecord	Boolean	true	本次通话是否需要服务端录音，会覆盖SDK调用相关接口设置的值。

呼入回调接口

如果您在我们平台配置了呼入HTTP回调地址，我们在隶属于您的呼叫送达到我们平台时会调用您的这个HTTP接口获取您希望呼叫的融合通信账号列表（也就是被叫）。

- 请求参数。

参数名称	参数类型	必填与否	样例取值	参数说明
phoneNumber	String	必须	1876713****	主叫号码。
platformNumber	String	必须	1876713****	被叫号码。
channelId	String	必须	12323123123	本次通话的唯一标识。

- 返回参数。

出参名称	出参类型	样例取值	参数说明
success	Boolean	true	是否允许当前操作，为true表示允许。
model	Object	{ “calleeInfos” : [{ “rtcId” :200000000000****}]}	被叫信息列表。

  - model结构体

出参名称	出参类型	样例取值	参数说明
mixedCalleeInfos	Array	[{"type": "voip", "voipId": "200000000000****"}, {"type": "pstn", "platformNumber": "05718684****", "phoneNumber": "1380000****", "sequence": 2, "selfLineName": "qwe"}]	被叫信息。
timeLimit	Long	100	本次通话可允许的最大通话时长，单位为秒，可选。
needRecord	Boolean	true	（即将支持）本次通话是否需要服务端录音，会覆盖SDK调用相关接口设置的值。

- mixedCalleeInfo结构体

出参名称	出参类型	样例取值	参数说明
type	String	voip	枚举值，voip表示呼入到voip端，pstn表示转呼至pstn端。
rtcId	String	2000000000****	type为voip时需要该字段，被叫数据语音账号。
phoneNumber	String	1876713****	type为pstn时需要该字段，转呼时的被叫电话号码。
platformNumber	String	05710000****	type为pstn时需要该字段，转呼时使用的主叫电话号码，需要为已在平台购买且开通数据语音服务的固话号码。
sequence	String	1	呼叫次序。

注意

对于sequence字段，数据语音服务从sequence为1开始逐批振铃呼叫：

- 即首先同时呼叫sequence为1的所有被叫，待所有被叫都挂断后，再同时呼叫sequence为2的所有被叫，依次类推。
- 一旦有任何一个被叫接听，则认为通话已建立，中断其他已进行或正在进行的呼叫，若没有下一序列被叫，则中止呼叫。
- sequence可以重复表示同时呼叫，但必为从1开始连续的数字，无该字段是默认为1，如1-2-2-3-5，则会依次呼叫到5，sequence最大值为10，相同sequence最多也为10个。
- 目前不支持所有被叫中包含多个voip被叫，不支持相同sequence下同时包含voip与pstn被叫，也不支持相同sequence中既包含voip被叫又包含pstn被叫，若发生这样的现象，则将会随机取一个被叫。

3.3. 回执消息

3.3.1. 话单消息

当您使用融合通信服务时，可以通过使用MNS的Queue模型来接收通话的回执消息。

消息的订阅

云通信的所有业务消息都会通过MNS消息服务向外发送。目前融合通信服务支持的消息类型有：ArtcCdrReport（呼叫话单消息）。

注意 以上消息类型需要**消息队列名称**，获取名称请联系您的客户经理。

请按照以下步骤完成消息接收：

1. 替换您自己的AK与SK。
2. 设置相应的消息类型与消息队列名称。
3. 启动应用开始接收消息。

> **说明** 不同类型的消息返回的消息体内包含的字段是不一样的，您需要依据订阅的消息类型做适当修改。

消息类型

呼叫话单消息

消息类型为：ArtcCdrReport，您可以通过订阅这个消息获取呼叫的话单信息，包括通话时长，通话发起时间，结束时间等。

点对点消息体格式

名称	类型	示例值	描述
detail	struct	-	话单消息明细
callInInfo	struct	-	呼入信息
callOutInfo	struct	-	呼出信息
channelId	String	amdc7a6a9ds90sad0a	一次通话过程唯一Id
startTime	String	2017-06-01 10:00:00	通话开始时间，未接通没有开始时间
endTime	String	2017-06-01 10:01:00	通话结束时间，未接通则为空
duration	String	60	通话时长，未接通为0
caller	String	20000000000000001	主叫用户ID
callee	String	20000000000000002	被叫用户ID
alicomRtcType	String	voip2voip	通话类型，当前类型有：voip2voip，video2video等
extend	String	123456	扩展字段

示例代码

```
{
  "detail":{
    "callInInfo":{ #呼入信息
      "caller":"20000000000000001", #呼入主叫
      "callee":"20000000000000002", # 被叫
      "startTime":"2017-06-01 10:00:00",#开始时间
      "endTime":"2017-06-01 10:01:00", #结束时间
      "duration":"60" #呼叫持续时间
    },
    "callOutInfo":[
      { #呼出信息
      "caller":"20000000000000001", #呼出主叫
      "callee":"20000000000000002", # 呼出被叫
      "startTime":"2017-06-01 10:00:00",#开始时间（未接通没有开始时间）
      "endTime":"2017-06-01 10:01:00", #结束时间
      "duration":"60" #呼叫持续时间
    },
    {
      "caller":"20000000000000001", #呼出主叫
      "callee":"20000000000000002", # 被叫
      "startTime":"2017-06-01 10:00:00",#开始时间，（未接通没有开始时间）
      "endTime":"2017-06-01 10:01:00", #结束时间
      "duration":"60" #呼叫持续时间
    }
    ]
  }
}
```

集成步骤

下载消息SDK

下载融合通信对应的消息DEMO工程,工程所依赖的jar包都放在lib目录下，将对应的jar包引入到您的工程中即可编写接收消息的程序。

SDK&DEMO下载地址。

```
/**
 * 只能用于接收云通信的消息，不能用于接收其他业务的消息
 */
public class ReceiveAlicomMsgDemo {
    private static Log logger=LogFactory.getLog(ReceiveAlicomMsgDemo.class);
    static class MyMessageListener implements MessageListener{
        private Gson gson=new Gson();
        @Override
        public boolean dealMessage(Message message) {
            System.out.println("message handle: " + message.getReceiptHandle());
            System.out.println("message body: " + message.getMessageBodyAsString());
            System.out.println("message id: " + message.getMessageId());
            System.out.println("message dequeue count:" + message.getDequeueCount());
            try{
                Map<String,Object> contentMap=gson.fromJson(message.getMessageBodyAsString(
), HashMap.class);
                //依据自己的消息类型，获取对应的字段
                String uuid=(String)contentMap.get("uuid");
                //TODO 这里开始写业务代码
            }catch(com.google.gson.JsonSyntaxException e){
               logger.error("error_json_format:"+message.getMessageBodyAsString(),e);
            }
            Boolean dealResult=true;
            return dealResult;//返回true，则工具类自动删除已拉取的消息。
        }
    }
    public static void main(String[] args) throws com.aliyuncs.exceptions.ClientException,
ParseException {
        DefaultAlicomMessagePuller puller=new DefaultAlicomMessagePuller();
        String accessKeyId="yourAccessKeyId";
        String accessKeySecret="yourAccessKeySecret";
        String messageType="ArtcCdrReport"; //注意替换成您需要获取的消息类型
        String queueName="yourQueueName";//联系您的客户经理获取queueName
        puller.startReceiveMsg(accessKeyId,accessKeySecret ,messageType,queueName, new MyMe
ssageListener());
    }
}
```

3.3.2. 语音消息回执

当您使用语音服务的API接口发送外呼后，可以通过MNS消息队列消费模式来接收语音服务的回执消息。

回执消息模式和配置流程，请参见回执消息简介与配置流程。

消息类型

融合通信提供的消息类型为呼叫录音消息（VoiceRecordReport）。

示例代码

```
/**
 * 只能用于接收云通信的消息，不能用于接收其他业务的消息
 */
public class ReceiveAlicomMsgDemo {
    private static Log logger=LogFactory.getLog(ReceiveAlicomMsgDemo.class);
    static class MyMessageListener implements MessageListener{
        private Gson gson=new Gson();
        @Override
        public boolean dealMessage(Message message) {
            System.out.println("message handle: " + message.getReceiptHandle());
            System.out.println("message body: " + message.getMessageBodyAsString());
            System.out.println("message id: " + message.getMessageId());
            System.out.println("message dequeue count:" + message.getDequeueCount());
            try{
                Map<String,Object> contentMap=gson.fromJson(message.getMessageBodyAsString(
), HashMap.class);
                //依据自己的消息类型，获取对应的字段
                String callId=(String)contentMap.get("call_id");
                String startTime=(String)contentMap.get("start_time");
                String endTime=(String)contentMap.get("end_time");
                String duration=(String)contentMap.get("duration");
                String statusCode=(String)contentMap.get("status_code");
                String statusMsg=(String)contentMap.get("status_msg");
                String outId=(String)contentMap.get("out_id");
                String dtmf=(String)contentMap.get("dtmf");
                //TODO 这里开始写业务代码
            }catch(com.google.gson.JsonSyntaxException e){
               logger.error("error_json_format:"+message.getMessageBodyAsString(),e);
            }
            Boolean dealResult=true;
            return dealResult;//返回true，则工具类自动删除已拉取的消息。
        }
    }
    public static void main(String[] args) throws com.aliyuncs.exceptions.ClientException,
ParseException {
        DefaultAlicomMessagePuller puller=new DefaultAlicomMessagePuller();
        String accessKeyId="yourAccessKeyId";
        String accessKeySecret="yourAccessKeySecret";
        String messageType="VoiceReport"; //注意替换成你自己需要获取的消息的类型
        String queueName="yourQueueName";//在云通信页面开通相应业务消息后，就能在页面上获得对应的q
ueueName
        puller.startReceiveMsg(accessKeyId,accessKeySecret ,messageType,queueName, new MyMe
ssageListener());
    }
}
```

4.客户端SDK开发指南

4.1. Android

4.1.1. Android开发

客户可通过接入Android SDK快速实现Android端上的音频通话及视频通话等功能，依托阿里云的研发资源，我们为客户提供安全、可靠、弹性、低成本的实时通信服务。

功能特性

提供VoIP to PSTN，PSTN to VoIP, VoIP to VoIP通话场景的服务，接入方可根据需求选择对应的应用场景。

开发准备

1. 已完成创建阿里云账号并完成企业实名认证。
2. 获取阿里云访问密钥。具体操作，请参见获取AccessKey。
3. 购买语音号码。具体操作，请参见购买号码。

SDK下载地址

Android端SDK下载地址，请参见Android数据语音2.1.0。

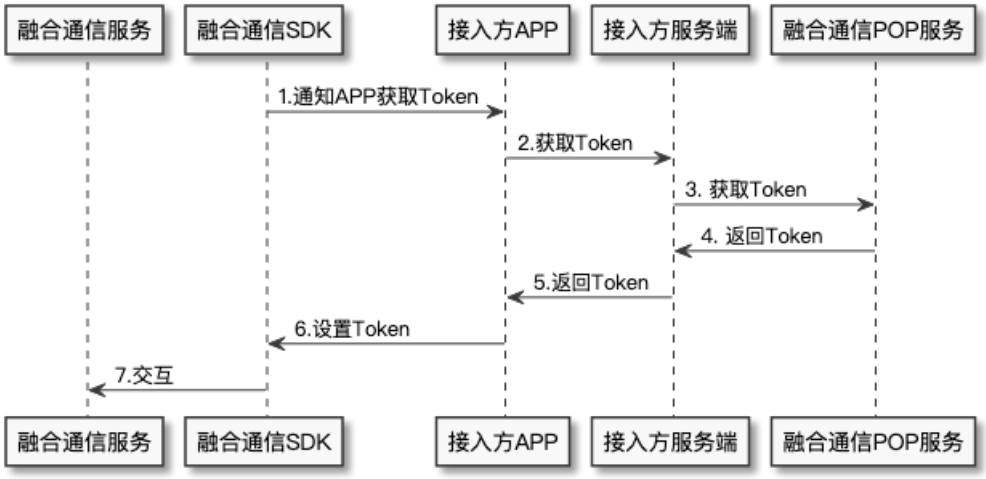
接入前准备

1．获取Token

为了保证安全性，融合通信SDK通过服务端下发的临时token作为身份标识与服务进行交互。临时token可通过融合通信服务POP接口从阿里云获得，更多详情请参见服务端API指南。

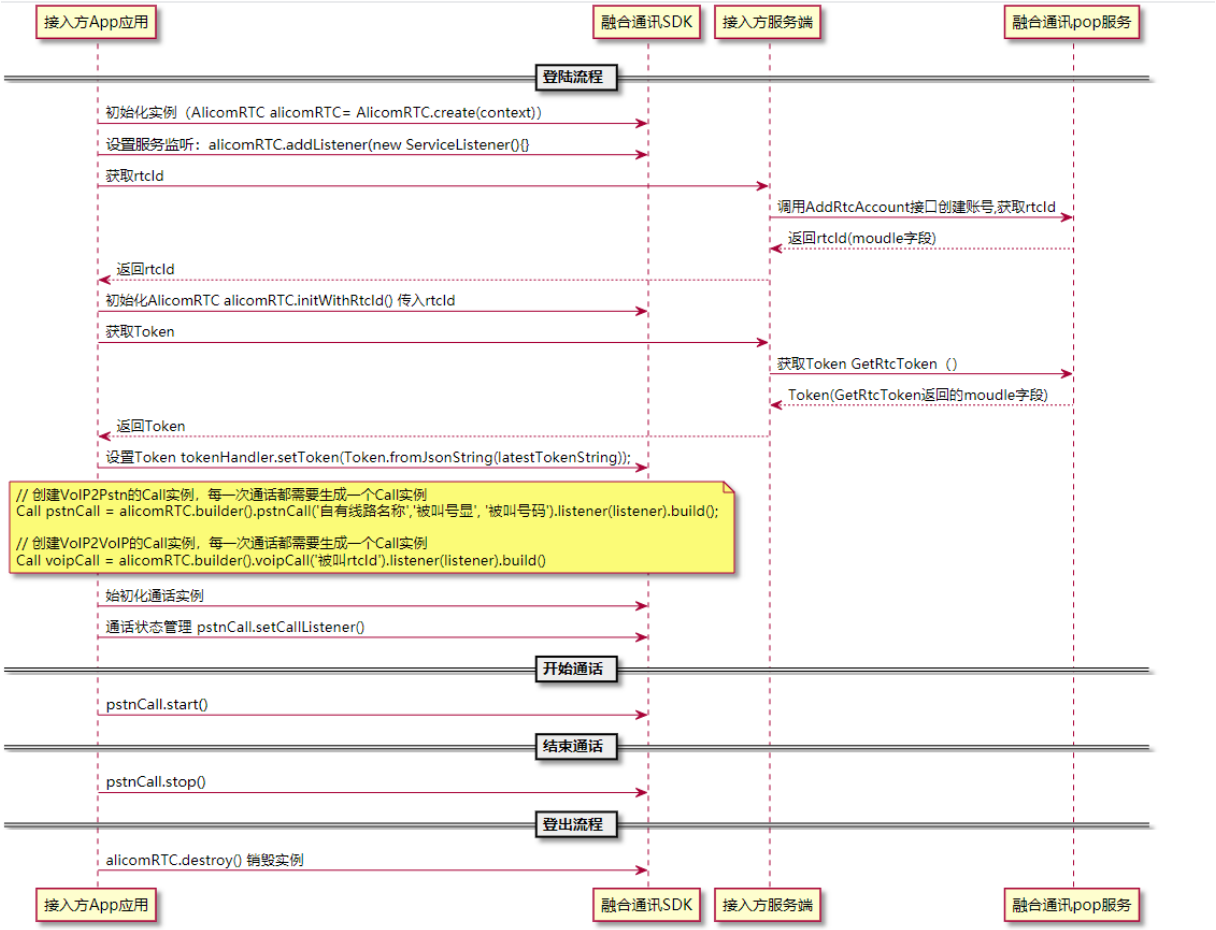
> **说明** SDK在token即将失效或其他需求而需要更新token时，会通过回调通知接入方，App需自行实现接口获取最新token并传递给SDK。

推荐的交互流程如下：



2．确认版本

融合通信SDK分为音频版本与视频版本，视频版本支持所有功能，音频版本仅支持音频通话，可以通过版本号区分当前版本是否支持视频（音频版本号以.audio结尾），也可以调用API判断。



设备以及系统

设备要求：搭载 Android 系统的设备

系统要求：语音版Android 4.0.2(API 14) 及其以上，视频版Android 4.1、4.1.1

开发工具要求：Android studio

开发环境搭建

1．导入SDK

- gradle导入（暂未对外开放）。

  在项目目录下的build.gradle文件里面的allprojects里面加入Maven仓库。

```
allprojects {
repositories {
      ...
      maven { url "http://mvnrepo.alibaba-inc.com/mvn/repository" }
}
```

  配置依赖

  gradle 3.4及以上版本

```
implementation('com.alicom.rtc:alicomRTC:latest.release@aar'){
     transitive true
 }
or
 api('com.alicom.rtc:alicomRTC:latest.release@aar'){
     transitive true
 }
```

gradle 3.4以下版本

```
compile('com.alicom.rtc:alicomRTC:latest.release@aar'){
     transitive true
 }
```

- 手工导入

  请先下载融合通信SDK，解压后放至项目目录，然后根据项目打包环境，自行引入本地依赖。gradle配置参考：

  ```
  implementation fileTree(dir: 'libs', include: ['*.jar', '*.aar'])
  ```

  在项目目录下的build.gradle文件里面的allprojects里面加入Maven仓库。

  ```
  allprojects {
  repositories {
        ...
       maven { url "http://maven.aliyun.com/nexus/content/groups/public/" }
  }
  ```

  如若使用本地依赖，请确保项目中已引入以下依赖：

  ```
  implementation 'com.alibaba:fastjson:1.2.48'
  implementation 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.1.1'
  //如果启用crash组件，增加SDK稳定性请增加以下依赖
  implementation 'com.ucweb.wpk:crashsdk:3.2.0.1@aar'
  ```

  1.7.0版本SDK加入了crash组件，用来保障增加SDK稳定性。如果启用crash组件还需将demo工程lib目录下的crashshield-release.aar拷贝到自己的lib目录下并完成依赖implementation fileTree(dir: 'libs', include: ['*.jar', '*.aar'])

  > 注意 如若接入方项目构建中因引入其他依赖包从而导致出现类重复或冲突，请联系业务经理。

配置so架构

单架构配置

目前SDK中暂只包含armeabi架构so库，故在打包时请在build.gradle加入以下配置：

```
android{
   ......
defaultConfig {
   ndk {
       abiFilters 'armeabi'
  }
}
}
```

多架构配置

1.修改build.gradle文件中artc版本

由原来的 implementation 'com.taobao.android:artc_engine:1.9.145.0@aar' 改为 implementation 'com.taobao.android:artc_engine:3.0.1.12@aar'，或者手工导入最新的3.0.1.12版本的包。

2.修改build.gradle文件中ndk的配置

```
ndk {
            abiFilters  "armeabi","armeabi-v7a", "arm64-v8a"
  }
```

> **说明** 若需要其他架构so库，请联系业务经理。

混淆配置

主线版本混淆配置：

```
-keep class * implements java.io.Serializable{*;}
-keep class org.artc.webrtc.**{*;}
-keep class org.eclipse.paho.client.mqttv3.**{*;}
-keep class com.taobao.artc.internal.**{*;}
-keep class com.taobao.artc.**{*;}
-keepclasseswithmembernames class * {
   native <methods>;
}
#crash组件
-keep class com.uc.crashsdk.** { *; }
-keep interface com.uc.crashsdk.** { *; }
```

权限配置

```
<uses-permission android:name="android.permission.INTERNET"/>
<!--视频版本必须-->
<uses-permission android:name="android.permission.CAMERA"/>
```

4.1.2. SDK使用说明

本文为您介绍了在Android开发时，SDK的使用说明及示例。

创建实例

```
//context建议传入Application，设置成单例对象
AlicomRTC alicomRTC = AlicomRTC.create(context);
```

> 注意 AlicomRTC实例一旦需要重新创建新的实例请先destroy旧的实例再创建新的防止出现业务混乱。

参数描述

参数	类型	说明
context	Context	传ApplicationContext对象

设置参数与回调（可选）

```
//添加一个监听服务状态的回调
alicomRTC.addListener(new ServiceListener() {
        //服务连接成功，当前处于可用状态
        @Override
        public void onServiceAvailable() {}
        //服务连接失败、或者连接断开了，正在销毁资源，当前处于不可操作
        @Override
        public void onServiceUnavailable(int errCode, String errMsg) {}
        //服务已销毁完成，当前处于未连接状态
        @Override
        public void onServiceIdle(){}
        //收到点对点音频来电
        @Override
        public void onReceivingAudioCall(Call call){}
        //收到点对点视频频来电
        @Override
        public void onReceivingVideoCall(VideoCall videoCall){}

    });
//可选，设置本地默认的视频采集分辨率
alicomRTC.setDefaultCaptureConfig(CaptureConfig.FPS15_360P);
//可选，设置默认的呼叫超时时间
alicomRTC.setDefaultCallTimeout(60);
```

参数描述

参数	类型	说明
serviceListener	ServiceListener	传监听服务状态的回调

连接服务

```
/*
 *连接服务,有账号模式需调用initWithRtcId传入融合通信账号,
 *无账号模式则调用initWithCustomId传入自定义账号,
 *以及TokenUpdater的实现用于获取token
 *(Token的具体获取过程由接入方自己实现,
 *由接入方app从接入方服务端获取,
 *而服务端通过pop服务从云通信获取)。
 *SDK每过一段时间就会请求新的token,
 *接入方需要实现方法实时获取id对应最新token。
 *连接成功时会在第2步添加的ServiceListener中回调。
 */
TokenUpdater tokenUpdater =  new TokenUpdater() {
    @Override
    public void updateToken (TokenHandler tokenHandler){
        //get latest token string from network, then
        tokenHandler.setToken(Token.fromJsonString(latestTokenString));
    }
};
//调用此方法进行初始化连接
alicomRTC.initWithRtcId(RtcId, tokenUpdater);
```

参数描述

参数	类型	说明
rtcId	String	RtcId，接入方App应用>接入方服务端：获取rtcId接入方服务端>融合通讯pop服务：调用AddRtcAccount接口创建账号，获取rtcId。
tokenUpdater	TokenUpdater	更新token回调，获取tokenHandler来更新token，用Token.fromJsonString函数来将字符串格式的token转成Token对象。

> **注意** tokenUpdater回调函数的实现，更新的token设置一定要保证是token最新的有效token，尽量避免获取token超时的情况。

开始通话

当ServiceListener中的onServiceAvailable回调被唤起时，表示服务已经连接成功，此时可以开始发起各类通话：

发起voip2voip

```
//Call代表了一通具体的通话
alicomRTC.builder().voipCall(targetRtcId).listener(listener).build().start();
```

参数描述

参数	类型	说明
targetUserId	String	对方的RtcId

发起voip2pstn呼叫

自有线路

```
//customLineName为自有线路名称,targetShowNumber为被叫方显示的来电号码，需要从云通信购买可用号码
alicomRTC.builder().pstnCall(customLineName,targetShowNumber, targetPhoneNumber).listener(listener).build().start();
```

参数描述

参数	类型	说明
customLineName	String	自有线路名称
targetShowNumber	String	被叫号显，被叫方显示的来电号码
targetPhoneNumber	String	被叫电话号码

非自有线路

```
//targetShowNumber为被叫方显示的来电号码，需要从云通信购买可用号码
alicomRTC.builder().pstnCall(customLineName,targetShowNumber, targetPhoneNumber).listener(listener).build().start();
```

参数描述

参数	类型	说明
targetShowNumber	String	被叫号显，被叫方显示的来电号码
targetPhoneNumber	String	被叫电话号码

接听电话

```
1.//在ServiceListener中
public void onReceivingAudioCall(Call call) {
    call.start();
}
```

挂断/拒绝电话

```
call.stop();
```

销毁服务

```
alicomRTC.destroy();
```

本地静音，使自己静音，使对方无法听到自己的声音，别人不会知道，与Participant中的mute属性无关

```
call.muteLocalAudiocall();
```

使自己取消本地静音状态

```
call.unmuteLocalAudio();
```

打开或关闭扬声器

```
call.speakerOn(boolean on);
```

参数描述

参数	类型	说明
on	boolean	true表示打开扬声器，false表示关闭扬声器

开启pstn电话服务端录音

```
pstnCallBuilder.serverRecordEnabled(true)
```

参数描述

参数	类型	说明
on	boolean	true表示开启服务端录音，false表示不开启服务端录音

判断当前是否是本地静音状态

```
call.isLocalAudioMuted()
```

获取当前通话设置中扬声器是否打开

```
call.isSpeakerOn()
```

设置通话CallListener监听

```
call.setCallListener(call.setCallListener(new CallListener() {
            //响铃中
            @Override
            public void onCalleeRinging(Talk talk) {
            }
            //连接中
            @Override
            public void onCalleeConnecting(Talk talk) {
            }
            //已接通
            @Override
```

```
            public void onActive(Talk talk) {
            }
            @Override
            public void onDtmfData(String s, long l, Talk talk) {
            }
            @Override
            public void onConnected(Talk talk) {
            }
            //挂断中
            @Override
            public void onStoppping(int i, String s, Talk talk) {
            }
            //已挂断
            @Override
            public void onStopped(int i, String s, Talk talk) {
            }
            @Override
            public void onNetworkQuality(int i, Talk talk) {
            }
            @Override
            public void onMediaStatistics(MonitorStats monitorStats, Talk talk) {
            }
        });new CallListener() {
            //响铃中
            @Override
            public void onCalleeRinging(Talk talk) {
            }
            //连接中
            @Override
            public void onCalleeConnecting(Talk talk) {
            }
            //已接通
            @Override
            public void onActive(Talk talk) {
            }
            @Override
            public void onDtmfData(String s, long l, Talk talk) {
            }
            @Override
            public void onConnected(Talk talk) {
            }
            //挂断中
            @Override
            public void onStoppping(int i, String s, Talk talk) {
            }
            //已挂断
            @Override
            public void onStopped(int i, String s, Talk talk) {
            }
            @Override
            public void onNetworkQuality(int i, Talk talk) {
            }
            @Override
            public void onMediaStatistics(MonitorStats monitorStats, Talk talk) {
```

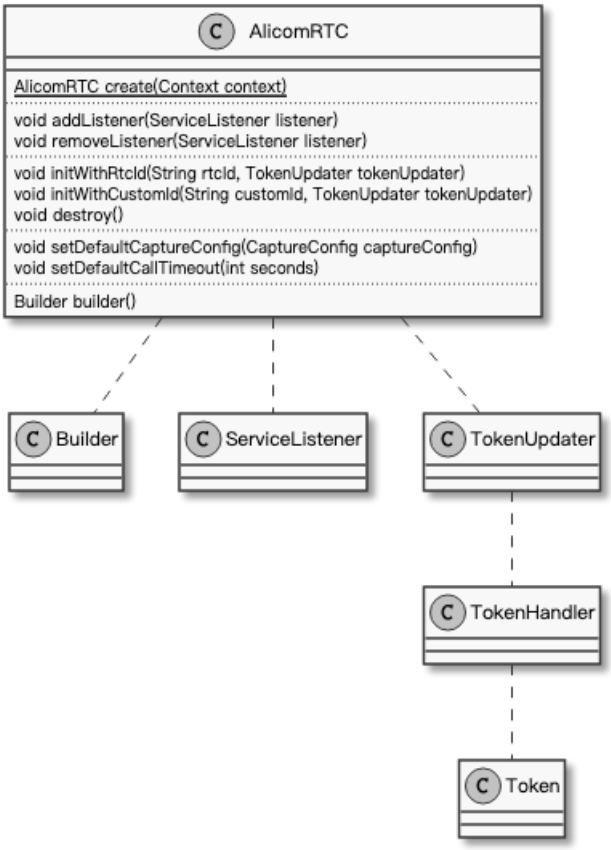
```
            }
        });
```

> **注意** 注意安卓9.0系统对App退后台的麦克风做了限制，为防止通话的时候程序退后台引起的通话被静音问题，请在App退后台情况下发送前台通知来防止通话被静音。

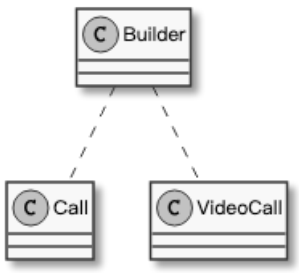
API概述

入口

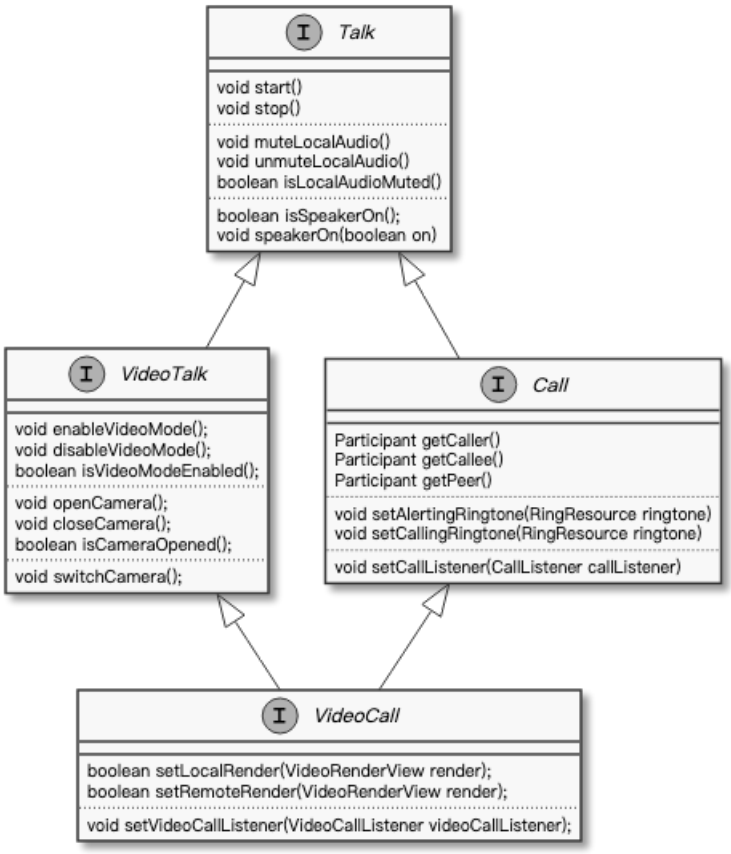
- 类AlicomRTC为API入口，包含初始化服务、添加生命周期回调、销毁服务、设置全局默认属性、构建通话入口以及一些全局公共方法。
- 类TokenUpdater是接入方需要实现的获取最新token的方法，SDK在需要时会唤起该回调获取最新token，接入方在获取到最新token串后可以调用Token.fromJsonString()方法快捷的获取实例并调用TokenHandler.setToken()方法设置token。



- 调用AlicomRTC.builder方法可以获得构建器用于创建各类通话，并可以设置部分参数。

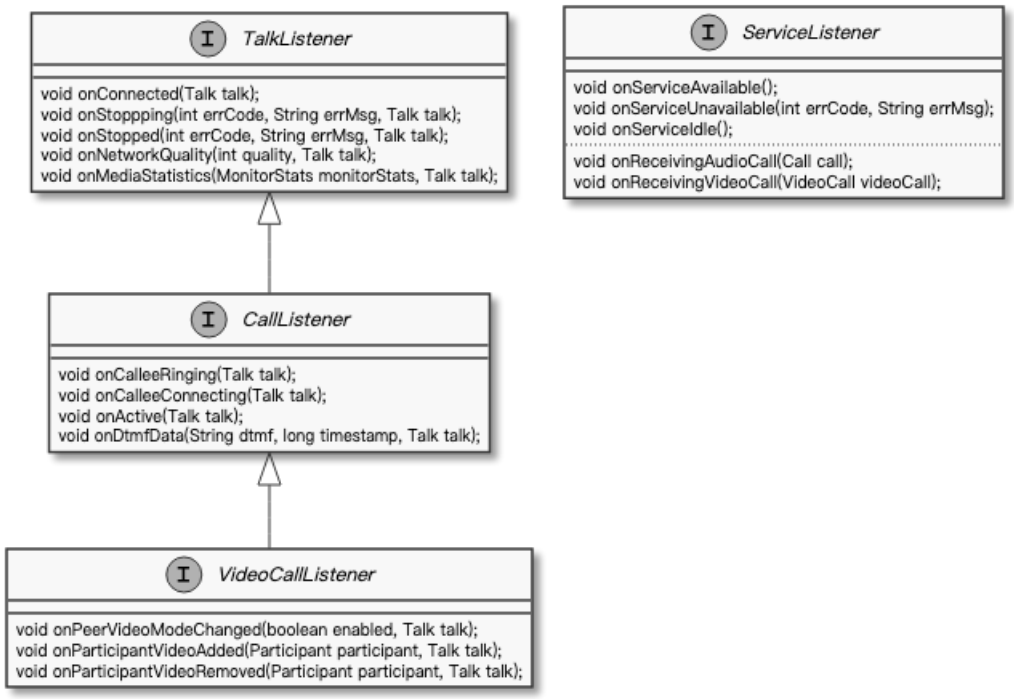


- 通信
  - 融合通信服务SDK将不同通话功能与类型分拆到了相应的类。比如VideoCall就表示视频点对点通话，包含了视频点对点通话中所有的操作方法，每一个该类的实例表示具体的一次通话，不可重复使用。而基类Talk与VideoTalk分别定义了音频与视频通话的公共方法，没有对应的具体实现

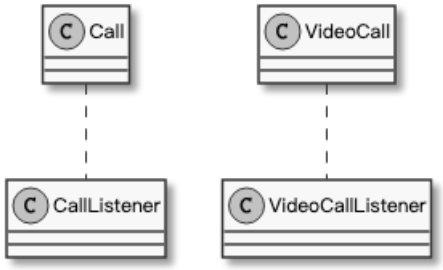


- 监听

- 接口TalkListener中定义了所有公共回调，具体类型的监听回调接口都继承与该接口。



- 每一个具体的通话类型都对应一个事件监听回调，接入方可以在构建通话时设置监听，也可以调用相应set方法设置监听。



其他API参见：API Reference。

错误码

错误码	含义
1000100	MQTT网络连接失败
1000101	消息发送失败
1000102/1000103/1000104	TOKEN无效/TOKEN获取失败/TOKEN上传失败（统归TOKEN更新失败）
1000105	MQTT连接断开
1000106	MQTT订阅topic失败
2000000	本地主动挂断

错误码	含义
2000050	当前版本不支持视频
2000099	对端主动挂断
2000100	本端主叫时，不要拨打自己
2000101	本端主叫时，拨打的号码为空
2000102	连接媒体服务器超时
2000103	本端被叫时，本地超时未响应，于是本地主动拒绝
2000104	本端主叫时，被叫不在线
2000105	本端被叫时，接听发生错误
2000106	本端主叫时，被叫拒绝
2000107	本端被叫时，主叫取消
2000108	本端主叫时，被叫无响应
2000109	本端主叫时，被叫正在通话中
2000110	本端主叫时，被叫版本过低
2000111/2000112	roomId为空/被踢下线（多方通话的错误码）
2000113	当前服务处于不可用状态
2000116	本地主动销毁服务，以及导致的正在进行的通话中断
2000117	状态不对，当前有别的通话正在进行中，无法进行新的通话
2000118	被叫太多
2000120	自定义消息为空或长度超过1024
2000121	网络异常
2000122	对方接收自定义消息失败
2000130	没有录音权限
2000131	没有相机权限
2000132	主叫呼叫多个被叫时，对于被叫，该通呼叫已被其他设备接听
2000133	主叫呼叫多个被叫时，对于被叫，该通呼叫已被其他设备拒绝

错误码	含义
2000134	本地主动销毁服务
2000135	没有麦克风设备
2000136	浏览器不允许使用麦克风
2000137	系统禁用麦克风或者麦克风被占用
3110000	当前账号在别处登录，被踢了，以及导致正在进行的通话中断
3110001	服务异常
3999999	服务器系统异常
3100000	被叫未注册/未登录
3100001	主叫通话异常
3100002	参数不正确
3100003	TOKEN验证失败
3100004	账号错误
3100005	业务停机
3100006	无效号显
3100007	无权操作
3100009	未订购对应产品
3100018/3100019	多方通话的错误码
3100020	被叫不合法
3100021	被叫连接媒体失败
3000000-3300000间别的错误码	服务端其他错误
3900000-4200000	媒体连接失败
8000000	未知错误

常见问题

1. 事件回调会在什么线程回调呢？

   请注意，SDK中所有的Listener回调都会在主线程回调到接入方。

2. 每次SDK回调需要更新Token时都要向服务端获取最新的Token吗？

   是的。

3. 可以实例化多个AlicomRTC吗？

   建议将AlicomRTC作为单例使用，否则多个实例间会造成Token互斥以及底层资源冲突。

4.2. iOS

4.2.1. iOS开发

客户可通过接入iOS SDK快速实现iOS端上的音频通话及视频通话等功能，依托阿里云的研发资源，我们为客户提供安全、可靠、弹性、低成本的实时通信服务。

功能特性

提供VoIP to PSTN，PSTN to VoIP, VoIP to VoIP通话场景的服务，接入方可根据需求选择对应的应用场景。

前提条件

- 客户端需要从阿里云申请融合通信账号后才能接入。您可以使用AddRtcAccount创建账号。
- 通过融GetRtcToken临时token，作为身份标识与服务进行交互，保证安全性。

  > **说明** SDK在token即将失效或其他需求而需要更新token时，会通过回调通知接入方，接入方需自行实现接口获取最新token并传递给SDK。

- 设备以及系统：
  - 支持所有iOS设备。
  - 支持 iOS 8.x 及以上系统。
  - 支持 armv64、armv7、i386、x86_64 四架构，静态库。
- 开发工具建议使用Xcode 11及以上。

SDK下载地址

IOS端SDK下载地址，请参见iOS数据语音2.1.0。

快速开始

1. 安装CocoaPods。

   使用CocoaPods，目前SDK只能收到引入，所依赖的第三方库需要使用Cocoapods，具体安装流程如下：

   i. 确保电脑已经正确安装Ruby环境（Mac电脑自带，需要确认下是否有安装）

   ii. 在Mac终端窗口中输入如下命令：

   ```
   sudo gem install cocoapods
   ```

2. 创建新工程。
   - 选择file > new > project创建新工程。
   - 用CocoaPods初始化项目。

     a. 进入您所创建项目所在路径，输入如下命令创建Podfile文件。

     ```
     pod init
     ```

b. 编辑Podfile文件。

```
platform :ios, '8.0'
target 'AlicomRTCSDKDemo' do
    pod 'OpenSSL', '~> 1.0.203'
    pod 'AliyunOSSiOS'
end
```

> 注意 如果OpenSSL无法正确安装，建议使用OpenSSL-Universal进行。

3. 导入SDK。
   - SDK 库名：AlicomRTCSDK.framework
     a. 下载AlicomRTCSDK.framework。
     b. 下载ArtcSDK.framework。
     c. 下载ArtcMediaEngine.framework。
     d. 在工程上右击选择**Add Files To>工程名称**，将刚刚下载的三个SDK添加进来。
     e. 选择对应的target，选择Frameworks，Libraries,and Embedded Content将ARTCMediaEngine.framework的Embed设置为Embed & Sign。
   - 在工程target中**Linked Frameworks and Libraries**添加**CoreTelephony.framework**，**libc++.tbd**，**libz.tbd**系统库。
   - 在工程 target 中 **Build Settings** 的 **Other Linker Flags** 增加 **-ObjC** 配置。
   - pod依赖：

```
pod 'OpenSSL', '~> 1.0.203'
```

```
pod 'AliyunOSSiOS'
```

4. OC兼容Swift。

   在已有的OC工程中新建一个Swift文件，命名为xxx.swift（可任意命名），会弹出提示，选择Create Bridging Header建立桥接文件，系统会建立**工程名-Bridging-Header.h**。

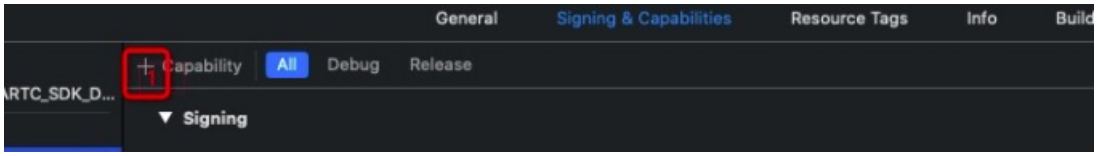
5. Swift兼容OC。

   创建一个oc文件，命名xxx，继承的父类不受限制，可以是NSObject，系统弹出提示，选择Create Bridging Header，系统会建立**工程名-Bridging-Header.h**。

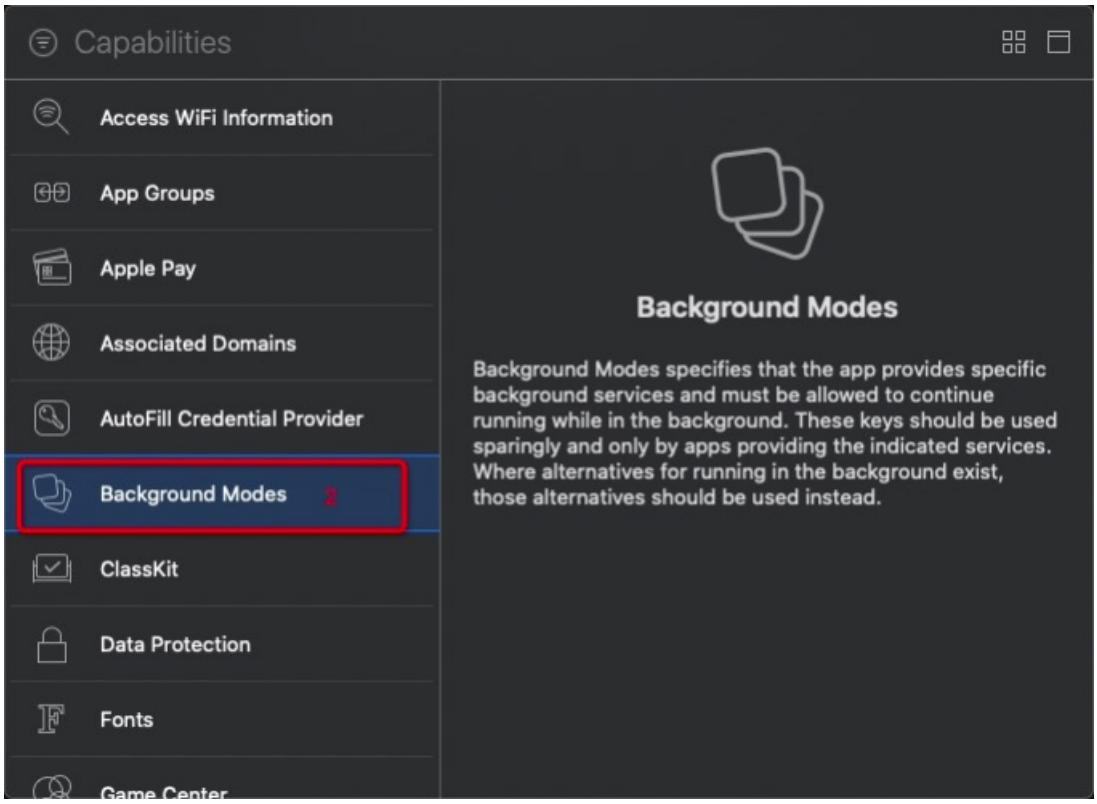
6. 开通后台权限

   由于iOS系统的限制需要注册相应的后台服务才可以在应用退到后台之后继续使用VoIP服务，这边以xcode11.4.1为例，具体开通流程如下：

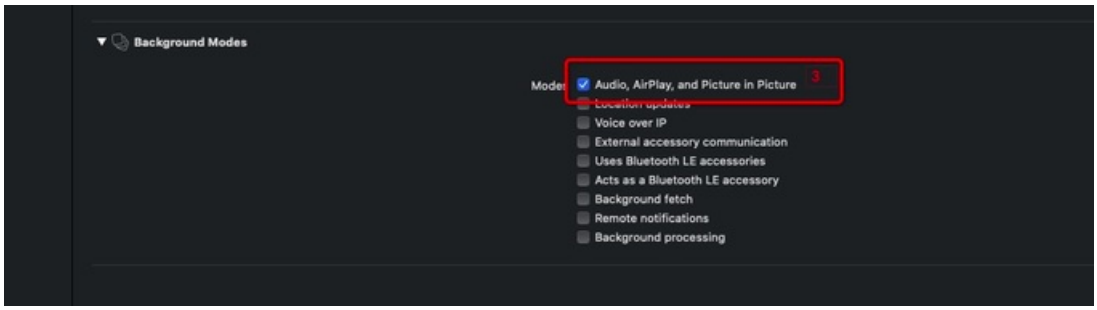
   i. 单击+号。



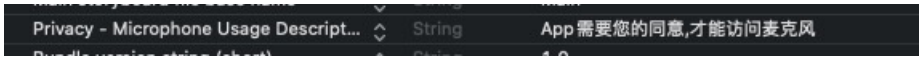
ii. 单击Background Modes。



iii. 选择Audio,AirPlay,and Picture in Picture。



7. 开通麦克风权限Privacy - Microphone Usage Description。



8. 关闭bitcode。

   设置Build Settings>Build Options>Enable Bitcode为NO。

9. 设置Strip Style。

   选择target>Build Settings>Deployment>Strip Style，将需要编译类型全部设置为Debugging Symbols。

4.2.2. SDK使用说明

本文为您介绍了在iOS开发时，SDK的使用说明及示例。

创建实例

1. 引入头文件。

```
import <AlicomRTCSDK/AlicomRTCSDK.h>
```

2. 创建AlicomRTC实例。

```
[[AlicomRTC shareInstance] initWithRtcId:rtcid];
```

> 说明
> - rtcid：融合通信账号ID 账号由服务端的AddRtcAccount接口生成（Module字段），这里需要App端调用服务端接口进行获取app->appserver->aliServer。
> - 初始化之前不需要自己主动获取Token，Token的获取会在初始化完成之后，通过回调告知何时需要获取，只要实现updateToken给SDK设置Token即可。

3. 设置相关代理。

```
[ACToken shareInstance].delegate = self;
[AlicomRTC shareInstance].delegate = self;
```

4. ACTokenDelegate代理回调

   通知客户更新Token

```
//刷新token使用
- (void)updateToken:(void(^)(ACToken *token))tokenHandler
```

> 说明 用户在实现该接口的时候可以向服务端获取token，通过服务端的GetRtcToken接口实现token获取再返给客户端。

5. AlicomRTCServiceDelegate代理回调。

```
@required
/**
 初始化成功，服务可用
 */
-(void)onServiceAvailable;
/**
 初始化失败，服务不可用
 @param errCode 错误码
 @param errMsg 错误信息
 */
-(void)onServiceUnavailableWithErrorCode:(NSInteger)errCode errorMsg:(NSString *)errMsg
;
@optional
/**
 自定义消息接受回调
 @param sequence 消息序列号
 @param content 消息内容
 @param sourceRtcId 消息来源
 @param timestamp 时间戳
 */
-(void)onCustomMessageReceivedWithSequence:(long long)sequence sourceRtcId:(NSString *)
sourceRtcId content:(NSString *)content timestamp:(long long)timestamp;
/**
 自定义消息发送回调
 @param sequence 消息序列号
 @param isSuccess 发送消息是否成功
 @param errCode 失败的话，有错误码，否则为0
 @param errMsg 失败的话，错误信息。否则为nil
 */
-(void)onCustomMessageSendedWithSequence:(long long)sequence isSuccess:(BOOL)isSuccess
errorCode:(NSInteger)errCode errorMsg:(NSString *)errMsg;
```

说明 其中 `onServiceAvailable` 和 `onServiceUnavailableWithErrorCode:errorMsg:` 是必须实现的接口。

6. 初始初化通话实例。

```
/**
 voip2voip通话
 @param calleeRtcIds 被叫rtcId账号列表
 @param cancleFlag 通话被拒接标志，默认为CALL_CANCLE_FLAG_WHEN_ONE_CALLEE_REJECT
 @param extendString  拓展字段(非必要参数)
 @return Call的对象（需判断返回是否为空值）
 */
- (Call *)createVoipCall:(NSArray *)calleeRtcIds cancleFlag:(CALL_CANCLE_FLAG)cancleFla
g withExtendString:(NSString *)extendString;
/**
 voip2pstn通话
 @param calleePhoneNumber 被叫电话号码
 @param calleeShowNumber  被叫显示来电号码
 @param customLine  自有线路名称，无则传入nil
 @param extendString  拓展字段（非必要参数）
 @return Call的对象（需判断返回是否为空值）
 */
- (Call *)createPstnCall:(NSString *)calleePhoneNumber andCalleeShowNumber:(NSString *)
calleeShowNumber customLine:(NSString *)customLine withExtendString:(NSString *)extendS
tring;
/**
 自定义消息发送（仅支持有账号模式）
 @param targetRtcId 目标rtctId账号
 @param content 自定义消息内容
 */
- (void)sendCustomMessageWithTargetRtcId:(NSString *)targetRtcId content:(NSString *)co
ntent;
```

7. 在创建完成通话实例之后需要进行通话状态管理，调用如下方法：

```
-(void)start;
```

8. 停止呼叫。

```
-(void)stop;
```

9. （可选）通话状态管理,需要实现CallDelegate。

```
/**
 被叫来电，振铃彩铃
 @param call Call对象
 */
-(NSData *)ringtoneWithCallee:(Call*)call;
/**
 主叫呼叫中，振铃彩铃
 @param call Call对象
 */
-(NSData *)ringtoneWithCaller:(Call*)call;
/**
 音频电话来电，被叫振铃，并返回振铃数据
 @param call Call对象
 */
-(void)onReceivingAudioCall:(Call*)call;
/**
 视频电话来电，被叫振铃，并返回振铃数据
```

```
 @param call VideoCall对象
 */
-(void)onReceivingVideoCall:(Call*)call;
/**
 媒体连接成功了
 @param call Call对象
 */
-(void)onCallConnected:(Call*)call;
/**
 接通对方了，主叫振铃
 @param call Call对象
 */
-(void)onCallRinging:(Call*)call;
/**
 被叫接听了，主叫停止振铃
 @param call Call对象
 */
-(void)onCallAnswered:(Call*)call;
/**
 通话结束
 @param call Call对象
 @param code code
 @param msg  msg
 */
-(void)onCallStopped:(Call *)call withCode:(NSInteger)code andMsg:(NSString *)msg;
/**
 音频输出模式变更
 @param call call对象
 @param mode 输出mode
 */
-(void)onPlayModeChanged:(Call*)call withMode:(Audio_Session_Mode)mode;
/**
 音频接收dtmf数据
 @param call call对象
 @param dtmf 数据内容,timestamp 时间戳
 */
-(void)onDtmfData:(Call*)call withDtmf:(NSString *)dtmf withTimestamp:(long long)timest
amp;
/**
 通话中的网络质量
 @param call Call对象，quality，网络质量，0-好，1-中，2-差
 */
-(void)onNetworkQuality:(Call*)call withQuality:(int)quality;
/**
 通话中的指标统计
 @param call Call对象，model StatisticModel 对象
 */
-(void)onMediaStatistics:(Call*)call withModel:(StatisticModel *)model;
/**
 音视频模式变更
 @param call call对象
 @param mode 输出mode，0-视频，1-音频
 */
-(void)onMediaModeChanged:(Call*)call withMode:(int )mode;
```

功能使用

- 本地静音。

```
-(void)mute;
```

- 取消本地静音。

```
-(void)unmute;
```

- 呼叫默认超时时间设置，默认45s，可设置时间区间30s~90s。

```
-(void)setCallTimeout:(int) time;
```

- 发送dtmf，仅在点对点通话中对端为PSTN且通话中才能发送，且只支持 `0-9、*、#` 输入。

```
-(BOOL)sendDtmfData:(NSString *)dtmf;
```

- 切换音频播放路由。

```
-(void)audioSpeakerIsOn:(BOOL)isOn; //isOn，音频播放路由：YES-扬声器播放，NO-听筒播放。
```

> **说明** 调用该方法前需要确保通话已经建立。

4.3. Web

4.3.1. Web开发

客户可通过接入Web SDK快速实现浏览器上的音频通话及视频通话等功能，依托阿里的研发资源，我们为客户提供安全、可靠、弹性、低成本的实时通信服务。

功能特性

提供浏览器端VoIP to PSTN，PSTN to VoIP, VoIP to VoIP通话场景的服务，接入方可根据需求选择对应的应用场景。

开发准备

客户端需要从阿里云申请融合通信账号后才能接入，为了保证安全性，融合通信SDK通过服务端下发的临时token作为身份标识与服务进行交互。临时token可通过融合通信服务pop接口从阿里云获得（详见服务端接入文档），并且具有时效性。SDK在token即将失效或其他需求而需要更新token时，会通过回调通知接入方，接入方需自行实现接口获取最新token并传递给SDK。

npm地址

1.7.0版本

使用环境

使用环境如下：

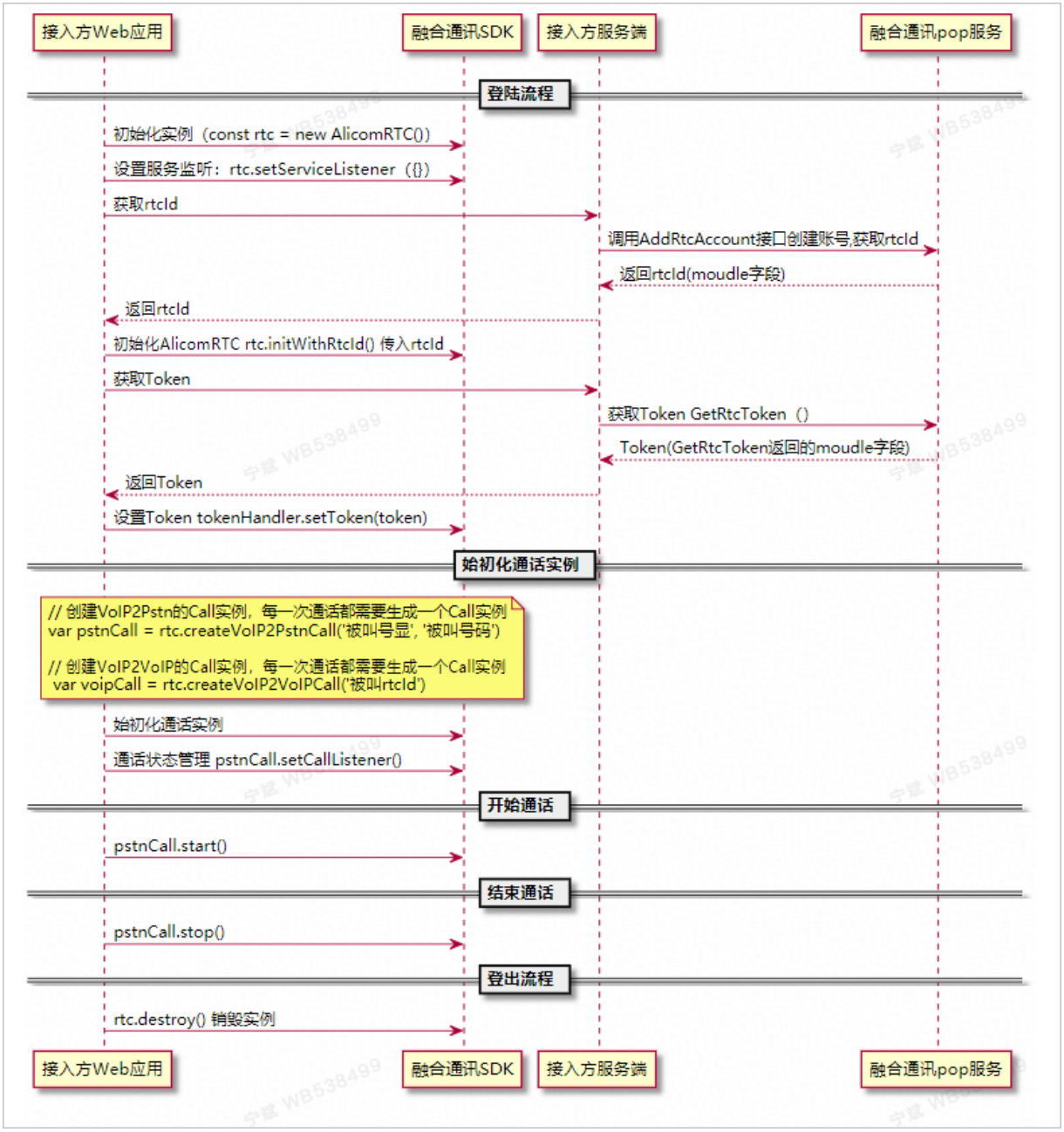
- 融合通信服务仅支持在HTTPS环境使用。
- 请保证在音频设备完好且允许麦克风使用的情况下此可以使用融合通信服务。

- 浏览器限制：

浏览器	版本
Chrome	不低于65版本
Safari	不低于11版本
Firefox	不低于65版本

> **说明** 为保证稳定性，建议使用最新版Chrome使用该功能。

快速开始



1. 开发环境配置

```
npm install aliyun-voip-web-sdk -S
```

2. 导入SDK

```
import AlicomRTC from 'aliyun-voip-web-sdk';
```

创建实例

创建AlicomRTC实例

```
初始化实例
const rtc = new AlicomRTC();
```

设置AlicomRTC的参数与回调（可选）

```
/**
 * 设置服务监听
 */
rtc.setServiceListener({
  /**
   * 服务连接成功，当前处于可用状态
   */
  onServiceAvailable: function () {
    // 此时可以开始呼叫操作
  },
  /**
   * 服务连接失败、或者连接断开了，当前处于不可用状态，服务正在销毁。
   * @param errCode {number} 错误码
   * @param errMsg {string} 错误描述
   */
  onServiceUnavailable: function (errCode, errMsg) {
  },
  /**
   * 服务闲时
   * @param errCode {number} 错误码
   * @param errMsg {string} 错误描述
   */
  onServiceIdle: function (errCode, errMsg) {
  },
  /**
   * 作为被叫，收到点对点音频来电
   * @param call {Call} 表示这通来电的call实例
   */
  onReceivingAudioCall: function(call) {
    // todo: 已有通话的情况下拒绝新呼入，或是挂断当前通话接听
  },
})
```

回调函数说明

- onServiceAvailable

服务连接成功回调，表示当前处于可用状态。可在回调函数中开始呼叫操作。

- onServiceUnavailable

  服务连接失败回调，表示当前处于不可用状态，服务正在销毁。

- onServiceIdle

  服务闲时，表示当前服务出于空闲状态。可在回调函数中设置服务未初始化空闲状态。

- onReceivingAudioCall

  作为被叫，收到点对点音频来电。可在回调函数中进行已有通话的情况下拒绝新呼入，或是挂断当前通话接听的操作。

回调函数的参数说明

- onReceivingAudioCall：参数call 表示这通来电的call实例。
- onServiceUnavailable onServiceIdle参数 errCode错误码 errMsg 错误描述。

连接服务

```
/**
 * 使用rtcId初始化AlicomRTC实例
 * @param rtcId 融合通信账号ID 账号由服务端的AddRtcAccount接口生成（Module字段）
 */
rtc.initWithRtcId(rtcId, {
  updateToken: function (tokenHandler) {
    var token = {} // todo: 通过服务端接口获取Token
    tokenHandler.setToken(token)
  }
})
```

参数数名

- rtcId：融合通信账号ID 账号由服务端的AddRtcAccount接口生成（Module字段）
- updateToken： 通知客户更新token的回调事件

在此回调事件中，客户可向客户服务端获取token（即GetRtcToken接口的出参Module字段，详见服务端文档），并通过 `tokenHandler.setToken(token)` 回传至SDK。

初始初化通话实例

当ServiceListener中的onServiceAvailable回调被唤起时，表示服务已经连接成功，此时可以开始创建通话实例，设置并开始通话。

```
/**
 * 创建VoIP2Pstn的Call实例，每一次通话都需要生成一个Call实例
 */
const pstnCall = rtc.createVoIP2PstnCall(calleeShowNumber, calleePhoneNumber)
/**
 * 创建VoIP2VoIP的Call实例，每一次通话都需要生成一个Call实例
 */
const voipCall = rtc.createVoIP2VoIPCall(calleeRtcId)
```

参数说明

- calleeShowNumber 被叫号显，被叫收到电话时显示的来电号码

- calleePhoneNumber 被叫号码
- calleeRtcId 被叫的rtcId 即被叫的融合通信账号ID

通话相关回调设置

```
/**
 * 通话相关回调设置
 */
pstnCall.setCallListener({
  /**
   * 电话已通知到被叫方，被叫振铃中
   * @param talk 通话对象
   */
  onCalleeRinging(talk: Talk): void;
  /**
   * 被叫已接听电话，正在连接中。（拨打类型为PSTN与呼叫中心的电话时无此回调）
   * @param talk 通话对象
   */
  onCalleeConnecting(talk: Talk): void;
  /**
   * 电话拨通了可以正常通话了。主叫中代表被叫已接听，被叫中表示电话已接听
   * @param talk 通话对象
   */
  onActive(talk: Talk): void;
  /**
   * 对端传来DTMF信息，每次回调只传入一个字符，如果是一连串输入的话则会多次回调，需要接入方自行根据时间戳处理
   * @param dtmf 对端传入的DTMF信息
   * @param timestamp 时间戳
   * @param talk 通话对象
   */
  onDtmfData(dtmf: string, timestamp: number, talk: Talk): void;
   /**
   * 通话成功连接到媒体服务器
   * @param talk
   */
  onConnected(talk: Talk): void;
  /**
   * 通话正在断开
   * @param errCode 错误码
   * @param errMsg 错误消息
   * @param talk 通话对象
   */
  onStopping(errCode: number, errMsg: String, talk: Talk): void;
  /**
   * 通话已结束
   * @param errCode 错误码
   * @param errMsg 错误消息
   * @param talk 通话对象
   */
  onStopped(errCode: number, errMsg: string, talk: Talk): void;
  /**
   * 通话中网络质量消息
   * @param quality 质量
```

```
    * @param talk 通话对象
    */
  onNetworkQuality(quality: NetworkQuality, talk: Talk): void;
  /**
    * 通话中的监控数据回调
    * @param monitorStats 监控数据
    * @param talk 通话对象
    */
  onMediaStatistics(monitorStats: any, talk: Talk): void;
})
```

参数说明

- onCalleeRinging 电话已通知到被叫方，被叫振铃中。
- onCalleeConnecting 被叫已接听电话，正在连接中。
- onActive 电话拨通了可以正常通话了。被叫已接听。
- onDtmfData 对端传来DTMF信息每次回调只传入一个字符，如果是一连串输入的话则会多次回调，需要接入方自行根据时间戳处理。
- onConnected 通话成功连接到媒体服务器。
- onStopping 通话正在断开。
- onStopped 通话已结束。
- onNetworkQuality 通话中网络质量消息；
- onMediaStatistics 通话中的监控数据回调。

主叫开始呼叫 / 被叫开始接听

```
pstnCall.start()
```

停止呼叫/通话

```
pstnCall.stop()
```

销毁实例

```
rtc.destroy()
```

4.3.2. SDK接口

本文为您介绍了Web SDK的功能说明以及错误码描述。

SDK DEMO使用

- VoIP Web SDK的Demo项目已开源至Github社区，您可通过Demo演示快速进入开发，VoIP Web Demo。
- AK和SK即AccessKey ID和AccessKey Secret是您访问阿里云API的密钥对，可以在密钥管理平台管理查看。
- 融合通信账号可以由AddRtcAccount接口获取。

功能使用

- 设置默认的呼叫超时时间。

```
rtc.setDefaultCallTimeout(seconds)
```

> **说明** seconds number类型，呼叫超时时间，单位为秒，限制在30秒~90秒之间；小于30按30算，大于90按90算。

- 是否允许日志上传，默认不上传。

```
/**
 * 是否允许日志上传，默认不上传
 * @param enable true 上传日志  false 不上传
 */
setUploadEnable(enable)
```

> **说明**
> - 为排查问题方便，建议设置为true。
> - 此接口适用版本1.7.0及以上。

- （可选）开启服务端录音。

```
pstnCall.setServerRecordEnabled(true)
```

- （可选）本地禁音。

```
pstnCall.muteLocalAudio()
```

- （可选）本地取消禁音。

```
pstnCall.unmuteLocalAudio()
```

- 发送dtmf。

```
pstnCall.sendDtmfData(dtmf)
```

> **说明** 只能是0-9、*、#组成的字符串，最大长度不超过32位。

- 获取当前通话的ChannelId。

  当通话遇到问题时提供ChannelId，可以用来查当前通话的日志埋点。

```
getChannelId(): string
```

错误码

错误码	数值	原因
ERROR_MQTT_CONNECT_FAIL	1000100	mqtt连接失败。
ERROR_UPLOAD_TOKEN_FAIL	1000104	token上传失败。

错误码	数值	原因
ERROR_LOCAL_STOP	2000000	本地主动挂断。
ERROR_REMOTE_HANGUP	2000099	对端主动挂断。
ERROR_CALL_SELF	2000100	拨打自己。
ERROR_CALL_EMPTY	2000101	拨打电话为空。
ERROR_CALLEE_ALERTING_TIMEOUT	2000103	被叫时，振铃超时。
ERROR_REMOTE_REFUSE	2000106	主叫时，对端拒绝应答。
ERROR_REMOTE_CANCEL	2000107	被叫时，对端取消呼叫。
ERROR_NO_ANSWER	2000108	被叫无人接听。
ERROR_SERVICE_UNAVAILABLE	2000113	AlicomRTC服务不可用。
ERROR_LOCAL_DESTROY	2000116	本地主动销毁服务。
ERROR_TIME_OUT	2000102	joinChannel超时。
ERROR_NETWORK_INVALID	2000121	网络异常。
ERROR_MICROPHONE_NO_PERMISSION	2000130	麦克风无权限。
ERROR_MICROPHONE_NO_DEVICE	2000135	没有麦克风设备。
ERROR_MICROPHONE_NOT_ALLOWED	2000136	浏览器不允许使用麦克风。
ERROR_MICROPHONE_NOT_READABLE	2000137	系统禁用麦克风或者麦克风被占用。
ERROR_SERVER_BASE	3000000	服务端返回业务异常的错误基准。
ERROR_SERVER_KICKED	3110000	账号被登录或被踢。
ERROR_SERVER_UNAVAILABLE	3110001	服务异常。
ERROR_MEDIA_BASE	4000000	媒体SDK返回异常的错误基准。