ALIBABA CLOUD

阿里云

音视频通信
快速入门

文档版本：20220608

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.入门概述 06
2.开通服务 07
3.创建应用 09
4.集成客户端SDK 10
4.1. Android 10
4.2. iOS 11
4.3. Mac 15
4.4. Windows 17
4.5. Electron 20
4.6. 微信小程序 22
4.7. Web 22
4.8. Linux 23
4.8.1. C++ 23
4.8.2. Java 23
5.实现基本功能 25
5.1. Android 25
5.2. iOS 29
5.3. Mac 33
5.4. Windows 37
5.5. Electron 41
5.6. Web 44
5.7. Linux 46
5.7.1. C++ 46
5.7.2. Java 51
6.实现基本功能（旧版） 56
6.1. Android 56

6.2. iOS ---- 61
6.3. Mac ---- 65
6.4. Windows ---- 69

1.入门概述

阿里云RTC为您提供搭建音视频实时通话的全套SDK。通过阅读本文，您可以了解根据RTC SDK实现音视频通话的流程。

入门流程

步骤	操作	描述
1	开通服务	您可以快速开通RTC服务并购买时长包。
2	创建应用	通过在控制台创建应用，可以获取您的应用ID。
3	集成客户端SDK	阿里云RTC为您提供了移动端、桌面端、Web端等SDK，帮助您快速集成SDK。
4	实现基本功能	RTC SDK为您提供了初始化SDK、加入频道、发布和订阅、离开频道等基本功能的实现方法。
5	应用管理	您可以使用控制台查看、管理应用。

2.开通服务

通过阅读本文，您可以了解如何开通阿里云RTC服务及如何购买时长包。

前提条件

您已经注册了阿里云账号并完成账号实名认证。注册地址请参见阿里云官网。注册指引请参见注册阿里云账号。实名认证指引请参见个人实名认证或企业实名认证。

开通服务

1. 登录阿里云音视频通信。
2. 单击**立即开通**。
3. 根据实际情况选择**服务类型**、**计费方式**等配置项，并选中**服务协议**。

4. 单击**立即开通**，完成开通服务。

> **说明** 阿里云音视频通信服务开通页面默认选择服务类型、计费方式和服务区域，如果您想开通海外服务，请先咨询客户经理或提交工单。

购买音视频通信时长包（资源包）

1. 登录音视频通信RTC控制台。
2. 在**概览**页面右上角，单击**购买**。

> **说明** 目前阿里云RTC仅支持购买国内版资源包。

3. 根据实际情况选择**资源类型**、**包规格**等配置项，并单击**立即购买**。

4. 选中**服务协议**，并单击**去支付**。
5. 确认订单，单击**确认支付**完成购买资源包。

后续步骤

当您完成开通RTC服务并购买时长包后，您可以在控制台对应用进行操作，详情请参见应用管理。

3.创建应用

通过阅读本文，您可以了解如何在控制台创建应用。

前提条件

- 您已经注册了阿里云账号并完成账号实名认证。注册地址请参见阿里云官网。注册指引请参见注册阿里云账号。实名认证指引请参见个人实名认证或企业实名认证。
- 您已经开通音视频通信服务。具体操作，请参见开通服务。

操作步骤

1. 登录音视频通信RTC控制台。
2. 在左侧导航栏单击**应用管理**，进入应用管理配置界面。
3. 单击**创建应用**。
4. 在**音视频通信（按量付费）**页面，选中**服务协议**。

选项	说明
服务类型	RTC服务类型默认为**通用服务**。
计费方式	RTC计费方式默认为**按时长**。
服务区域	服务区域默认开通中国。如果您想开通美国区域，请先咨询客户经理或提交工单。

5. 单击**立即开通**，完成创建应用。
6. 返回**应用管理**页面，您可以查看新开通应用信息。

> **说明** 应用ID是阿里云RTC服务器用于区分不同应用的唯一标识。

后续步骤

您可以在音视频通信RTC控制台管理和查询应用。具体操作，请参见应用管理。

4.集成客户端SDK

4.1. Android

通过阅读本文，您可以了解Android端集成SDK的方法。

环境要求

Android端具体环境要求，更多信息，请参见使用限制。

集成SDK

方法一：Maven集成（推荐）

1. 在根目录的build.gradle中添加Maven仓库地址：

```
allprojects {
    repositories {
        google()
        jcenter()
        //添加RTC需要的Maven地址
        maven {
            url "http://maven.aliyun.com/nexus/content/groups/public/"
        }
    }
}
```

2. 在项目的*app/build.gradle*文件中，添加如下行：

```
dependencies {
        ...
    //依赖的RTC SDK
    implementation 'com.aliyun.rtc:AliRTC-Full:1.17.9.2005112'
}
```

> **说明** 此处Maven依赖的版本仅供参考，获取最新的Maven依赖，请参见SDK下载。

3. 在*app/src/main/AndroidManifest.xml*文件中添加如下代码，获取相应的设备权限。

```
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
```

4. （可选）配置防止代码混淆。

   在*proguard-rules.pro*文件中，添加 `-keep` 类的配置，可以防止混淆RTC SDK公共类名称。

```
-keep class com.serenegiant.**{*;}
-keep class org.webrtc.**{*;}
-keep class com.alivc.**{*;}
```

方法二：手动集成

1. 下载并解压Android SDK，下载地址，请参见SDK下载。
2. 复制SDK文件AliRTCSdk.aar到App模块下的libs文件夹中。
3. 在*app/src/main/AndroidManifest.xml*文件中添加如下代码，获取相应的设备权限。

```
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
```

4. 配置防止代码混淆。

   在*proguard-rules.pro*文件中，添加 `-keep` 类的配置，可以防止混淆RTC SDK公共类名称。

```
-keep class com.serenegiant.**{*;}
-keep class org.webrtc.**{*;}
-keep class com.alivc.**{*;}
```

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能。具体操作，请参见Android端实现基本功能。

4.2. iOS

通过阅读本文，您可以了解iOS端集成SDK的方法。

前提条件

- 环境中已安装Xcode 9.0或以上版本，更多信息，请参见Xcode。
- 您需要持有Apple开发证书或个人账号。

环境要求

iOS端具体环境要求，更多信息，请参见使用限制。

pod方式集成

> **注意** 请确保您的Mac已经安装Ruby环境。

1. 打开终端窗口。
2. 安装CocoaPods。

   ```
   sudo gem install cocoapods
   ```

3. 创建Podfile文件。

进入项目所在路径，执行以下命令创建Podfile文件。

pop init

4. 编辑Podfile文件。

```
platform :ios, '8.0'
target 'AliRTCPodTest' do
    pod 'AliRTCSdk', '1.17.44'
end
```

> **说明** 此处pod版本号仅供参考，获取最新的pod版本号，请参见SDK下载。

5. 安装SDK。

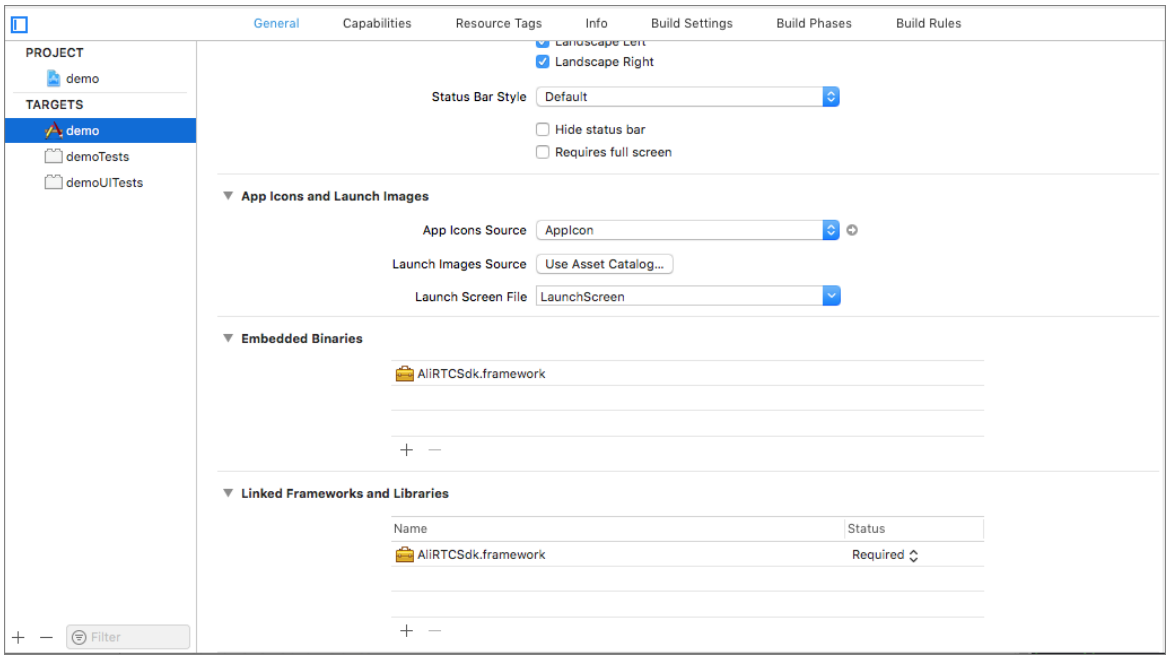
pod install

命令执行完毕之后，会生成*.xcworkspace文件，表示SDK集成完成。

手动集成

1. 下载并解压iOS SDK，下载地址请参见SDK下载。
2. 新建工程，将解压后的SDK文件复制到工程中。
3. 在**General**页签中将SDK中*AliRTCSdk.framework*文件加入到工程。

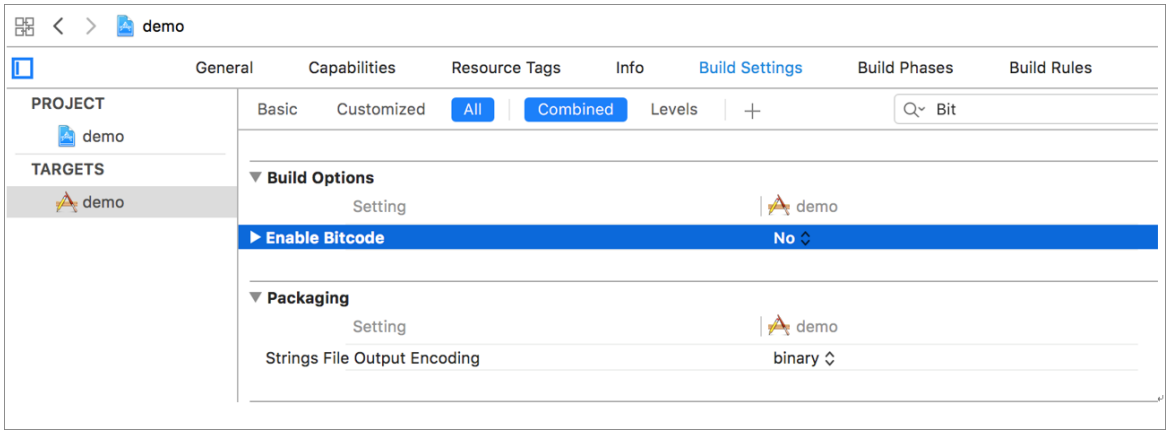
> **说明** iOS SDK1.7版本以上为动态库SDK，需要加载到Embedded Binaries中。



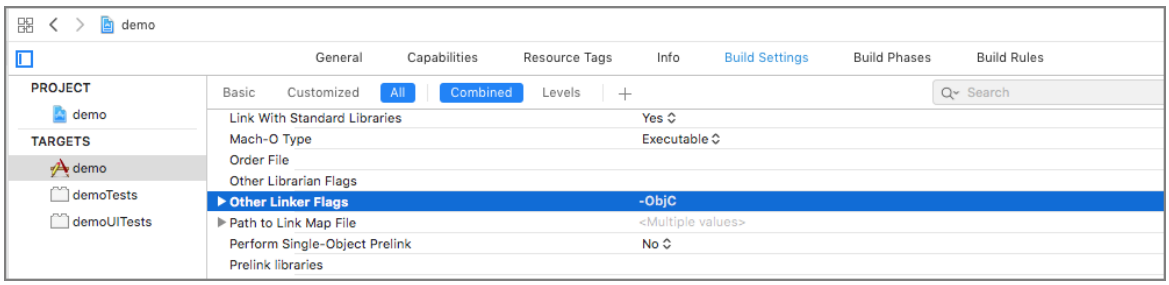
4. 在**Build Phases**页签中添加以下系统依赖。
   - *libc++.tbd*
   - *CoreMedia.framework*
   - *AVFoundation.framework*
   - *libz.tbd*

- *libresolv.tbd*
- *AudioToolbox.framework*
- *VideoToolbox.framework*

5. 在Build Settings页签中设置Enable Bitcode为No。

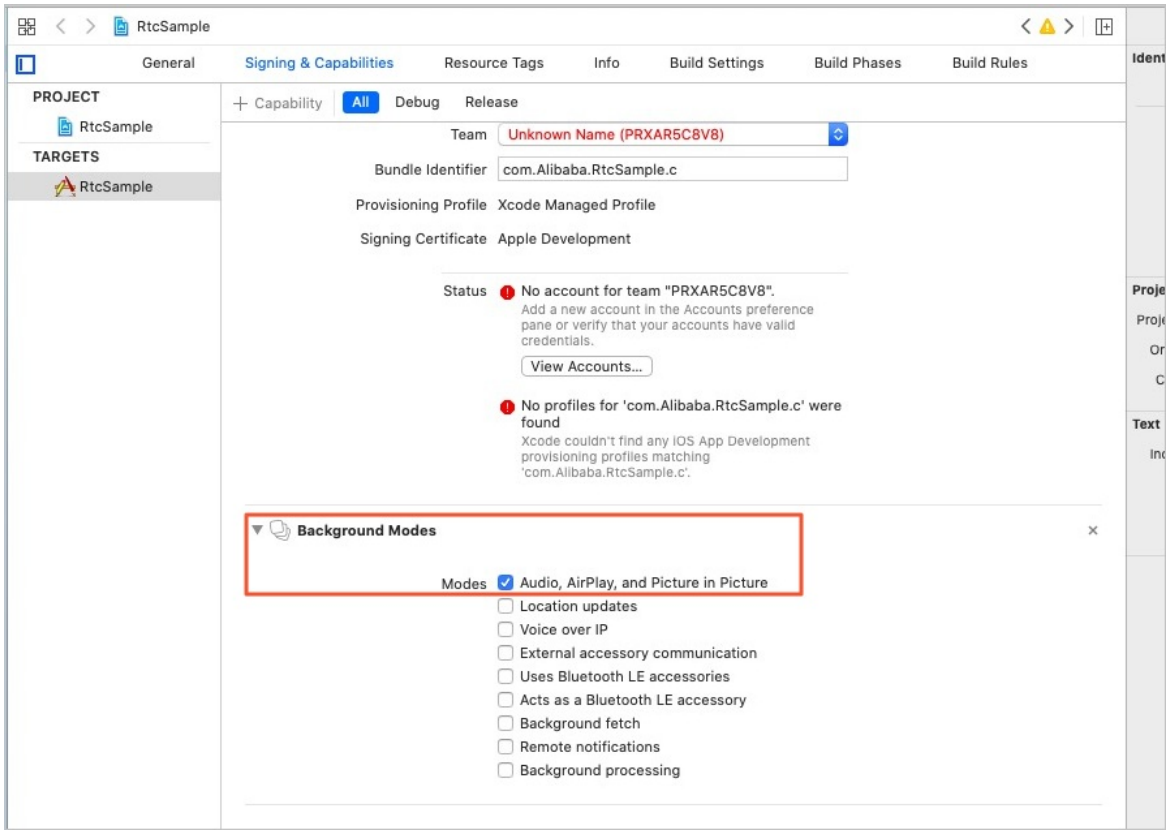


6. 在Build Settings页签中添加*-ObjC*链接选项。

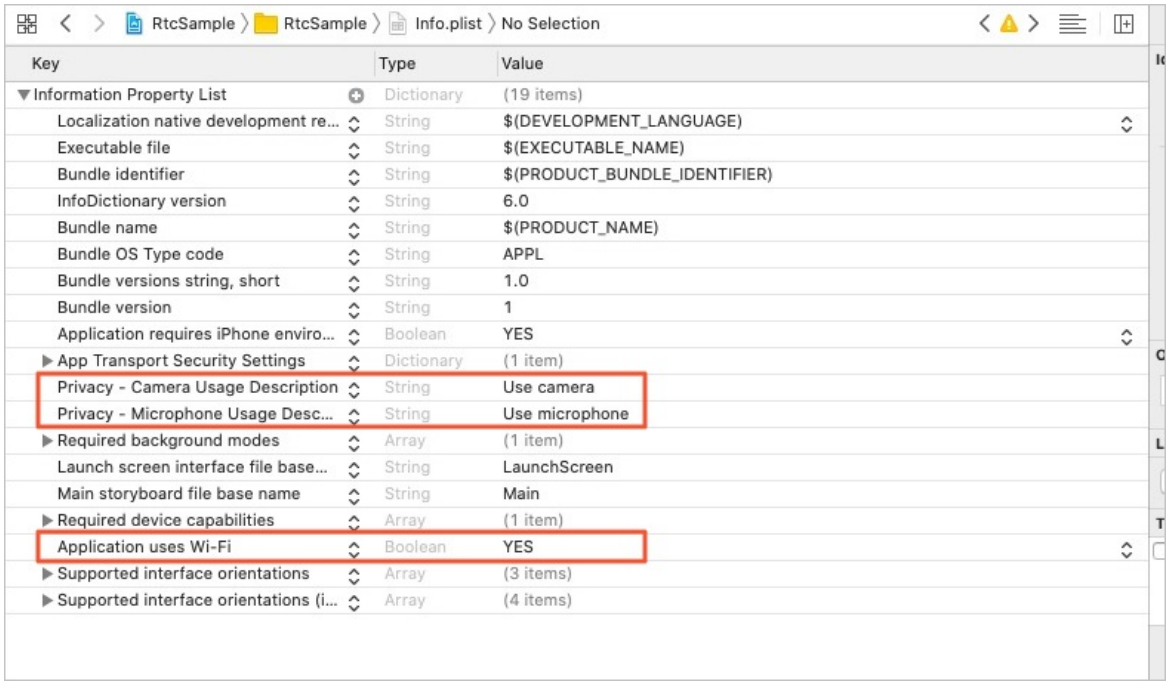


7. 在Signing & Capabilities页签中打开后台音频权限。

> **说明** 为保障应用进入手机后台之后，通话可以保持不中断，建议您开启后台音频权限，SDK默认进入后台之后继续推送音频流。



8. 编辑*info.plist*文件，添加权限。



9. 使用Xcode连接终端设备，按**Commond+B**，如果界面提示**Build Success**，表示SDK集成成功。

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能。具体操作，请参见iOS端实现基本功能。

4.3. Mac

通过阅读本文，您可以了解Mac端集成SDK的方法。

前提条件

- 环境中已安装Xcode 9.0或以上版本，更多信息，请参见Xcode。
- 您需要持有Apple开发证书或个人账号。
- 如果使用Mac mini等不包含自带摄像头和麦克风的设备，需要插入外置摄像头和麦克风。

环境要求

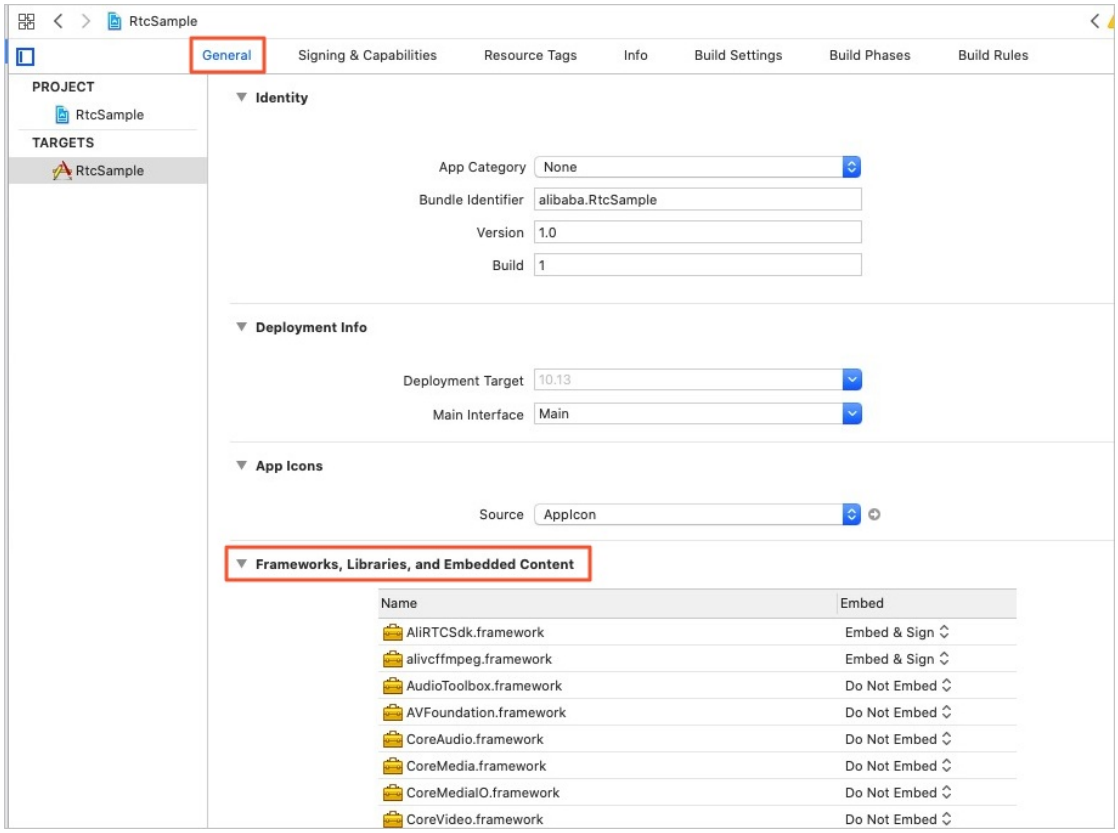
Mac端具体环境要求，更多信息，请参见使用限制。

集成SDK

1. 下载并解压Mac SDK，下载地址请参见SDK下载。
2. 新建工程，将解压后的SDK文件复制到工程中。
3. 在工程中添加SDK中的依赖文件。
    i. 在**Build Phases**页签中，在**Link Binary With Libraries**区域添加依赖文件*AliRTCSdk.framework*和*UTDID.framework*。

ii. 在**General**页签中，在**Frameworks, Libraries, and Embedded Content**区域中添加*UTDID.framework*，并将对应的**Embed**属性设置成**Embed & Sign**。

> **说明** Mac SDK1.1版本增加了*UTDID.framework*，该库为动态库，需要加载到**Embedded Binaries**中。

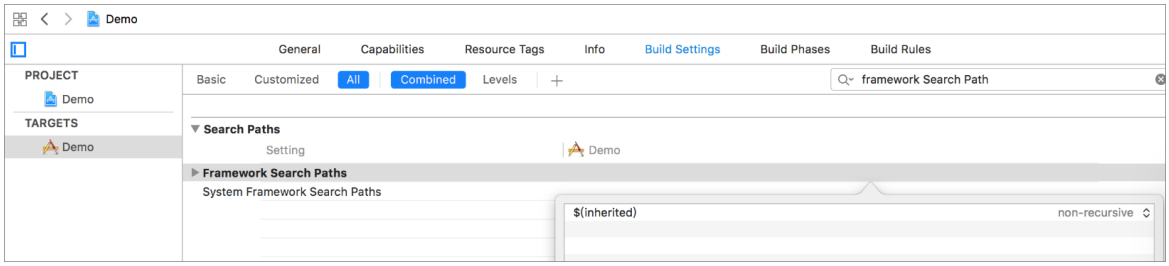


4. 在**Build Phases**页签中添加以下系统依赖。

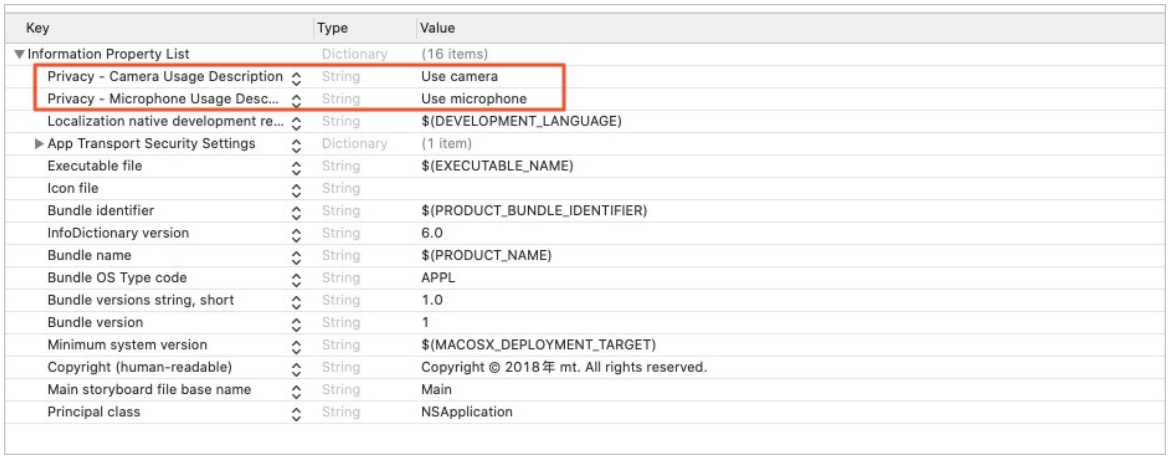
   相关系统库如下所示：

   - libc++.tbd
   - libresolv.tbd
   - libcurl.tbd
   - libz.tbd
   - CoreMedia.framework
   - CoreAudio.framework
   - AudioToolbox.framework
   - AVFoundation.framework

5. 在**Build Settings**页签中，在**Framework Search Path**区域，将*AliRTCSDK.framework*文件夹拖入弹框内。

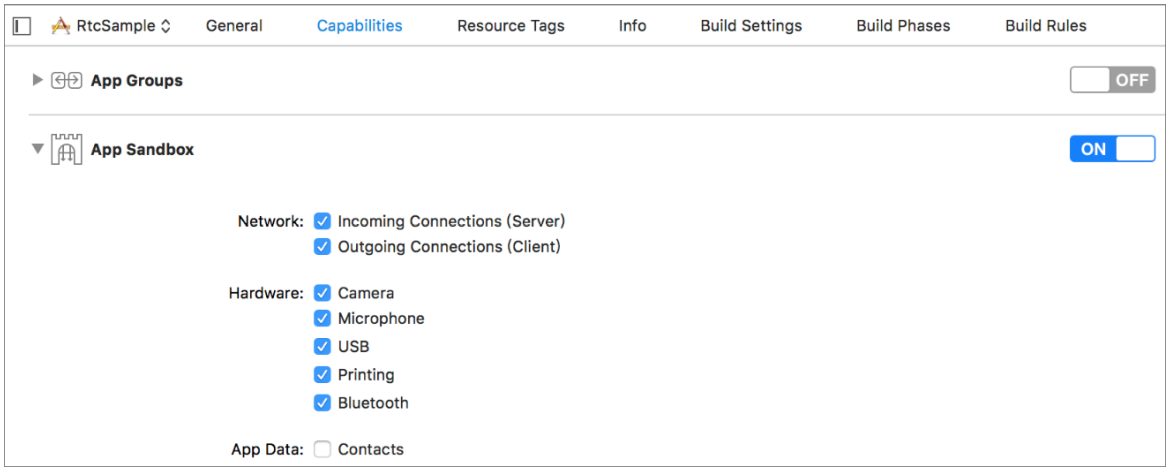


6. 编辑*info.plist*文件，添加权限。



Key	Type	Value
Information Property List	Dictionary	(16 items)
Privacy - Camera Usage Description	String	Use camera
Privacy - Microphone Usage Desc...	String	Use microphone
Localization native development re...	String	$(DEVELOPMENT_LANGUAGE)
App Transport Security Settings	Dictionary	(1 item)
Executable file	String	$(EXECUTABLE_NAME)
Icon file	String	
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Minimum system version	String	$(MACOSX_DEPLOYMENT_TARGET)
Copyright (human-readable)	String	Copyright © 2018年 mt. All rights reserved.
Main storyboard file base name	String	Main
Principal class	String	NSApplication

7. 在**Signing & Capabilities**页签中设置权限。



8. 按**Commond+B**，如果界面提示**Build Success**，表示SDK集成成功。

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能，详情请参见Mac端实现基本功能。

4.4. Windows

通过阅读本文，您可以了解Windows端集成SDK的方法。

前提条件

环境中已安装Visual Studio 2010或以上版本。

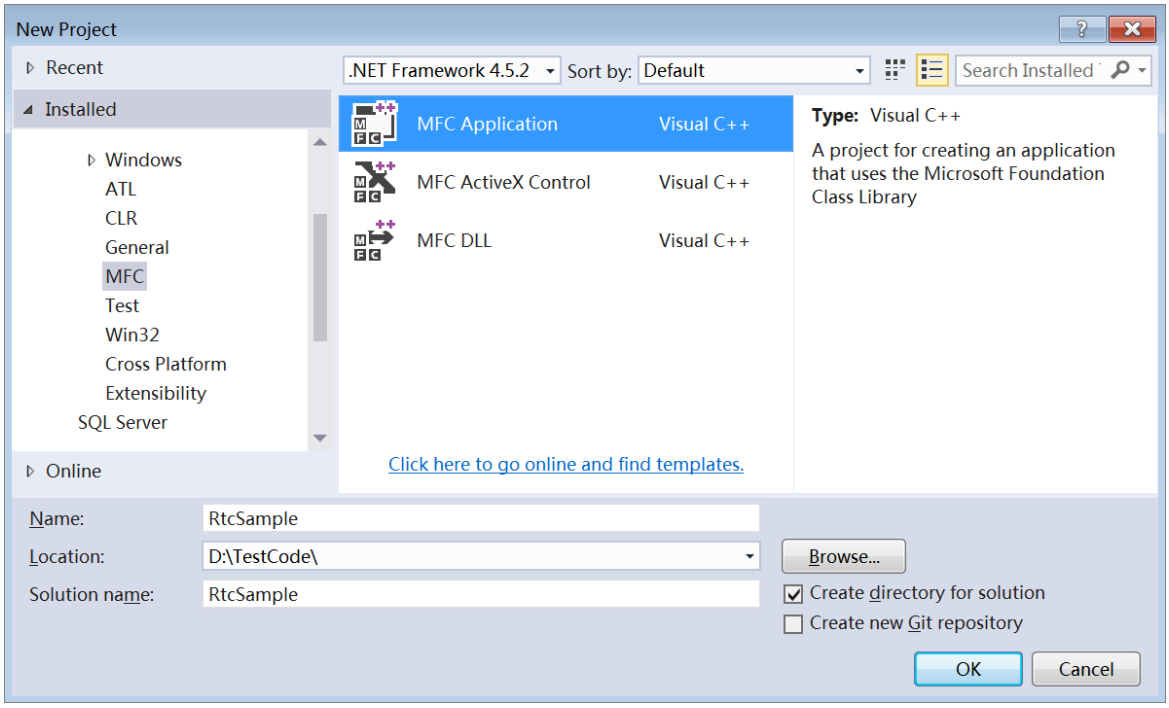
环境要求

Windows端具体环境要求，更多信息，请参见使用限制。

集成SDK

> **说明** 本文以MFC的工程为例说明，您也可以根据实际情况选择其他UI框架。

1. 下载并解压Windows SDK，下载地址请参见SDK下载。
2. 使用Visual Studio创建一个MFC Dialog based类型的工程，并输入工程名，例如：RtcSample。

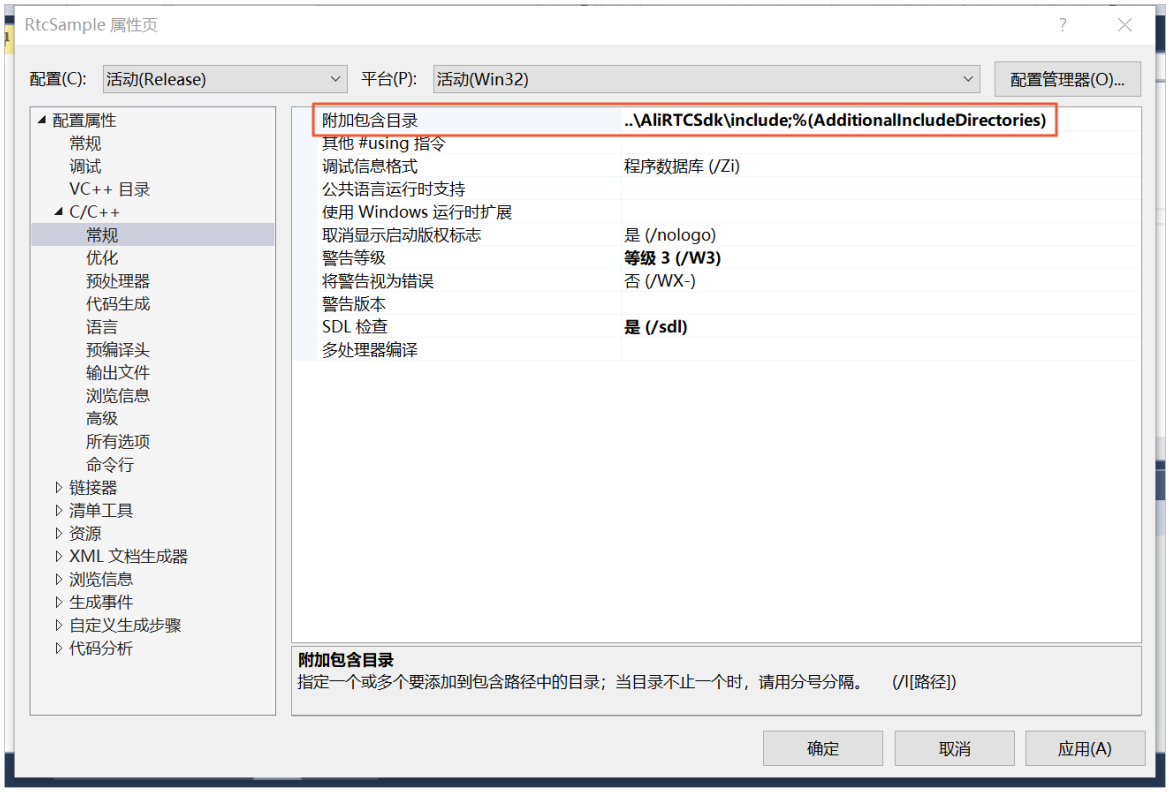


3. 将解压好的SDK的相关文件移动到*.sln*文件所在的目录。
4. 配置RtcSample工程的属性，添加SDK库及头文件的路径。

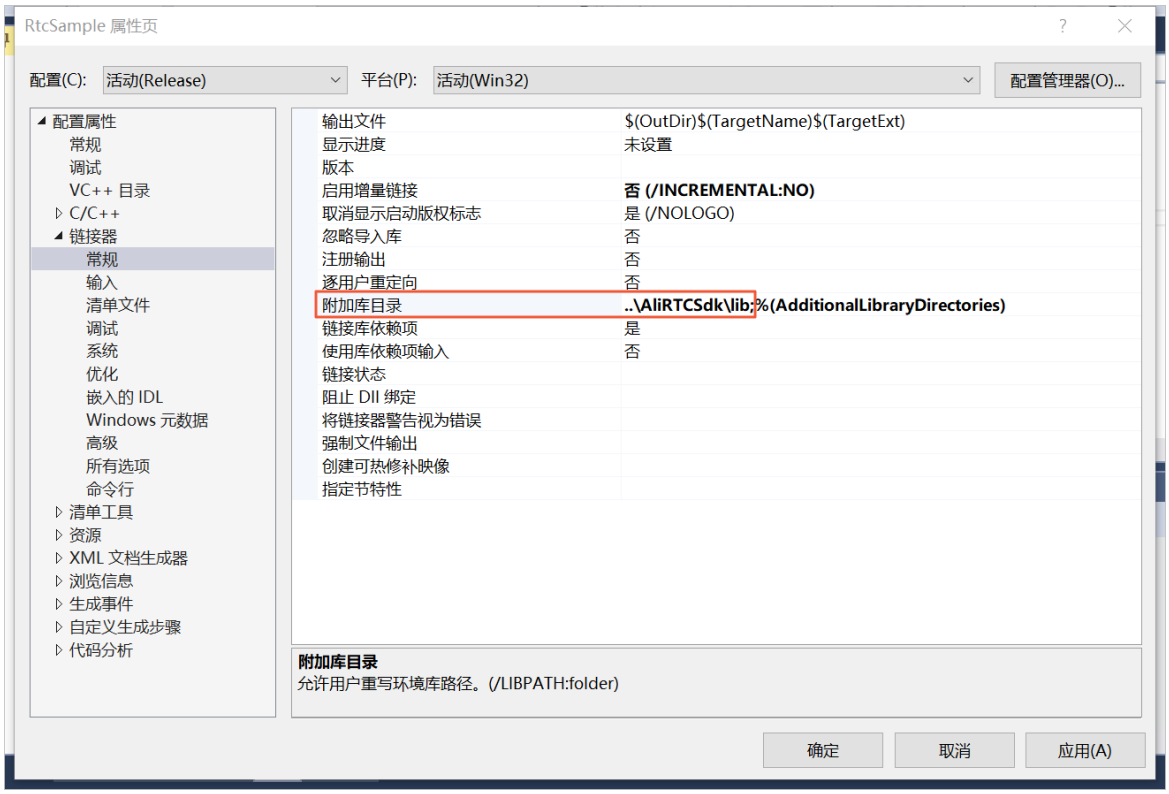
> **说明** 当前版本SDK只支持Release x86版。

5. 设置32位工程属性。

6. 在工程属性页面，设置依赖头文件的路径。



7. 设置静态库的路径。



8. 复制*AliRTCSdk.dll*文件及依赖的*alivcffmpeg.dll*文件到程序的执行路径下。
9. 执行编译。如果编译成功，表示SDK集成成功。

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能，详情请参见Windows端实现基本功能。

4.5. Electron

通过阅读本文，您可以了解集成Electron SDK的方法。

环境要求

- Electron版本：6.0.10或12.0.11。
- 开发平台：Mac、Windows（建议Windows端使用32位Node文件）。

前提条件

环境中已安装Node.js 6.0或以上版本。具体操作，请参见安装Node.js。

操作步骤

1. 修改项目中*package.json*文件。
   i. 配置 `postinstall` ，指定Electron版本。

```
"scripts": {
    "postinstall": "node node_modules/aliyun-webrtc-electron-sdk/dist/bin/alirtcdow
n -v 12.0.11"
  ...
}
```

命令	是否必填	说明
-v	必填	Electron版本，当前支持的版本为6.0.10或12.0.11。
-p	选填	系统类型，取值： ■ darwin：Mac系统。 ■ win32：Windows系统。
-a	选填	系统位数，取值： ■ ia32：对应Windows系统。 ■ x64：对应Mac系统。

   ii. 配置 `dependencies` ，添加Electron SDK。

```
"dependencies": {
  ...
    "aliyun-webrtc-electron-sdk": "2.5.2"
  ...
}
```

> **说明**
> - 此处以Electron SDK版本为2.5.2举例说明，具体版本请以实际为准。
> - 您也可以通过执行**npm i aliyun-webrtc-electron-sdk**命令安装Electron SDK。

2. 在*package.json*文件所在的目录下执行以下命令，安装项目依赖。

   npm install

3. 修改示例代码中*package.json*文件，配置 `build` ，指定Electron SDK解压后的路径。

```
"build": {
    "extraResources": [
      "./node_modules/aliyun-webrtc-electron-sdk/**"
  ]
}
```

> **说明** 由于Electron SDK用到的动态链接库打包成.asar文件后无法正常使用，因此需要配置解压后的路径。

完整的package.json配置示例，更多信息，请参见package.json配置参考。

package.json配置参考

```
{
  "name": "alirtc-electron-quick-start",
  "version": "1.0.0",
  "description": "An alirtc electron sdk quick start application",
  "main": "main.js",
  "scripts": {
    "postinstall": "node node_modules/aliyun-webrtc-electron-sdk/dist/bin/alirtcdown -v 12.
0.11",
    "start": "electron ."
  },
  "keywords": [
    "alirtc",
    "electron"
  ],
  "author": "alirtc",
  "license": "MIT",
  "devDependencies": {
    "electron": "^12.0.11",
    "electron-builder": "^20.28.4"
  },
  "dependencies": {
    "aliyun-webrtc-electron-sdk": "^2.5.2"
  },
  "build": {
    "extraResources": [
      "./node_modules/aliyun-webrtc-electron-sdk/**"
    ]
  }
}
```

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能，详情请参见Electron端实现基本功能。

4.6. 微信小程序

通过阅读本文，您可以了解集成小程序SDK的方法。

前提条件

- 开通RTC服务后默认不支持小程序端能力，如果您需要使用小程序端RTC服务，请提交工单开通。
- 使用小程序端RTC服务，您需要开通旁路转推功能。具体操作，请参见开通旁路转推服务。

操作步骤

1. 下载并解压小程序SDK，下载地址请参见SDK下载。
2. 新建工程，将解压后的SDK文件复制到项目中。
3. 在项目中使用到SDK的文件中，对*aliyun-webrtc-miniapp-sdk.js*文件进行引用。

```
// pages/meeting/meeting.js
const app = getApp()
const Utils = require('../../utils/util.js')
const AliRtcMiniappSDK = require("../../sdk/aliyun-webrtc-miniapp-sdk");
const max_user = 10;
const Layouter = require("../../utils/layout.js");

const AUTO_LEAVE_STATE = true
const LEAVE_TIMEOUT = 10 * 60 * 1000;

Page({
  data: {
    media: [],
    mic: true,
    camera: true,
    beauty: 5,
    isPublish: false,
  },
```

> **说明** 此处SDK路径仅供参考，具体路径请以实际为准。

4.7. Web

通过阅读本文，您可以了解Web端集成SDK的方法。

注意事项

- 集成Web客户端SDK必须使用HTTPS协议。
- 因系统原因，setAudioVolume接口暂不支持在iOS中使用。
- 关于纯订阅模式媒体文件播放失败的问题，请参见H5纯订阅模式媒体文件播放失败。

环境要求

Web端具体环境要求，请参见使用限制。

操作步骤

1. 下载并解压Web SDK，下载地址请参见SDK下载。
2. 新建工程，将解压后的SDK文件复制到项目中。
3. 在项目相应的前端页面文件中，对*aliyun-webrtc-sdk.js*文件进行引用。

```
<link rel="stylesheet" href="./index.css" />
<script src="./jquery-1.10.2.min.js"></script>
<script src="./aliyun-webrtc-sdk-1.7.0.min.js"></script>
```

> 说明 此处SDK文件名称仅供参考，具体名称请以实际为准。

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能，详情请参见Web端实现基本功能。

4.8. Linux

4.8.1. C++

通过阅读本文，您可以了解Linux（C++）端集成SDK的方法。

环境要求

Linux端具体环境要求，更多信息，请参见使用限制。

集成SDK

1. 下载并解压Linux（C++） SDK，下载地址请参见SDK下载。
2. 引入头文件和库文件。
   - AliRtcCoreService：RTC引擎的可执行程序。请放置在有执行权限的路径下，并记录好路径地址。建议放在业务进程的当前路径下。
   - include：包含需要引入的头文件。
   - lib：待导入的SDK动态库，需要配置正确的动态库链接地址。

   ```
   export PROJECT_PATH=当前工程的目录
   export LD_LIBRARY_PATH=$PROJECT_PATH/lib
   ```
3. 编译。

   g++ -std=c++11 -g -Wall -O2 -o test Test_Demo.cc -lAliRTCLinuxEngine -I/$PROJECT_PATH/include -L/$PROJECT_PATH/lib

> 说明 Test_Demo.cc为业务实现代码文件，您需要替换成自己的代码文件。

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能，详情请参见Linux（C++）端实现基本功能。

4.8.2. Java

通过阅读本文，您可以了解Linux（Java）端集成SDK的方法。

环境要求

Java环境：Java 6及以上版本。更多Linux端环境要求，请参见使用限制。

集成SDK

1. 下载并解压Linux（Java） SDK，下载地址请参见SDK下载。
2. 引入库文件。
   - AliRtcCoreService：RTC引擎的可执行程序。请放置在有执行权限的路径下，并记录好路径地址。建议放在业务进程的当前路径下。
   - libs：业务执行程序必须依赖的jar和so库，需要配置正确的动态库链接地址。

   ```
   export PROJECT_PATH=当前工程的目录
   export LD_LIBRARY_PATH=$PROJECT_PATH/libs
   ```

3. 编译。

   javac -cp $PROJECT_PATH/libs/alirtc_linux_java.jar -encoding utf-8 Test_Demo.java

后续步骤

完成集成SDK操作后，您可以实现音视频通信的基本功能，详情请参见Linux（Java）实现基本功能。

5.实现基本功能

5.1. Android

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Android端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

使用限制

本文仅供SDK 2.1及以上版本使用，如果您需要SDK 1.17版本对应的文档，请参见Android端实现基本功能。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   您需要创建AliRtcEngine实例，并注册回调。具体回调接口请参见回调及监听。

```
private void getInstance() {
  AliRtcEngine engine = AliRtcEngine.getInstance(getApplicationContext());
  engine.setRtcEngineEventListener(eventListener);
  engine.setRtcEngineNotify(engineNotify);
}
```

i. 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas布局进行本地预览视频。

```
private void setLocalViewConfig() {
    // 创建 SurfaceView 对象。
    private FrameLayout mLocalContainer;
    private SurfaceView mLocalView;
    mLocalView = engine.CreateRendererView(getApplicationContext());
    mLocalView.setZOrderMediaOverlay(true);
    mLocalContainer.addView(mLocalView);
    // 设置本地视图。
    AliVideoCanvas canvas = new AliRtcEngine.AliVideoCanvas();
  canvas.view = surfaceView;
  canvas.renderMode = AliRtcRenderModeAuto;
    engine.setLocalViewConfig(canvas,AliRtcVideoTrackCamera);
    engine.startPreview();
}
```

> 说明
>
> - AliRtcRenderMode提供四种渲染模式：
>   - AliRtcRenderModeAuto（推荐）：自动。
>   - AliRtcRenderModeStretch：拉伸填充视图，不保持视频比例。
>   - AliRtcRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
>   - AliRtcRenderModeClip：在保持视频宽高比的同时缩放，并裁剪以适合视图。
> - AliRtcRenderMirrorMode在本地或远端均可设置镜像模式，并提供三种镜像模式：
>   - AliRtcRenderMirrorModeOnlyFront：只有前置摄像头预览镜像，其余不镜像。
>   - AliRtcRenderMirrorModeAllEnabled：全部镜像。
>   - AliRtcRenderMirrorModeAllDisable：全部不镜像。

ii. （可选）取消本地预览。

```
engine.stopPreview();
```

iii. 设置发布与订阅。

- SDK默认入会后自动发布音频流与视频流，如果您不希望自动发布音频和视频，可以在入会前通过以下接口设置：

```
engine.publishLocalAudioStream(false);//默认不发布音频流
engine.publishLocalVideoStream(false);//默认不发布视频流
```

- SDK默认入会后自动订阅远端的音频流与视频流，如果您不希望自动订阅音频与视频，可以在入会前通过以下接口设置：

```
engine.setDefaultSubscribeAllRemoteAudioStreams(false);//默认不订阅音频流
engine.setDefaultSubscribeAllRemoteVideoStreams(false);//默认不订阅视频流
```

2. 加入频道。

```
AliRtcAuthInfo userInfo = new AliRtcAuthInfo();
authinfo.channelId   = /* 您的channelId */;
authinfo.appId     = /* 您的Appid */;
authinfo.nonce     = /* 您的nonce */;
authinfo.userId    = /* 您的userId */;
authinfo.token     = /* 您的token */;
authinfo.timestamp = /* 您的timestamp */;
authinfo.gslb      = /* 您的gslb地址 */;
engine.joinChannel(userInfo, mUsername);
```

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地音频流和视频流

     如果您在入会前没有设置发布音频流和视频流，则入会后会自动发布本地的音频流和视频流；如果您在入会前设置取消发布音频流和视频流，则入会后需要调用以下接口进行手动发布：

     ```
     engine.publishLocalAudioStream(true);//发布音频流
     engine.publishLocalVideoStream(true);//发布视频流
     ```

   - 发布小流

     SDK默认不发布小流，如果您需要推送小流，请调用以下接口（入会前、后均可调用）：

     ```
     engine.publishLocalDualStream(true);
     ```

   - 取消发布本地音频流和视频流

     如果您需要取消发布本地的音频流和视频流，请调用以下接口：

     ```
     engine.publishLocalAudioStream(false);//取消发布音频流
     engine.publishLocalVideoStream(false);//取消发布视频流
     ```

4. 订阅或取消订阅远程流。
   - 订阅远端音频流和视频流

     如果您在入会前没有设置订阅音频流和视频流，则入会后会自动订阅远端的音频流和视频流；如果您在入会前设置取消订阅音频流和视频流，则入会后需要调用以下接口进行手动订阅：

     ```
     engine.setDefaultSubscribeAllRemoteAudioStreams(true);//订阅全部的远端音频流
     engine.setDefaultSubscribeAllRemoteVideoStreams(true);//订阅全部的远端视频流
     ```

     > **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

     订阅成功后，您可以在onRemoteTrackAvailableNotify回调里渲染远端的视频画面：

     ```
      public void onRemoteTrackAvailableNotify(String uid, AliRtcEngine.AliRtcAudioTrack a
     udioTrack,
                                              AliRtcEngine.AliRtcVideoTrack videoT
     rack) {
             // UI或者逻辑处理，例如渲染远端视频流的操作如下。
             if(videoTrack == AliRtcVideoTrackCamera) {
             // 视频流
             AliRtcEngine.AliRtcVideoCanvas canvas = new AliRtcEngine.AliRtcVideoCanvas();
             canvas.renderMode = /* renderMode */;
             canvas.view = view;/* 渲染view */
             engine.setRemoteViewConfig(canvas, uid, AliRtcVideoTrackCamera);
           }
         });
     }
     ```

   - 取消订阅远端音频流和视频流

     如果您需要取消订阅远端的音频流和视频流，请调用以下接口：

     ```
     engine.setDefaultSubscribeAllRemoteAudioStreams(false);//取消订阅远端音频流
     engine.setDefaultSubscribeAllRemoteVideoStreams:(false);//取消订阅远端视频流
     ```

     > **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

   - 订阅特定用户的音频流和视频流

     当已取消订阅所有的音频流和视频流之后，如果您需要订阅某个远端用户的音频流和视频流，可以通过调用以下接口实现（如果需要取消订阅此远端用户的音频流和视频流，参数sub传入false即可）：

     ```
     engine.subscribeRemoteAudioStream(uid,true);//订阅特定用户的音频流
     engine.subscribeRemoteVideoStream(uid,AliRtcVideoTrackCamera,true);//订阅特定用户的视频流
     ```

   - 订阅小流

     如果您需要订阅小流，请调用以下接口：

     ```
     engine.setRemoteDefaultVideoStreamType(AliRtcVideoStreamTypeLow);
     ```

> **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

- 取消订阅所有远端用户音频流和视频流

  如果您希望当前会议中及后续入会的用户全部取消订阅，请调用以下接口：

```
engine.subscribeAllRemoteAudioStreams(false);
engine.subscribeAllRemoteVideoStreams(false);
```

> **说明**
> - 当以上接口设置为false时，优先级最高，即使setDefaultSubscribeAllRemoteAudioStreams和setDefaultSubscribeAllRemoteVideoStreams设置为true时也不生效。
> - 当以上接口设置为true时，是否订阅以setDefaultSubscribeAllRemoteAudioStreams和setDefaultSubscribeAllRemoteVideoStreams设置为准。

5. 离开频道。

```
engine.leaveChannel();
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Android Demo。

5.2. iOS

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见iOS端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

使用限制

本文仅供SDK 2.1及以上版本使用，如果您需要SDK 1.17版本对应的文档，请参见iOS端实现基本功能。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   您需要创建AliRtcEngine实例，并注册回调。如果您在ViewController中持有AliRtcEngine实例，请声明属性。具体回调接口请参见回调及监听。

```
@interface ViewController () <AliRtcEngineDelegate>
@property (nonatomic, strong) AliRtcEngine *engine;
@end
```

```
self.engine = [AliRtcEngine sharedInstance:self extras:@""];
```

i. 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas布局进行本地预览视频。

```
AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
canvas.renderMode = AliRtcRenderModeAuto;
canvas.view = view; /* 预览窗口view，iOS为UIView对象，Mac为NSView对象*/
canvas.mirrorMode = AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled;
[self.engine setLocalViewConfig:canvas forTrack:AliRtcVideoTrackCamera];
[self.engine startPreview];
```

> 说明
>
> - AliRtcRenderMode提供四种渲染模式：
>   - AliRtcRenderModeAuto（推荐）：自动。
>   - AliRtcRenderModeStretch：拉伸填充视图，不保持视频比例。
>   - AliRtcRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
>   - AliRtcRenderModeCrop：在保持视频宽高比的同时缩放，并裁剪以适合视图。
> - AliRtcRenderMirrorMode在本地或远端均可设置镜像模式，并提供三种镜像模式：
>   - AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled：只有前置摄像头预览镜像，其余不镜像。
>   - AliRtcRenderMirrorModeAllEnabled：全部镜像。
>   - AliRtcRenderMirrorModeAllDisabled：全部不镜像。

ii. （可选）取消本地预览。

```
[self.engine stopPreview];
```

iii. 设置发布与订阅。

- SDK默认入会后自动发布音频流与视频流，如果您不希望自动发布音频和视频，可以在入会前通过以下接口设置：

```
[self.engine publishLocalAudioStream:NO];//默认不发布音频流
[self.engine publishLocalVideoStream:NO];//默认不发布视频流
```

- SDK默认入会后自动订阅远端的音频流与视频流，如果您不希望自动订阅音频与视频，可以在入会前通过以下接口设置：

```
[self.engine setDefaultSubscribeAllRemoteAudioStreams:NO];//默认不订阅音频流
[self.engine setDefaultSubscribeAllRemoteVideoStreams:NO];//默认不订阅视频流
```

2. 加入频道。

```
AliRtcAuthInfo *authinfo = [[AliRtcAuthInfo alloc]init];
authinfo.channelId   = /* 您的channelId */;
authinfo.appId     = /* 您的Appid */;
authinfo.nonce     = /* 您的nonce */;
authinfo.userId    = /* 您的userId */;
authinfo.token     = /* 您的token */;
authinfo.timestamp = /* 您的timestamp */;
authinfo.gslb      = /* 您的gslb地址 */;
[self.engine joinChannel:authinfo name:@"userName" onResult:^(NSInteger errCode,NSStrin
g * _Nonnull channel,NSInteger elapsed){
    // 加入频道UI处理
}];
```

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地音频流和视频流

     如果您在入会前没有设置发布音频流和视频流，则入会后会自动发布本地的音频流和视频流；如果您在入会前设置取消发布音频流和视频流，则入会后需要调用以下接口进行手动发布：

     ```
     [self.engine publishLocalAudioStream:YES];//发布音频流
     [self.engine publishLocalVideoStream:YES];//发布视频流
     ```

   - 发布小流

     SDK默认不发布小流，如果您需要推送小流，请调用以下接口（入会前、后均可调用）：

     ```
     [self.engine publishLocalDualStream:YES];
     ```

   - 取消发布本地音频流和视频流

     如果您需要取消发布本地的音频流和视频流，请调用以下接口：

```
[self.engine publishLocalAudioStream:NO];//取消发布音频流
[self.engine publishLocalVideoStream:NO];//取消发布视频流
```

4. 订阅或取消订阅远程流。
   - 订阅远端音频流和视频流

     如果您在入会前没有设置订阅音频流和视频流，则入会后会自动订阅远端的音频流和视频流；如果您在入会前设置取消订阅音频流和视频流，则入会后需要调用以下接口进行手动订阅：

     ```
     [self.engine setDefaultSubscribeAllRemoteAudioStreams:YES];//订阅全部的远端音频流
     [self.engine setDefaultSubscribeAllRemoteVideoStreams:YES];//订阅全部的远端视频流
     ```

     > **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

     订阅成功后，您可以在onRemoteTrackAvailableNotify回调里渲染远端的视频画面：

     ```
     - (void)onRemoteTrackAvailableNotify:(NSString *_Nonnull)uid audioTrack:(AliRtcAudioTrack)audioTrack videoTrack:(AliRtcVideoTrack)videoTrack {
         dispatch_async(dispatch_get_main_queue(), ^{
             // UI或者逻辑处理，例如渲染远端视频流的操作如下。
             if(videoTrack == AliRtcVideoTrackCamera) {
             // camera track
             AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
             canvas.renderMode = /* renderMode */;
             canvas.view = view;/* 渲染view */
             [self.engine setRemoteViewConfig:canvas uid:uid forTrack:AliRtcVideoTrackCamera];
           }
         });
     }
     ```

   - 取消订阅远端音频流和视频流

     如果您需要取消订阅远端的音频流和视频流，请调用以下接口：

     ```
     [self.engine setDefaultSubscribeAllRemoteAudioStreams:NO];//取消订阅全部的远端音频流
     [self.engine setDefaultSubscribeAllRemoteVideoStreams:NO];//取消订阅全部的远端视频流
     ```

     > **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

   - 订阅特定用户的音频流和视频流

     当已取消订阅所有的音频流和视频流之后，如果您需要订阅某个远端用户的音频流和视频流，可以通过调用以下接口实现（如果需要取消订阅此远端用户的音频流和视频流，参数sub传入NO即可）：

     ```
     [self.engine subscribeRemoteAudioStream:uid sub:YES];//订阅特定用户的音频流
     [self.engine subscribeRemoteVideoStream:uid track:AliRtcVideoTrackCamera sub:YES];//订阅特定用户的视频流
     ```

   - 订阅小流

     如果您需要订阅小流，请调用以下接口：

```
[self.engine setRemoteDefaultVideoStreamType:AliRtcVideoStreamTypeLow];
```

> **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

- 取消订阅所有远端用户音频流和视频流

  如果您希望当前会议中及后续入会的用户全部取消订阅，请调用以下接口：

```
[self.engine subscribeAllRemoteAudioStreams:NO];
[self.engine subscribeAllRemoteVideoStreams:NO];
```

> **说明**
>
> - 当以上接口设置为NO时，优先级最高，即使setDefaultSubscribeAllRemoteAudioStreams和setDefaultSubscribeAllRemoteVideoStreams设置为YES时也不生效。
> - 当以上接口设置为YES时，是否订阅以setDefaultSubscribeAllRemoteAudioStreams和setDefaultSubscribeAllRemoteVideoStreams设置为准。

5. 离开频道。

```
[self.engine leaveChannel];
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行iOS Demo。

5.3. Mac

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Mac端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

使用限制

本文仅供SDK 2.1及以上版本使用，如果您需要SDK 1.17版本对应的文档，请参见Mac端实现基本功能。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   您需要创建AliRtcEngine实例，并注册回调。如果您在ViewController中持有AliRtcEngine实例，请声明属性。具体回调接口请参见回调及监听。

```
@interface ViewController () <AliRtcEngineDelegate>
@property (nonatomic, strong) AliRtcEngine *engine;
@end
```

```
self.engine = [AliRtcEngine sharedInstance:self extras:@""];
```

i. 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas布局进行本地预览视频。

```
AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
canvas.renderMode = AliRtcRenderModeAuto;
canvas.view = view; /* 预览窗口view，iOS为UIView对象，Mac为NSView对象*/
canvas.mirrorMode = AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled;
[self.engine setLocalViewConfig:canvas forTrack:AliRtcVideoTrackCamera];
[self.engine startPreview];
```

> 说明
>
> - AliRtcRenderMode提供四种渲染模式：
>   - AliRtcRenderModeAuto（推荐）：自动。
>   - AliRtcRenderModeStretch：拉伸填充视图，不保持视频比例。
>   - AliRtcRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
>   - AliRtcRenderModeCrop：在保持视频宽高比的同时缩放，并裁剪以适合视图。
> - AliRtcRenderMirrorMode在本地或远端均可设置镜像模式，并提供三种镜像模式：
>   - AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled：只有前置摄像头预览镜像，其余不镜像。
>   - AliRtcRenderMirrorModeAllEnabled：全部镜像。
>   - AliRtcRenderMirrorModeAllDisabled：全部不镜像。

ii. （可选）取消本地预览。

```
[self.engine stopPreview];
```

iii. 设置发布与订阅。

- SDK默认入会后自动发布音频流与视频流，如果您不希望自动发布音频和视频，可以在入会前通过以下接口设置：

```
[self.engine publishLocalAudioStream:NO];//默认不发布音频流
[self.engine publishLocalVideoStream:NO];//默认不发布视频流
```

- SDK默认入会后自动订阅远端的音频流与视频流，如果您不希望自动订阅音频与视频，可以在入会前通过以下接口设置：

```
[self.engine setDefaultSubscribeAllRemoteAudioStreams:NO];//默认不订阅音频流
[self.engine setDefaultSubscribeAllRemoteVideoStreams:NO];//默认不订阅视频流
```

2. 加入频道。

```
AliRtcAuthInfo *authinfo = [[AliRtcAuthInfo alloc]init];
authinfo.channelId   = /* 您的channelId */;
authinfo.appId     = /* 您的Appid */;
authinfo.nonce     = /* 您的nonce */;
authinfo.userId    = /* 您的userId */;
authinfo.token     = /* 您的token */;
authinfo.timestamp = /* 您的timestamp */;
authinfo.gslb      = /* 您的gslb地址 */;
[self.engine joinChannel:authinfo name:@"userName" onResult:^(NSInteger errCode,NSStrin
g * _Nonnull channel,NSInteger elapsed){
    // 加入频道UI处理
}];
```

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地音频流和视频流

     如果您在入会前没有设置发布音频流和视频流，则入会后会自动发布本地的音频流和视频流；如果您在入会前设置取消发布音频流和视频流，则入会后需要调用以下接口进行手动发布：

     ```
     [self.engine publishLocalAudioStream:YES];//发布音频流
     [self.engine publishLocalVideoStream:YES];//发布视频流
     ```

   - 发布小流

     SDK默认不发布小流，如果您需要推送小流，请调用以下接口（入会前、后均可调用）：

     ```
     [self.engine publishLocalDualStream:YES];
     ```

   - 取消发布本地音频流和视频流

     如果您需要取消发布本地的音频流和视频流，请调用以下接口：

```
[self.engine publishLocalAudioStream:NO];//取消发布音频流
[self.engine publishLocalVideoStream:NO];//取消发布视频流
```

4. 订阅或取消订阅远程流。
   - 订阅远端音频流和视频流

     如果您在入会前没有设置订阅音频流和视频流，则入会后会自动订阅远端的音频流和视频流；如果您在入会前设置取消订阅音频流和视频流，则入会后需要调用以下接口进行手动订阅：

     ```
     [self.engine setDefaultSubscribeAllRemoteAudioStreams:YES];//订阅全部的远端音频流
     [self.engine setDefaultSubscribeAllRemoteVideoStreams:YES];//订阅全部的远端视频流
     ```

     > **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

     订阅成功后，您可以在onRemoteTrackAvailableNotify回调里渲染远端的视频画面：

     ```
     - (void)onRemoteTrackAvailableNotify:(NSString *_Nonnull)uid audioTrack:(AliRtcAudioTrack)audioTrack videoTrack:(AliRtcVideoTrack)videoTrack {
         dispatch_async(dispatch_get_main_queue(), ^{
             // UI或者逻辑处理，例如渲染远端视频流的操作如下。
             if(videoTrack == AliRtcVideoTrackCamera) {
             // camera track
             AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
             canvas.renderMode = /* renderMode */;
             canvas.view = view;/* 渲染view */
             [self.engine setRemoteViewConfig:canvas uid:uid forTrack:AliRtcVideoTrackCamera];
           }
         });
     }
     ```

   - 取消订阅远端音频流和视频流

     如果您需要取消订阅远端的音频流和视频流，请调用以下接口：

     ```
     [self.engine setDefaultSubscribeAllRemoteAudioStreams:NO];//取消订阅全部的远端音频流
     [self.engine setDefaultSubscribeAllRemoteVideoStreams:NO];//取消订阅全部的远端视频流
     ```

     > **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

   - 订阅特定用户的音频流和视频流

     当已取消订阅所有的音频流和视频流之后，如果您需要订阅某个远端用户的音频流和视频流，可以通过调用以下接口实现（如果需要取消订阅此远端用户的音频流和视频流，参数sub传入NO即可）：

     ```
     [self.engine subscribeRemoteAudioStream:uid sub:YES];//订阅特定用户的音频流
     [self.engine subscribeRemoteVideoStream:uid track:AliRtcVideoTrackCamera sub:YES];//订阅特定用户的视频流
     ```

   - 订阅小流

     如果您需要订阅小流，请调用以下接口：

```
[self.engine setRemoteDefaultVideoStreamType:AliRtcVideoStreamTypeLow];
```

> **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

- 取消订阅所有远端用户音频流和视频流

  如果您希望当前会议中及后续入会的用户全部取消订阅，请调用以下接口：

```
[self.engine subscribeAllRemoteAudioStreams:NO];
[self.engine subscribeAllRemoteVideoStreams:NO];
```

> **说明**
>
> - 当以上接口设置为NO时，优先级最高，即使setDefaultSubscribeAllRemoteAudioStreams和setDefaultSubscribeAllRemoteVideoStreams设置为YES时也不生效。
> - 当以上接口设置为YES时，是否订阅以setDefaultSubscribeAllRemoteAudioStreams和setDefaultSubscribeAllRemoteVideoStreams设置为准。

5. 离开频道。

```
[self.engine leaveChannel];
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Mac Demo。

5.4. Windows

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Windows端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

使用限制

本文仅供SDK 2.1及以上版本使用，如果您需要SDK 1.17版本对应的文档，请参见Windows端实现基本功能。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   您需要创建AliEngine实例，并注册回调。具体回调接口请参见AliEngineEventListener。

```
mpEngine = AliEngine::Create(pConfig);//pConfig是SDK初始化配置，当前版本请使用空字符串说明
mpEngine->SetEngineEventListener(pListener);//pListener是您实现的AliRtcEventListener对象
，用于接收SDK的回调
```

i. 本地预览。在创建完AliEngine实例后，您可以创建canvas布局进行本地预览视频。

```
AliEngineVideoCanvas canvas;
canvas.displayView = (void*)GetSafeHwnd();
canvas.renderMode = AliEngineRenderMode::AliEngineRenderModeAuto;
mpEngine->SetLocalViewConfig(canvas, AliEngineVideoTrackCamera);
mpEngine->StartPreview();
```

> 说明
>
> - AliEngineRenderMode提供五种渲染模式：
>   - AliEngineRenderModeAuto（推荐）：自动。
>   - AliEngineRenderModeStretch：拉伸填充视图，不保持视频比例。
>   - AliEngineRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
>   - AliEngineRenderModeCrop：在保持视频宽高比的同时缩放，并裁剪以适合视图。
>   - AliEngineRenderModeScroll：滚动视图以显示更多内容。
> - AliEngineRenderMirrorMode在本地或远端均可设置镜像模式，并提供三种镜像模式：
>   - AliEngineRenderMirrorModeOnlyFrontMirror：只有前置摄像头预览镜像，其余不镜像。
>   - AliEngineRenderMirrorModeAllMirror：全部镜像。
>   - AliEngineRenderMirrorModeAllNoMirror：全部不镜像。

ii. （可选）取消本地预览。

```
engine.stopPreview();
```

iii. 设置发布与订阅。

- SDK默认入会后自动发布音频流与视频流，如果您不希望自动发布音频和视频，可以在入会前通过以下接口设置：

```
mpEngine->PublishLocalAudioStream(false);
mpEngine->PublishLocalVideoStream(false);
```

- SDK默认入会后自动订阅远端的音频流与视频流，如果您不希望自动订阅音频与视频，可以在入会前通过以下接口设置：

```
mpEngine->SetDefaultSubscribeAllRemoteAudioStreams(false);
mpEngine->SetDefaultSubscribeAllRemoteVideoStreams(false);
```

2. 加入频道。

```
AliEngineAuthInfo authinfo;
authinfo.channelId   = /* 频道ID */;
authinfo.appId     = /* 应用ID */;
authinfo.token     = /* 频道鉴权令牌Token */;
authinfo.nonce     = /* 随机码 */;
authinfo.userId   = /* 用户ID */;
authinfo.timestamp = /* 时间戳 */;
    authinfo.gslbCount = 1;    /* GSLB地址个数 */;
authinfo.gslb = new char*[authinfo.gslbCount]; /* GSLB地址组 */;
    for (int i = 0; i < authinfo.gslbCount; i++)
    {
        authinfo.gslb[i] = "https://rgslb.rtc.aliyuncs.com";
    }
    authinfo.agentCount = 1;/* AGENT地址个数 */;
    authinfo.agent = new char*[authinfo.agentCount];/* AGENT地址组 */;
    for (int i = 0; i < authinfo.agentCount; i++)
    {
        authinfo.agent[i] = "example.com";
    }
m_pEngine->joinChannel(authinfo,userName /* 显示名称 */);
```

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地音频流和视频流

     如果您在入会前没有设置发布音频流和视频流，则入会后会自动发布本地的音频流和视频流；如果您在入会前设置取消发布音频流和视频流，则入会后需要调用以下接口进行手动发布：

     ```
     mpEngine->PublishLocalAudioStream(true);
     mpEngine->PublishLocalVideoStream(true);
     ```

   - 发布小流

SDK默认不发布小流，如果您需要推送小流，请调用以下接口（入会前、后均可调用）：

```
mpEngine->PublishLocalDualStream(true);
```

- 取消发布本地音频流和视频流

  如果您需要取消发布本地的音频流和视频流，请调用以下接口：

```
mpEngine->PublishLocalAudioStream(false);
mpEngine->PublishLocalVideoStream(false);
```

4. 订阅或取消订阅远程流。
   - 订阅远端音频流和视频流

     如果您在入会前没有设置订阅音频流和视频流，则入会后会自动订阅远端的音频流和视频流；如果您在入会前设置取消订阅音频流和视频流，则入会后需要调用以下接口进行手动订阅：

```
mpEngine->SetDefaultSubscribeAllRemoteAudioStreams(true);
mpEngine->SetDefaultSubscribeAllRemoteVideoStreams(true);
```

     说明 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

     订阅成功后，您可以在OnRemoteTrackAvailableNotify回调里渲染远端的视频画面：

```
AliEngineVideoCanvas canvas;
canvas.displayView = (void*)view->GetSafeHwnd();
canvas.mirrorMode = view->mCameraMirror ? AliEngineRenderMirrorModeAllMirror : AliEng
ineRenderMirrorModeAllNoMirror;
mpEngine->SetRemoteViewConfig(canvas, uid.c_str(), AliEngineVideoTrackCamera);
```

   - 取消订阅远端音频流和视频流

     如果您需要取消订阅远端的音频流和视频流，请调用以下接口：

```
mpEngine->SetDefaultSubscribeAllRemoteAudioStreams(false);
mpEngine->SetDefaultSubscribeAllRemoteVideoStreams(false);
```

     说明 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

   - 订阅特定用户的音频流和视频流

     当已取消订阅所有的音频流和视频流之后，如果您需要订阅某个远端用户的音频流和视频流，可以通过调用以下接口实现（如果需要取消订阅此远端用户的音频流和视频流，参数sub传入false即可）：

```
mpEngine->SubscribeRemoteAudioStream(uid.c_str(), true);
mpEngine->SubscribeRemoteVideoStream(uid.c_str(), AliEngineVideoTrackCamera, true);
```

   - 订阅小流

     如果您需要订阅小流，请调用以下接口：

```
mpEngine->SetRemoteDefaultVideoStreamType(AliEngineVideoStreamTypeLow);
```

> **说明** 此接口作用于后续入会的远端用户，对于调用此接口之前已经入会的远端用户，此接口不产生影响。

- 取消订阅所有远端用户音频流和视频流

  如果您希望当前会议中及后续入会的用户全部取消订阅，请调用以下接口：

```
mpEngine->SubscribeAllRemoteVideoStreams(false);
mpEngine->SubscribeAllRemoteAudioStreams(false);
```

> **说明**
>
> - 当以上接口设置为false时，优先级最高，即使SetDefaultSubscribeAllRemoteAudioStreams和SetDefaultSubscribeAllRemoteVideoStreams设置为true时也不生效。
> - 当以上接口设置为true时，是否订阅以SetDefaultSubscribeAllRemoteAudioStreams和SetDefaultSubscribeAllRemoteVideoStreams设置为准。

5. 离开频道。

```
m_pEngine->LeaveChannel();
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Windows Demo。

5.5. Electron

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Electron端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   创建AliRtcEngine实例。

```
var aliElectronRtc = new AliRtcEngine({AliEngineOption});
```

   本地预览。在创建完AliRtcEngine实例后，您可以通过div标签播放。

```
aliElectronRtc.setLocalViewConfig(
  HTMLDivElement    //HTML中的div元素
);//预览之前为本地预览设置渲染窗口以及绘制参数
aliElectronRtc.startPreview();
```

2. 加入频道。

```
aliElectronRtc.joinChannel(
  {
    channel,  //频道号
    userid,   //用户ID
    appid,    //应用ID
    nonce,    //令牌随机码
    token,    //令牌
    timestamp,   //时间戳
    gslb,      //服务器地址
    displayName   //用户名字
  },
  {
    autoSubscribeAudio:Boolean,  //自动订阅音频，默认true
    autoSubscribeVideo:Boolean   //自动订阅视频，默认true
  }
);
```

说明

- 加入频道成功后，如果中途需要加入其他频道，必须先调用leaveChannel离开当前频道。
- 如果加入频道失败，需要重试时无需再调用leaveChannel。
- 该接口是异步接口，是否成功加入频道，通过onJoinChannelResult判断。

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流。如果需要让远程订阅本地的流，您需要调用publishLocalVideoStream接口，发布本地流。

     ```
     aliElectronRtc.publishLocalVideoStream(
       enable   //boolean  是否允许发布视频流，true（默认值）：允许（发布）；false：不允许（取消发布）
     );
     ```

   - 取消发布本地流。

     ```
     aliElectronRtc.publishLocalAudioStream(
      enable   //boolean  是否允许发布音频流，true（默认值）：允许（发布）；false：不允许（取消发布）
     );
     ```

4. 订阅或取消订阅远程流。
   - 订阅和取消订阅所有远端视频流。

     ```
     aliElectronRtc.subscribeAllRemoteVideoStreams(
      sub //是否订阅 boolean
     );
     ```

   - 订阅和取消订阅指定远端视频流。

     ```
     aliElectronRtc.subscribeRemoteVideoStream(
      uid,   //远端用户ID
       track,  //视频流类型，大流或小流
       sub    //是否订阅
     );
     ```

   - 订阅和取消订阅所有远端音频流。

     ```
     aliElectronRtc.subscribeAllRemoteAudioStreams(
      sub   //是否订阅 boolean
     );
     ```

   - 订阅和取消订阅指定远端音频流。

     ```
     aliElectronRtc.subscribeRemoteAudioStream(
      uid,    //远端用户ID
      sub    //是否订阅 boolean
     );
     ```

5. 离开频道。

   ```
   aliElectronRtc.leaveChannel();
   ```

   > **说明** 该接口是异步接口，是否成功加入频道，通过onJoinChannelResult判断。

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Electron Demo。

5.6. Web

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Web端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   创建AliRtcEngine实例。

   ```
   var aliWebrtc = new AliRtcEngine();
   ```

   本地预览。在创建完AliRtcEngine实例后，您可以通过video标签播放。

   ```
   aliWebrtc.startPreview(
       video    // html中的video元素
   ).then(()=>{
   }).catch((error) => {
       // 预览失败
   });
   ```

2. 加入频道。

   ```
   aliWebrtc.joinChannel({
       userid,          // 用户ID
       channel,         // 频道
       appid,           // 应用ID
       nonce,           // 随机码
       timestamp,       // 时间戳
       gslb,            // gslb服务地址
       token,           // 令牌
   },displayName).then(()=>{
       // 入会成功
   } ,(error)=>{
       // 入会失败，打印错误内容，可以看到失败原因
       console.log(error.message);
   });
   ```

   > **说明** 加入频道需要传递两个参数。第一个是AliRtcAuthInfo，第二个参数是频道里显示的用户名。

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。

参数	描述
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)` 。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]` ，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流。如果需要让远程订阅本地的流，您需要调用publish接口，发布本地流，远程会接收到onPublisher回调。

```
aliWebrtc.publish().then(()=>{
} ,(error)=>{
    console.log(error.message);
});
```

   - 取消发布本地流。当您取消发布本地流时，远程会收到onUnPublisher回调。

```
aliWebrtc.unPublish().then(()=>{
} ,(error)=>{
    console.log(error.message);
});
```

4. 订阅onPublisher回调或onUnPublisher回调。
   - 订阅onPublisher回调。当远程用户推流时，在SDK里会触发onPublisher回调，通过订阅这个回调，能够得到频道里已经推流的用户。

```
aliWebrtc.on('onPublisher',(publisher) =>{
    //远程发布者userId
    console.log(publisher.userId);
    //远程发布名字
    console.log(publisher.displayName);
    //远程流内容，streamConfigs是数组格式
    console.log(publisher.streamConfigs);
  });
```

   - 订阅onUnPublisher回调。当远程用户结束推流时，会触发这个回调。

```
aliWebrtc.on('onUnPublisher',(publisher) =>{
    //远程发布者userId
    console.log(publisher.userId);
    //远程发布名字
    console.log(publisher.displayName);
});
```

> 说明 onPublisher、onUnPublisher回调只有加入频道以后才会触发。

5. 订阅或取消订阅远程流。
   - 订阅和显示远程流。通过subscribe方法订阅远程流，订阅成功后在调用setDisplayRemoteVideo显示远程流。

```
aliWebrtc.subscribe(userId).then((userId)=>{
    aliWebrtc.setDisplayRemoteVideo(
        userId,        // 用户ID
        video,         // html中用于显示stream对象的video元素
        1              // 1表示摄像头流（大流和小流），2表示屏幕分享流
    )
},(error)=>{
    console.log(error.message);
});
```

> 说明
> - Web SDK 1.10及以上版本不支持onMediaStream事件。
> - 音频流无需设置视图，订阅后可以自动播放。

   - 取消订阅。通过unSubscribe方法可以取消订阅远程流。

```
aliWebrtc.unSubscribe(userId).then(() => {
},(error)=>{
    console.log(error.message);
});
```

6. 离开频道。

```
aliWebrtc.leaveChannel().then(()=>{
} ,(error)=>{
    console.log(error.message);
});
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Web Demo。

5.7. Linux

5.7.1. C++

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Linux（C++）端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

```
//初始化SDK
AliRTCSdk::Linux::EngineEventHandlerInterface *linuxEventHandler = new AliRTCSdk::Linux
::FakeLinuxEventHandler();
AliRTCSdk::Linux::AliRTCEngineInterface *linuxEngine = (AliRTCSdk::Linux::AliRTCEngineI
nterface*)CreateAliRTCEngineInstance(linuxEventHandler, 42000, 45000, nullptr, nullptr)
;
// 如果创建失败，需要销毁EventHandler实例
if (linuxEngine == nullptr) {
    delete linuxEventHandler;
    linuxEventHandler = nullptr;
    return;
}
```

> **注意**
>
> - 创建一个SDK实例需要占用一个系统端口进行音视频数据传输，建议端口号范围设置为42000～45000，并保证其他服务不会占用此范围的端口。
> - 示例代码中传入的第四个参数表示log存放的路径，如果设置为nullptr，会默认放在*/tmp*路径下。
> - 示例代码中传入的第五个参数表示可执行程序AliRtcCoreService存放的绝对路径，如果设置为nullptr，会默认在当前路径下寻找。

2. 加入频道。

```
AliRTCSdk::Linux::AuthInfo authInfo;
authInfo.appid = "";              //应用ID
authInfo.channel = room.c_str();  //频道ID
authInfo.userid = "";             //用户ID
authInfo.username = "";           //用户名称
authInfo.nonce = "";              //随机码
authInfo.token = "";              //频道鉴权令牌
authInfo.timestamp = 1591430980;  //频道过期时间戳
int gslbCount = 1;
authInfo.gslb_count = gslbCount;
const char *gslbArray[gslbCount];
if (gslbCount > 0)
{
  for(int i = 0; i < gslbCount; i++)
  {
    gslbArray[i] = "*****";
  }
  authInfo.gslb = gslbArray;
}
int agentCount = 0;
authInfo.agent_count = agentCount;
const char *agentArray[agentCount];
if (agentCount > 0)
{
  for (int i = 0; i < agentCount; i++)
  {
    agentArray[i] = "";
  }
  authInfo.agent = agentArray;
}
// 初始化入会的设置
AliRTCSdk::Linux::JoinChannelConfig joinConfig;
// 关闭自动录制
joinConfig.recordingMode = AliRTCSdk::Linux::RecordingManually;
// 关闭自动推流
joinConfig.publishMode = AliRTCSdk::Linux::PublishManually;
// 入会
linuxEngine->JoinChannel(authInfo, joinConfig);
```

参数	描述
appid	应用ID，在控制台应用管理页面创建和查看。
channel	频道ID。1~64位，支持大小写字母、数字、下划线（_）、短划线（-）。
userid	用户ID。1~64位，支持大小写字母、数字、下划线（_）、短划线（-）。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。

参数	描述
username	用户名。
nonce	随机码。需要加上前缀AK-，由字母[a-zA-Z]和数字[0-9]组成，不包含特殊字符，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	频道过期时间戳。代表令牌有效时间为当前时间+所选择小时数。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channel + userId + nonce + timestamp)`。
gslb	gslb服务地址，该参数是数组类型，当前请使用：`["https://****"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。
agent	agent服务地址，该参数是数组类型，当前请使用：`["https://****"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流
     - 自动发布模式：加入频道成功后，即可发布本地流，无需再次调用publish接口。
     - 手动发布模式：加入频道成功后，可通过以下接口发布本地流。

     在发布Yuv和Pcm数据之前，需要设置推流的音视频参数。

```
// 开启Yuv输入，使用视频流（原相机流）进行推送
// 注意：第二个参数useTexture目前只能设置为false
linuxEngine->SetExternalVideoSource(true, false, AliRTCSdk::Linux::VideoSourceCamera);
// 开启Pcm输入
// 第二个参数为pcm的采样率，请根据实际情况设置
// 第三个参数为pcm的channel数，请根据实际情况设置
linuxEngine->SetExternalAudioSource(true, 16000, 2);
```

     如果发布过程中需要变更配置或者停止发布，需要按如下流程先重新设置配置参数，然后再调用publish接口。

     > **说明** Linux SDK通过原相机流和屏幕流通道传递视频流，Linux SDK不支持摄像头采集视频。

```
//设置是否推视频流（原相机流）。true表示允许发布，false表示不允许。
linuxEngine->ConfigLocalCameraPublish(true);
//设置是否推音频流。true表示允许发布，false表示不允许。
linuxEngine->ConfigLocalAudioPublish(true);
//设置是否推视频流（原屏幕流）。true表示允许发布，false表示不允许。
linuxEngine->ConfigLocalScreenPublish(true);
//设置是否推次要视频流。true表示允许发布，false表示不允许。
linuxEngine->ConfigLocalSimulcast(true, AliRTCSdk::Linux::VideoSourceCamera);
linuxEngine->publish();
while (true) {
    ...
   // 推Yuv数据
   linuxEngine->PushExternalVideoFrame(&sample, AliRTCSdk::Linux::VideoSourceCamera);
   // 推Pcm数据
   linuxEngine->PushExternalAudioFrameRawData(cache_buf, frame_length, 0);
}
```

- 取消发布本地流

```
// 设置是否推视频流（原相机流）
linuxEngine->ConfigLocalCameraPublish(false);
// 设置是否推音频流
linuxEngine->ConfigLocalAudioPublish(false);
// 设置是否推视频流（原屏幕流）
linuxEngine->ConfigLocalScreenPublish(false);
// 设置是否推次要视频流
linuxEngine->ConfigLocalSimulcast(false, AliRTCSdk::Linux::VideoSourceCamera);
linuxEngine->publish();
```

发布和取消发布本地流回调代码如下所示：

```
/**
* @brief 推流结果回调
* @param result 返回0表示成功，返回其他表示失败
* @param isPublished true表示推流成功，false表示停止推流
*/
virtual void OnPublishChangedNotify(int result, bool isPublished) = 0;
```

4. 录制和取消录制。
   - 录制
     - 自动录制模式：加入频道成功后，无需调用以下接口，即可开始录制。
     - 手动录制模式：加入频道成功后，可通过以下接口进行录制操作。

       ```
       linuxEngine->StartRecording();
       ```

   - 取消录制

     ```
     linuxEngine->StopRecording();
     ```

     录制回调示例代码如下所示：

```
/**
* @brief 远端用户上线回调
* @param uid 远端用户ID
*/
virtual void OnRemoteUserOnLineNotify(const char * uid) = 0;
/**
* @brief 远端用户下线回调
* @param uid 远端用户ID
*/
virtual void OnRemoteUserOffLineNotify(const char * uid) = 0;
/**
* @brief 音频原始数据的回调
* @param frame 音频原始数据
*/
virtual void OnAudioFrameReceived(const AliRTCSdk::Linux::AudioFrame * frame) = 0;
/**
* @brief 视频原始数据的回调
* @param uid 远端用户ID
* @param frame 视频原始数据
*/
virtual void OnVideoFrameReceived(const char * uid, const AliRTCSdk::Linux::VideoFrame
* frame) = 0;
```

5. 离开频道。

```
linuxEngine->LeaveChannel();
```

6. 销毁SDK。

```
linuxEngine->Release();
linuxEngine = nullptr;
```

5.7.2. Java

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Linux（Java）端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

```
AliRTCLinuxEngineListener engineEventHandler = new EngineListener();
AliRTCLinuxEngine linuxEngine = AliRTCLinuxEngine.createInstance(engineEventHandler, 42
000, 45000, null, CORE_SERVICE_PATH);
```

> 注意
>
> - 创建一个SDK实例需要占用一个系统端口进行音视频数据传输，建议端口号范围设置为42000～45000，并保证其他服务不会占用此范围的端口。
> - 示例代码中传入的第四个参数表示log存放的路径，如果设置为null，会默认放在*/tmp*路径下。
> - 示例代码中传入的第五个参数表示可执行程序AliRtcCoreService存放的绝对路径，如果设置为null，会默认在当前路径下寻找。

2. 加入频道。

```
// TODO by user, need authinfo
AliRTCLinuxEngine.AuthInfo authInfo = new AliRTCLinuxEngine.AuthInfo();
authInfo.appid = "";      //应用ID
authInfo.channel = "";    //频道ID
authInfo.userid = "";     //用户ID
authInfo.username = "";   //用户名称
authInfo.nonce = "";      //随机码
authInfo.token = "";      //频道鉴权令牌
authInfo.timestamp = 0;   //频道过期时间戳
int gslbCount = 1;
authInfo.gslb_count = gslbCount;
String[] gslbArray = new String[gslbCount];
if (gslbCount > 0) {
    for (int i = 0; i < gslbCount; i++) {
        gslbArray[i] = "https://example.com";
    }
    authInfo.gslb = gslbArray;
}
int agentCount = 0;
authInfo.agent_count = agentCount;
String[] agentArray = new String[agentCount];
if (agentCount > 0) {
    for (int i = 0; i < agentCount; i++) {
        agentArray[i] = "https://example.com";
    }
    authInfo.agent = agentArray;
}
// 初始化入会的设置
AliRTCLinuxEngine.JoinChannelConfig joinConfig = new AliRTCLinuxEngine.JoinChannelConfi
g();
// 关闭录制功能
joinConfig.recordingMode = AliRTCLinuxEngine.RecordingMode.RecordingManually;
// 关闭自动推流
joinConfig.publishMode = AliRTCLinuxEngine.PublishMode.PublishManually;
// 入会
linuxEngine.joinChannel(authInfo, joinConfig);
```

参数	描述
appid	应用ID，在控制台应用管理页面创建和查看。

参数	描述
channel	频道ID。1~64位，支持大小写字母、数字、下划线（_）、短划线（-）。
userid	用户ID。1~64位，支持大小写字母、数字、下划线（_）、短划线（-）。 说明 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
username	用户名。
nonce	随机码。需要加上前缀AK-，由字母[a-zA-Z]和数字[0-9]组成，不包含特殊字符，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	频道过期时间戳。代表令牌有效时间为当前时间+所选择小时数。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channel + userId + nonce + timestamp)`。
gslb	gslb服务地址，该参数是数组类型，当前请使用：`["https://****"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。
agent	agent服务地址，该参数是数组类型，当前请使用：`["https://****"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流
     - 自动发布模式：加入频道成功后，即可发布本地流，无需再次调用publish接口。
     - 手动发布模式：加入频道成功后，可通过以下接口发布本地流。

     在发布Yuv和Pcm数据之前，需要设置推流的音视频参数。

```
// 开启Yuv输入，使用相机流进行推送
// 注意：第二个参数useTexture目前只能设置为false
linuxEngine.setExternalVideoSource(true, false, AliRTCLinuxEngine.VideoSource.VideoSourceCamera, AliRTCLinuxEngine.RenderMode.RenderModeAuto);
// 开启Pcm输入
// 第二个参数为pcm的采样率，请根据实际情况设置
// 第三个参数为pcm的channel数，请根据实际情况设置
linuxEngine.setExternalAudioSource(true, 16000, 2);
```

     如果发布过程中需要变更配置或者停止发布，需要按如下流程先重新设置配置参数，然后再调用publish接口。

> **说明** Linux SDK通过原相机流和屏幕流通道传递视频流，Linux SDK不支持摄像头采集视频。

```
// 设置是否推相机流。true表示允许发布相机流，false表示不允许。
linuxEngine.configLocalCameraPublish(true);
// 设置是否推音频流。true表示允许发布音频流，false表示不允许。
linuxEngine.configLocalAudioPublish(true);
//设置是否推屏幕流。true表示允许发布屏幕流，false表示不允许。
linuxEngine.configLocalScreenPublish(true);
//设置是否推次要视频流。true表示允许发布次要视频流，false表示不允许。
linuxEngine.configLocalSimulcast(true);
linuxEngine.publish();
while (true) {
    ...
    // 推Yuv数据
    linuxEngine.pushExternalVideoFrame(sample, AliRTCLinuxEngine.VideoSource.VideoSou
rceCamera);
    // 推Pcm数据
    linuxEngine.pushExternalAudioFrameRawData(buf, frame_length, 0);
}
```

- 取消发布本地流

```
// 设置是否推相机流。
linuxEngine.configLocalCameraPublish(false);
// 设置是否推音频流
linuxEngine.configLocalAudioPublish(false);
// 设置是否推屏幕流
linuxEngine.configLocalScreenPublish(false);
// 设置是否推次要视频流。
linuxEngine.configLocalSimulcast(false, VideoTrackCamera);
linuxEngine.publish();
```

发布和取消发布本地流回调代码如下所示：

```
/**
* @brief 推流结果回调
* @param result 返回0表示成功，返回其他表示失败
* @param isPublished true表示推流成功，false表示停止推流
*/
void onPublishChangedNotify(int result, boolean isPublished);
```

4. 录制和取消录制。
   - 录制
     - 自动录制模式：加入频道成功后，无需调用以下接口，即可开始录制。
     - 手动录制模式：加入频道成功后，可通过以下接口进行录制操作。

       ```
       linuxEngine.startRecording();
       ```

   - 取消录制

     ```
     linuxEngine.stopRecording();
     ```

录制回调示例代码如下所示：

```
/**
* @brief 远端用户上线回调
* @param uid 远端用户ID
*/
void onRemoteUserOnLineNotify(String uid);
/**
* @brief 远端用户下线回调
* @param uid 远端用户ID
*/
void onRemoteUserOffLineNotify(String uid);
/**
* @brief 音频原始数据的回调
* @param frame 音频原始数据
*/
void onAudioFrameReceived(AliRTCLinuxEngine.AudioFrame frame);
/**
* @brief 视频原始数据的回调
* @param uid 远端用户ID,
* @param frame 视频原始数据
*/
void onVideoFrameReceived(String uid, AliRTCLinuxEngine.VideoFrame frame);
```

5. 离开频道。

```
linuxEngine.eaveChannel();
```

6. 销毁SDK。

```
linuxEngine.destroy();
linuxEngine = null;
```

6.实现基本功能（旧版）

6.1. Android

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Android端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   您需要创建AliRtcEngine实例，并注册回调。相关回调有AliRtcEngineEventListener和AliRtcEngineNotify，具体回调接口请参见回调及监听。

   ```
   mEngine = AliRtcEngine.getInstance(getApplicationContext());
   mEngine.setRtcEngineEventListener(mEventListener);
   mEngine.setRtcEngineNotify(mEngineNotify);
   ```

   > **说明** 该接口只能在主线程调用，目前暂不支持多实例。

   mEventListener：操作回调监听（回调接口都在子线程）。

   ```
   private AliRtcEngineEventListener mEventListener = new AliRtcEngineEventListener() {
   };
   ```

   mEngineNotify：SDK事件通知（回调接口都在子线程）。

   ```
   private AliRtcEngineNotify mEngineNotify = new AliRtcEngineNotify() {
   };
   ```

i. 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas布局进行本地预览视频。

```
//创建canvas，canvas为SophonSurfaceView或者它的子类。
AliRtcEngine.AliVideoCanvas canvas = new AliRtcEngine.AliVideoCanvas();
//SDK内部提供进行播放的视图。
SophonSurfaceView surfaceView = new SophonSurfaceView(this);
surfaceView.setZOrderOnTop(true);
surfaceView.setZOrderMediaOverlay(true);
mSurfaceContainer.addView(surfaceView,new ViewGroup.LayoutParams(ViewGroup.LayoutPa
rams.MATCH_PARENT,ViewGroup.LayoutParams.MATCH_PARENT));
/* 预览窗口的视图 */
canvas.view = surfaceView;
canvas.renderMode = AliRtcRenderModeAuto;
mEngine.setLocalViewConfig(canvas, AliRtcVideoTrackCamera);
mEngine.startPreview();
//mSurfaceContainer为包裹mCanvas的父视图。
mSurfaceContainer.getChildAt(0).setVisibility(View.VISIBLE);
```

建议您把mAliVideoCanvas之上的所有视图单独放在一个透明的父布局，因为surfaceview的z-order问题可能会挡住contronlview的显示。

> **说明**
>
> AliRtcRenderMirrorMode提供三种镜像模式。
>
> - AliRtcRenderMirrorModeOnlyFront：只有前置摄像头预览镜像，其余不镜像。
> - AliRtcRenderMirrorModeAllEnabled：全部镜像。
> - AliRtcRenderMirrorModeAllDisable：全部不镜像。
>
> AliRtcRenderMode渲染模式有四种模式。
>
> - AliRtcRenderModeAuto（推荐）：自动。
> - AliRtcRenderModeStretch：拉伸填充视图，不保持视频比例。
> - AliRtcRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
> - AliRtcRenderModeClip：在保持视频宽高比的同时缩放，并裁剪以适合视图。

ii. 设置自动或者手动模式。

> **说明** 默认实现自动发布和订阅，您也可以通过代码手动发布和订阅。

- 自动发布模式： 如果您打开自动发布模式，加入频道之后，SDK将自动开始发布音视频流；如果关闭自动发布模式，则需要您调用publish接口之后才会发布音视频流。
- 自动订阅模式： 如果您打开自动订阅模式，加入频道之后，SDK将会自动订阅当前频道内其他人的音视频流；如果关闭自动订阅模式，则需要您调用subscribe接口之后才会订阅其他人的音视频流。

```
/**
*设置自动发布和订阅，只能在加入频道之前设置。
*@param autoPub：是否自动发布。取值：true|false。
*@param autoSub：是否自动订阅。取值：true|false。
*/
mEngine.setAutoPublish(true, true);
```

2. 加入频道。

```
AliRtcAuthInfo userInfo = new AliRtcAuthInfo();
userInfo.setConferenceId(/* 频道ID */);
userInfo.setAppid(/* 应用ID */);
userInfo.setNonce(/* 随机码 */);
userInfo.setTimestamp(/* 时间戳*/);
userInfo.setUserId(/* 用户ID */);
userInfo.setGslb(/* GSLB地址*/);
userInfo.setToken(/*鉴权令牌Token*/);
if(mEngine != null) {
mEngine.joinChannel(userInfo, /* 用户显示名称 */);
}
```

参数	描述
AppID	应用ID，在控制台**应用管理**页面创建和查看。
ChannelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
UserId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
Nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
TimeStamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
Token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
GSLB	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流
     - 自动发布模式下：加入频道成功后，即可发布本地流，无需再次调用publish接口。
     - 手动发布模式下：加入频道成功后，可通过以下接口发布本地流。

     如果发布过程中需要变更配置或者停止发布，需要按如下流程先重新设置配置参数，然后再调用publish接口。

```
//发布本地流设置。
//true表示允许发布音频流，false表示不允许。
mEngine.configLocalAudioPublish(true);
//true表示允许发布相机流，false表示不允许。
mEngine.configLocalCameraPublish(true);
//true表示允许发布屏幕流，false表示不允许。
mEngine.configLocalScreenPublish(true);
//true表示允许发布次要视频流，false表示不允许。
mEngine.configLocalSimulcast(true, AliRtcEngine.AliRtcVideoTrack.AliRtcVideoTrackCame
ra);
mEngine.publish();
```

   - 取消发布本地流

```
mEngine.configLocalAudioPublish(false);
mEngine.configLocalCameraPublish(false);
mEngine.configLocalScreenPublish(false);
mEngine.configLocalSimulcast(false, AliRtcEngine.AliRtcVideoTrack.AliRtcVideoTrackCam
era);
mEngine.publish();
```

   发布和取消发布本地流回调代码如下所示：

```
private AliRtcEngineEventListener mEventListener = new AliRtcEngineEventListener() {
@Override
public void onPublishResult(int result, String publishId) {
//发布本地流回调。
}
@Override
public void onUnpublishResult(int result) {
//取消发布本地流回调。
}
}
```

4. 订阅或取消订阅远程流。
   - 订阅远程流
     - 自动订阅模式下：加入频道成功后，即可订阅远端流，无需再次调用subscribe接口。
     - 手动订阅模式下：加入频道成功后，可通过以下接口订阅远端流。

     如果订阅过程中需要变更配置或者停止订阅，需要按如下流程先重新设置配置参数，然后再调用subscribe接口。

```
// 订阅远端音频流。
mEngine.configRemoteAudio(/* remoteUserID */, true);
// 订阅远端屏幕流。
mEngine.configRemoteScreenTrack(/* remoteUserID */, true);
// 订阅远端相机流。
mEngine.configRemoteCameraTrack(/* remoteUserID */, true, true);
// 订阅远端用户ID。
mEngine.subscribe(/* remoteUserID */);
```

- 取消订阅远程流

```
mEngine.configRemoteAudio(/* remoteUserID */, false);
mEngine.configRemoteScreenTrack(/* remoteUserID */, false);
mEngine.configRemoteCameraTrack(/* remoteUserID */, true, false);
mEngine.subscribe(/* remoteUserID */);
```

远程流回调代码如下所示：

```
private AliRtcEngineNotify mEngineNotify = new AliRtcEngineNotify() {
@Override
public void onRemoteUserUnPublish(AliRtcEngine rtcEngine, String userId) {
//远端用户停止发布通知，处于OB（observer）状态。
}
@Override
public void onRemoteUserOnLineNotify(String uid) {
//远端用户上线通知。
}
@Override
public void onRemoteUserOffLineNotify(String uid) {
//远端用户下线通知。
}
@Override
public void onRemoteTrackAvailableNotify(String uid,AliRtcEngine.AliRtcAudioTrackaudioT
rack,AliRtcEngine.AliRtcVideoTrack videoTrack) {
//远端用户发布音视频流变化通知。
}
public void onSubscribeResult(String uid,int result,AliRtcVideoTrack videoTrack,AliRtcA
udioTrack audioTrack) {
//订阅流回调，可以做UI及数据的更新。
}
}
```

5. 离开频道。
   - 如果SDK版本号大于1.7，请调用如下接口。

```
mEngine.leaveChannel();
```

   - 如果SDK版本号小于等于1.7，请增加timeout参数，一般设置为1000，表示该接口的调用超时时间为1秒，建议您在Activity的onDestroy接口中调用。

> **说明** 调用leaveChannel接口后请不要再操作AliRtcEngine实例。

```
mEngine.leaveChannel(1000);
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Android Demo。

6.2. iOS

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见iOS端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   您需要创建AliRtcEngine实例，并注册AliRtcEngineDelegate监听相关回调。如果您在ViewController中持有AliRtcEngine实例，请声明属性。

   ```
   @interface ViewController () <AliRtcEngineDelegate>
   @property (nonatomic, strong) AliRtcEngine *engine;
   @end
   ```

   回调详情请参见回调及监听。

   ```
   self.engine = [AliRtcEngine sharedInstance:self extras:@""];
   ```

   > **说明** 目前暂不支持多实例。

i. 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas布局进行本地预览视频。

```
AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
canvas.renderMode = AliRtcRenderModeAuto;
canvas.view = (AliRenderView *)view; /* 预览窗口view */
canvas.mirrorMode = AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled;
[self.engine setLocalViewConfig:canvas forTrack:AliRtcVideoTrackCamera];
[self.engine startPreview];
```

> **说明**
>
> - AliRtcRenderMode提供四种渲染模式。
>   - （推荐）AliRtcRenderModeAuto：自动模式。
>   - AliRtcRenderModeStretch：拉伸填充视图，不保持视频比例。
>   - AliRtcRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
>   - AliRtcRenderModeCrop：在保持视频宽高比的同时缩放，并裁剪以适合视图。
> - view必须是AliRenderView或者其子类。
> - AliRtcRenderMirrorMode在本地或远端均可设置镜像模式，并提供三种镜像模式。
>   - AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled：只有前置摄像头预览镜像，其余不镜像。
>   - AliRtcRenderMirrorModeAllEnabled：全部镜像。
>   - AliRtcRenderMirrorModeAllDisabled：全部不镜像。

您也可以取消本地预览。

```
[self.engine stopPreview];
```

ii. 设置发布与订阅。

> **说明** 默认实现自动发布和订阅，您也可以通过代码手动发布和订阅。

- 自动发布模式： 如果您打开自动发布模式，加入频道之后，SDK将自动开始发布音视频流；如果关闭自动发布模式，则需要您调用publish接口之后才会发布音视频流。
- 自动订阅模式： 如果您打开自动订阅模式，加入频道之后，SDK将会自动订阅当前频道内其他人的音视频流；如果关闭自动订阅模式，则需要您调用subscribe接口之后才会订阅其他人的音视频流。

```
/*
设置自动发布和订阅，只能在加入频道之前配置。
autoPublish：是否自动发布。取值：YES|NO。
autoSubscribe：是否自动订阅。取值：YES|NO。
*/
[self.engine setAutoPublish:YES withAutoSubscribe:YES];
```

2. 加入频道。

```
AliRtcAuthInfo *authinfo = [[AliRtcAuthInfo alloc]init];
authinfo.channel   = /* 您的channelId */;
authinfo.appid     = /* 您的Appid */;
authinfo.nonce     = /* 您的nonce */;
authinfo.user_id   = /* 您的userId */;
authinfo.token     = /* 您的token */;
authinfo.timestamp = /* 您的timestamp */;
authinfo.gslb      = /* 您的gslb地址 */;
[self.engine joinChannel:authinfo name:/* userName */ onResult:^(NSInteger errCode){
    // 加入频道UI处理
}];
```

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流
     - 自动发布模式下：加入频道成功后，即可发布本地流，无需再次调用publish接口。
     - 手动发布模式下：加入频道成功后，可通过以下接口发布本地流。

     如果发布过程中需要变更配置或者停止发布，需要按如下流程先重新设置配置参数，然后再调用publish接口。

     ```
     //YES表示允许发布音频流，NO表示不允许。
     [self.engine configLocalAudioPublish:YES];
     //YES表示允许发布相机流，NO表示不允许。
     [self.engine configLocalCameraPublish:YES];
     //YES表示允许发布次要视频流；NO表示不允许。
     [self.engine configLocalSimulcast:YES forTrack:AliRtcVideoTrackCamera];
     [self.engine publish:^(int err) {
     }];
     ```
   - 取消发布本地流

```
[self.engine configLocalAudioPublish:NO];
[self.engine configLocalCameraPublish:NO];
[self.engine configLocalSimulcast:NO forTrack:AliRtcVideoTrackCamera];
[self.engine publish:^(int err) {
}];
```

4. 订阅或取消订阅远程流。
   - 订阅远程流
     - 自动订阅模式下：加入频道成功后，即可订阅远端流，无需再次调用subscribe接口。
     - 手动订阅模式下：加入频道成功后，可通过以下接口订阅远端流。

     如果订阅过程中需要变更配置或者停止订阅，需要按如下流程先重新设置配置参数，然后再调用subscribe接口。

     ```
     //YES表示允许订阅音频流，NO表示不允许。
     [self.engine configRemoteAudio:/* remoteUserID */ enable:YES];
     //YES表示允许订阅屏幕流，NO表示不允许。
     [self.engine configRemoteScreenTrack:/* remoteUserID */ enable:YES];
     //第二个参数：YES表示优先拉取大流。
     //第三个参数：YES表示允许订阅相机流，NO表示不允许。
     [self.engine configRemoteCameraTrack:/* remoteUserID */ preferMaster:YES enable:YES];
     [self.engine subscribe:/* remoteUserID */ onResult:^(NSString *uid, AliRtcVideoTrack
     vt, AliRtcAudioTrack at) {
     }];
     ```

     无论是自动模式还是非自动模式，当您订阅成功后，通过delegate可以获取订阅的callback，然后您可以进行相关UI操作或逻辑处理。

     ```
     - (void)onSubscribeChangedNotify:(NSString *)uid audioTrack:(AliRtcAudioTrack)audioTr
     ack videoTrack:(AliRtcVideoTrack)videoTrack {
         dispatch_async(dispatch_get_main_queue(), ^{
             // UI或者逻辑处理，例如渲染远端视频流的操作如下。
             if(videoTrack & AliRtcVideoTrackCamera) {
             // camera track
             AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
             canvas.renderMode = /* renderMode */;
             canvas.view = (AliRenderView *)view;/* 渲染view */
             [self.engine setRemoteViewConfig:canvas uid:uid forTrack:AliRtcVideoTrackCame
     ra];
           }
         });
     }
     ```

     您可以通过下述delegate回调监听远端用户的流状态变更。例如，手动模式下，收到此回调后，可以获取到远端用户的发布状态，然后相应做出订阅（subscribe）操作，或者更新UI等。

     ```
     - (void)onRemoteTrackAvailableNotify:(NSString *)uid audioTrack:(AliRtcAudioTrack)aud
     ioTrack videoTrack:(AliRtcVideoTrack)videoTrack {
     }
     ```

   - 取消订阅远程流

```
[self.engine configRemoteAudio:/* remoteUserID */ enable:NO];
[self.engine configRemoteScreenTrack:/* remoteUserID */ enable:NO];
[self.engine configRemoteCameraTrack:/* remoteUserID */ preferMaster:YES enable:NO];
[self.engine subscribe:/* remoteUserID */ onResult:^(NSString *uid, AliRtcVideoTrack
vt, AliRtcAudioTrack at) {
}];
```

5. 离开频道。

```
[self.engine leaveChannel];
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行iOS Demo。

6.3. Mac

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Mac端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。

   您需要创建AliRtcEngine实例，并注册AliRtcEngineDelegate监听相关回调。如果您在ViewController中持有AliRtcEngine实例，请声明属性。

   ```
   @interface ViewController () <AliRtcEngineDelegate>
   @property (nonatomic, strong) AliRtcEngine *engine;
   @end
   ```

   Mac回调详情请参见回调及监听。

   ```
   self.engine = [AliRtcEngine sharedInstance:self extras:@""];
   ```

   > **说明** 目前暂不支持多实例。

i. 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas布局进行本地预览视频。

```
AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
canvas.renderMode = AliRtcRenderModeAuto;
canvas.view = (AliRenderView *)view; /* 预览窗口view */
canvas.mirrorMode = AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled;
[self.engine setLocalViewConfig:canvas forTrack:AliRtcVideoTrackCamera];
[self.engine startPreview];
```

> **说明**
>
> - AliRtcRenderMode提供四种渲染模式。
>   - （推荐）AliRtcRenderModeAuto：自动模式。
>   - AliRtcRenderModeStretch：拉伸填充视图，不保持视频比例。
>   - AliRtcRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
>   - AliRtcRenderModeCrop：在保持视频宽高比的同时缩放，并裁剪以适合视图。
> - view必须是AliRenderView或者其子类。
> - AliRtcRenderMirrorMode在本地或远端均可设置镜像模式，并提供三种镜像模式。
>   - AliRtcRenderMirrorModeOnlyFrontCameraPreviewEnabled：只有前置摄像头预览镜像，其余不镜像。
>   - AliRtcRenderMirrorModeAllEnabled：全部镜像。
>   - AliRtcRenderMirrorModeAllDisabled：全部不镜像。

您也可以取消本地预览。

```
[self.engine stopPreview];
```

ii. 设置发布与订阅。

> **说明** 默认实现自动发布和订阅，您也可以通过代码手动发布和订阅。

- 自动发布模式： 如果您打开自动发布模式，加入频道之后，SDK将自动开始发布音视频流；如果关闭自动发布模式，则需要您调用publish接口之后才会发布音视频流。
- 自动订阅模式： 如果您打开自动订阅模式，加入频道之后，SDK将会自动订阅当前频道内其他人的音视频流；如果关闭自动订阅模式，则需要您调用subscribe接口之后才会订阅其他人的音视频流。

```
/*
设置自动发布和订阅，只能在加入频道之前配置。
autoPublish：是否自动发布。取值：YES|NO。
autoSubscribe：是否自动订阅。取值：YES|NO。
*/
[self.engine setAutoPublish:YES withAutoSubscribe:YES];
```

2. 加入频道。

```
AliRtcAuthInfo *authinfo = [[AliRtcAuthInfo alloc]init];
authinfo.channel   = /* 您的channelId */;
authinfo.appid     = /* 您的Appid */;
authinfo.nonce     = /* 您的nonce */;
authinfo.user_id   = /* 您的userId */;
authinfo.token     = /* 您的token */;
authinfo.timestamp = /* 您的timestamp */;
authinfo.gslb      = /* 您的gslb地址 */;
[self.engine joinChannel:authinfo name:/* userName */ onResult:^(NSInteger errCode){
    // 加入频道UI处理
}];
```

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流
     - 自动发布模式下：加入频道成功后，即可发布本地流，无需再次调用publish接口。
     - 手动发布模式下：加入频道成功后，可通过以下接口发布本地流。

     如果发布过程中需要变更配置或者停止发布，需要按如下流程先重新设置配置参数，然后再调用publish接口。

     ```
     //YES表示允许发布音频流，NO表示不允许。
     [self.engine configLocalAudioPublish:YES];
     //YES表示允许发布相机流，NO表示不允许。
     [self.engine configLocalCameraPublish:YES];
     //YES表示允许发布次要视频流；NO表示不允许。
     [self.engine configLocalSimulcast:YES forTrack:AliRtcVideoTrackCamera];
     [self.engine publish:^(int err) {
     }];
     ```
   - 取消发布本地流

```
[self.engine configLocalAudioPublish:NO];
[self.engine configLocalCameraPublish:NO];
[self.engine configLocalSimulcast:NO forTrack:AliRtcVideoTrackCamera];
[self.engine publish:^(int err) {
}];
```

4. 订阅或取消订阅远程流。
   - 订阅远程流
     - 自动订阅模式下：加入频道成功后，即可订阅远端流，无需再次调用subscribe接口。
     - 手动订阅模式下：加入频道成功后，可通过以下接口订阅远端流。

     如果订阅过程中需要变更配置或者停止订阅，需要按如下流程先重新设置配置参数，然后再调用subscribe接口。

     ```
     //YES表示允许订阅音频流，NO表示不允许。
     [self.engine configRemoteAudio:/* remoteUserID */ enable:YES];
     //YES表示允许订阅屏幕流，NO表示不允许。
     [self.engine configRemoteScreenTrack:/* remoteUserID */ enable:YES];
     //第二个参数：YES表示优先拉取大流。
     //第三个参数：YES表示允许订阅相机流，NO表示不允许。
     [self.engine configRemoteCameraTrack:/* remoteUserID */ preferMaster:YES enable:YES];
     [self.engine subscribe:/* remoteUserID */ onResult:^(NSString *uid, AliRtcVideoTrack
     vt, AliRtcAudioTrack at) {
     }];
     ```

     无论是自动模式还是非自动模式，当您订阅成功后，通过delegate可以获取订阅的callback，然后您可以进行相关UI操作或逻辑处理。

     ```
     - (void)onSubscribeChangedNotify:(NSString *)uid audioTrack:(AliRtcAudioTrack)audioTr
     ack videoTrack:(AliRtcVideoTrack)videoTrack {
         dispatch_async(dispatch_get_main_queue(), ^{
             // UI或者逻辑处理，例如渲染远端视频流的操作如下。
             if(videoTrack & AliRtcVideoTrackCamera) {
             // camera track
             AliVideoCanvas *canvas = [[AliVideoCanvas alloc] init];
             canvas.renderMode = /* renderMode */;
             canvas.view = (AliRenderView *)view;/* 渲染view */
             [self.engine setRemoteViewConfig:canvas uid:uid forTrack:AliRtcVideoTrackCame
     ra];
           }
         });
     }
     ```

     您可以通过下述delegate回调监听远端用户的流状态变更。例如，手动模式下，收到此回调后，可以获取到远端用户的发布状态，然后相应做出订阅（subscribe）操作，或者更新UI等。

     ```
     - (void)onRemoteTrackAvailableNotify:(NSString *)uid audioTrack:(AliRtcAudioTrack)aud
     ioTrack videoTrack:(AliRtcVideoTrack)videoTrack {
     }
     ```

   - 取消订阅远程流

```
[self.engine configRemoteAudio:/* remoteUserID */ enable:NO];
[self.engine configRemoteScreenTrack:/* remoteUserID */ enable:NO];
[self.engine configRemoteCameraTrack:/* remoteUserID */ preferMaster:YES enable:NO];
[self.engine subscribe:/* remoteUserID */ onResult:^(NSString *uid, AliRtcVideoTrack
vt, AliRtcAudioTrack at) {
}];
```

5. 离开频道。

```
[self.engine leaveChannel];
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Mac Demo。

6.4. Windows

阿里云RTC的基本功能包含初始化SDK、加入频道、本地发布、订阅远端和离开频道等。通过阅读本文，您可以了解阿里云RTC的基本功能。

前提条件

- 您已下载并集成最新版本的SDK。具体操作，请参见Windows端集成SDK。
- 您已获取加入频道必需的频道鉴权令牌（Token）。具体操作，请参见使用Token鉴权。

操作步骤

> **说明** 本文中的实现方法仅供参考，您可以根据实际业务需求进行开发。

1. 初始化SDK。
   i. 如果CRtcSampleDlg类中有定义AliRTCEngine实例，则需要创建AliRTCEngine实例，并注册AliRtcEventListener监听相关回调。更多信息，请参见回调及监听。

```
m_pEngine = AliRtcEngine::sharedInstance(AliRtcEventListener* pListener, char* pCon
fig);//pListener是您实现的AliRtcEventListener对象，用于接收SDK的回调。pConfig是SDK初始化
配置，当前版本请使用空字符串
```

> **说明** 目前暂不支持多实例。

ii. 创建canvas布局进行本地预览视频。

- 开启本地预览。

```
// 获取预览窗口
AliVideoCanvas canvas;
canvas.renderMode = AliRtcRenderModeAuto;
canvas.hWnd = /*预览窗口句柄*/;
// 设置预览窗口
m_pEngine->setLocalViewConfig(canvas, AliRtcVideoTrackCamera);
m_pEngine->startPreview();
canvas.flip = true;//true表示镜像画面，false表示正常画面
```

> **说明** AliRtcRenderMode提供四种渲染模式。
>
> - （推荐）AliRtcRenderModeAuto：自动模式。
> - AliRtcRenderModeStretch：拉伸填充视图，不保持视频比例。
> - AliRtcRenderModeFill：在保持视频宽高比的同时缩放，填充黑边。
> - AliRtcRenderModeCrop：在保持视频宽高比的同时缩放，并裁剪以适合视图。

- 停止本地预览。

```
m_pEngine->stopPreview();
```

iii. 设置发布与订阅。

> **说明** 默认实现自动发布和订阅，您也可以通过代码手动发布和订阅。

- 自动发布模式： 如果您打开自动发布模式，加入频道之后，SDK将自动开始发布音视频流；如果关闭自动发布模式，则需要您调用publish接口之后才会发布音视频流。
- 自动订阅模式： 如果您打开自动订阅模式，加入频道之后，SDK将会自动订阅当前频道内其他人的音视频流；如果关闭自动订阅模式，则需要您调用subscribe接口之后才会订阅其他人的音视频流。

```
/*
设置自动发布和订阅，只能在joinChannel之前设置
autoPub: true表示自动发布，false表示手动发布
autoSub: true表示自动订阅，false表示手动订阅
*/
m_pEngine->setAutoPublishSubscribe(bool autoPub, bool autoSub);
```

2. 加入频道。

```
AliRtcAuthInfo authinfo;
authinfo.channel   = /* 频道ID */;
authinfo.appid     = /* 应用ID */;
authinfo.token     = /* 频道鉴权令牌Token */;
authinfo.nonce     = /* 随机码 */;
authinfo.user_id   = /* 用户ID */;
authinfo.timestamp = /* 时间戳 */;
authinfo.gslb      = /* GSLB地址 */;
...
// joinChannel是一个异步接口，设置回调函数
auto onJoinResult = [](void *opaque, int errCode) {
    if (errCode == 0) {
        // 加入频道成功
    } else {
        //加入频道失败
    }
};
m_pEngine->joinChannel(authinfo, /* 显示名称 */, onJoinResult, /*UserData，传递给onJoinResult回调函数*/);
```

参数	描述
appId	应用ID，在控制台**应用管理**页面创建和查看。
channelId	频道ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。
userId	用户ID。1~64位，由大小写字母、数字、下划线（_）、短划线（-）组成。 **说明** 同一个用户ID在其他端登录，先入会的端会被后入会的端踢出频道。
nonce	随机码。以前缀AK-开头，由大小写字母、数字组成，最大64字节。例如：AK-2b9be4b25c2d38c409c376ffd2372be1。
timestamp	过期时间戳。可以选择12小时、24小时、3天和7天，代表令牌有效时间。
token	频道鉴权令牌，计算方法：`token = sha256(appId + appKey + channelId + userId + nonce + timestamp)`。
gslb	服务地址，该参数是数组类型，当前请使用：`["https://rgslb.rtc.aliyuncs.com"]`，请您通过业务服务器下发到客户端SDK，不建议您将该地址固化在客户端代码。

3. 发布或取消发布本地流。
   - 发布本地流
     - 自动发布模式下：加入频道成功后，即可发布本地流，无需再次调用publish接口。
     - 手动发布模式下：加入频道成功后，可通过以下接口发布本地流。

     如果发布过程中需要变更配置或者停止发布，需要按如下流程先重新设置配置参数，然后再调用publish接口。

```
//发布本地流设置
//true表示允许发布屏幕共享流，false表示不允许
m_pEngine->configLocalScreenPublish(true);
//true表示允许发布音频流，false表示不允许
m_pEngine->configLocalAudioPublish(true);
//true表示允许发布相机流，false表示不允许
m_pEngine->configLocalCameraPublish(true);
//true表示允许发布次要视频流；false表示不允许
//子码流
m_pEngine->configLocalSimulcast(true, AliRtcVideoTrack::AliRtcVideoTrackCamera);
//屏幕共享流不支持子码流，因此第二个参数只能为AliRtcVideoTrack::AliRtcVideoTrackCamera。
// Call back when publish finished
auto onPubResult = [](void *opaque, int errCode) {
    if (errCode == 0) {
       // 发布成功
    } else {
       // 发布失败
    }
};
m_pEngine->publish(onPubResult, /*UserData，传递给onPubResult回调函数*/);
```

- 取消发布本地流

```
m_pEngine->configLocalScreenPublish(false);
m_pEngine->configLocalAudioPublish(false);
m_pEngine->configLocalCameraPublish(false);
m_pEngine->configLocalSimulcast(false, AliRtcVideoTrack::AliRtcVideoTrackCamera);
// 取消发布完成返回成功或者失败
auto onPubResult = [](void *opaque, int errCode) {
    if (errCode == 0) {
       // 取消发布成功
    } else {
       // 取发布失败
    }
};
m_pEngine->publish(onPubResult, /*UserData，传递给onPubResult回调函数*/);
```

4. 订阅或取消订阅远程流。
   - 订阅远程流
     - 自动订阅模式下：加入频道成功后，即可订阅远端流，无需再次调用subscribe接口。
     - 手动订阅模式下：加入频道成功后，可通过以下接口订阅远端流。

     如果订阅过程中需要变更配置或者停止订阅，需要按如下流程先重新设置配置参数，然后再调用subscribe接口。

```
//true表示允许订阅音频流，false表示不允许
m_pEngine->configRemoteAudio(/* remoteUserID */, true);
//true表示允许订阅屏幕共享流，false表示不允许
m_pEngine->configRemoteScreenTrack(/* remoteUserID */, true);
//第二个参数preferMaster true表示优先订阅大流，fales表示优先订阅次小流
//第三个参数enable true表示允许订阅相机流，false表示不允许
m_pEngine->configRemoteCameraTrack(/* remoteUserID */, true, true);
//Call back when subscribe finished
auto onSubResult = [](void *opaque, const AliRtc::String &uid, AliRtcVideoTrack vt, A
liRtcAudioTrack at) {
};
m_pEngine->subscribe(/* remoteUserID */, onSubResult, /*UserData*/);
```

无论是自动模式还是非自动模式，当您订阅成功后，通您可以通过订阅的回调，进行相关UI操作或逻辑处理。

```
auto onSubResult = [](void *opaque, const AliRtc::String &uid, AliRtcVideoTrack vt, A
liRtcAudioTrack at) {
    ...
    // 设置视频流的窗口
    if (vt == AliRtcVideoTrack::AliRtcVideoTrackCamera || vt == AliRtcVideoTrack::Ali
RtcVideoTrackBoth) {
        /*设置远端视频预览*/
    }
```

另外，您可以通过onRemoteTrackAvailableNotify回调获得远端用户的流状态变更。例如：手动模式下，收到此回调后，可以获取到远端用户的发布状态，然后相应做出订阅操作，或者更新UI等。

```
 void onRemoteTrackAvailableNotify(const AliRtc::String & uid,   AliRtcAudioTrack aud
ioTrack, AliRtcVideoTrack videoTrack) {
}
```

- 取消订阅远程流

```
m_pEngine->configRemoteAudio(/* remoteUserID */, false);
m_pEngine->configRemoteScreenTrack(/* remoteUserID */, false);
m_pEngine->configRemoteCameraTrack(/* remoteUserID */, true, false);
auto onSubResult = [](void *opaque, const AliRtc::String &uid, AliRtcVideoTrack vt, A
liRtcAudioTrack at) {
};
m_pEngine->subscribe(/* remoteUserID */, onSubResult, /*UserData*/);
```

5. 离开频道。

```
m_pEngine->leaveChannel();
```

后续步骤

您可以下载示例代码，快速运行Demo，实现频道内和其他人进行实时音视频通话，详情请参见运行Windows Demo。