阿里云
音视频通信

快速入门

文档版本：20190917

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
	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。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能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，恢复业务所需时间约10分钟。
	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。	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。	单击 确定。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	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。	`swich` *`{stand | slave}`*

目录

1 入门概述

本文档为您介绍了开发音视频通信的前提条件和操作步骤。操作步骤包括了创建应用、搭建App Server、集成客户端SDK、实现基本功能、应用管理五个方面，通过本教程，您可以快速使用阿里云音视频通信的基本功能。

前提条件

开发音视频通信的前提条件如下所示。

- 您已经完成注册阿里云账号，并完成实名认证，具体操作请参见#unique_4。
- 您已经开通音视频通信服务，具体操作请参见#unique_5。

操作步骤

1. #unique_6。通过在音视频通信控制台创建应用，可以获取您的应用ID。
2. #unique_7。搭建AppServer，是开发音视频通信的服务端操作，并生成频道鉴权令牌。
3. 集成客户端SDK。阿里云音视频通信为您提供了iOS、Android、Mac、Windows、Web五个端的SDK，帮助您快速集成SDK。
4. 实现基本功能。AliRtcSDK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。
5. #unique_10。您可以使用应用管理、用量查询、通信记录等控制台功能。

2 创建应用

本文为您介绍了阿里云音视频通信控制台创建应用的具体操作步骤，您可以通过在控制台创建应用来获取应用ID。

前提条件

在创建应用前，请您完成以下操作。

- 您已经完成注册阿里云账号，并完成实名认证，具体操作请参见#unique_4。
- 您已经开通音视频通信服务，具体操作请参见#unique_5。

操作步骤

1. 登录音视频通信控制台。
2. 在左侧导航栏单击应用管理。
3. 单击创建应用。

4. 在云产品开通页面，单击立即开通。

> **说明:**
> 新开通应用默认状态为“停用”，在应用状态为“停用”时，创建与加入频道等操作将不可用，您需要在购买时长包之后，将应用状态切换到”启用”。

5. 返回应用管理页面，您可以查看应用ID。

> **说明:**
> 应用ID是阿里云音视频通信服务器用于区分不同应用的唯一标识。

3 搭建App Server

应用服务器（App Server）是指由用户自行研发的程序层。通过对接阿里云RTC云端服务实现用户应用的业务逻辑，如用户管理、鉴权校验等。本文以开发工具IDEA使用Java语言为例，为您介绍搭建App Server的具体操作步骤。

前提条件

在进行操作前，您需要：

- 获取应用ID，具体操作请参见#unique_6。
- 获取AppKey，具体操作请参见查询AppKey。

> 注意：
> AppKey是应用的唯一鉴权凭证，请您妥善保管。如果不慎泄露，为避免造成严重损失，请及时提交工单，申请更新AppKey。

操作步骤

1. 根据您的需求下载对应版本的App Server源码，阿里云音视频通信为您提供以下版本。
   - Golang
   - Java
   - Python
   - C#
   - Nodejs
   - PHP
2. 创建新工程（Project），将下载的源码复制在工程文件里。
3. 在*pom.xml*文件中添加Maven依赖。

> 说明：
> 本文的版本号仅供参考，具体版本号请参见AliRtcAppServer。

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http
://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>jetbrains</groupId>
    <artifactId>maven-rtc</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>
        <dependency>
```

```
            <groupId>com.aliyun</groupId>
            <artifactId>aliyun-java-sdk-rtc</artifactId>
            <version>0.8.0</version>
        </dependency>
        <dependency>
            <groupId>com.sun.net.httpserver</groupId>
            <artifactId>http</artifactId>
            <version>20070405</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>commons-cli</groupId>
            <artifactId>commons-cli</artifactId>
            <version>1.2</version>
        </dependency>
        <dependency>
            <groupId>com.aliyun</groupId>
            <artifactId>aliyun-java-sdk-core</artifactId>
            <optional>true</optional>
            <version>3.7.1</version>
        </dependency>
    </dependencies>
</project>
```

4. 单击Edit Configurations，在Vm Options填入启动参数。

```
--listen=8080
--appid=xxxxxxxx
--appkey=xxxxxxxxx
--gslb=https://rgslb.rtc.aliyuncs.com
```

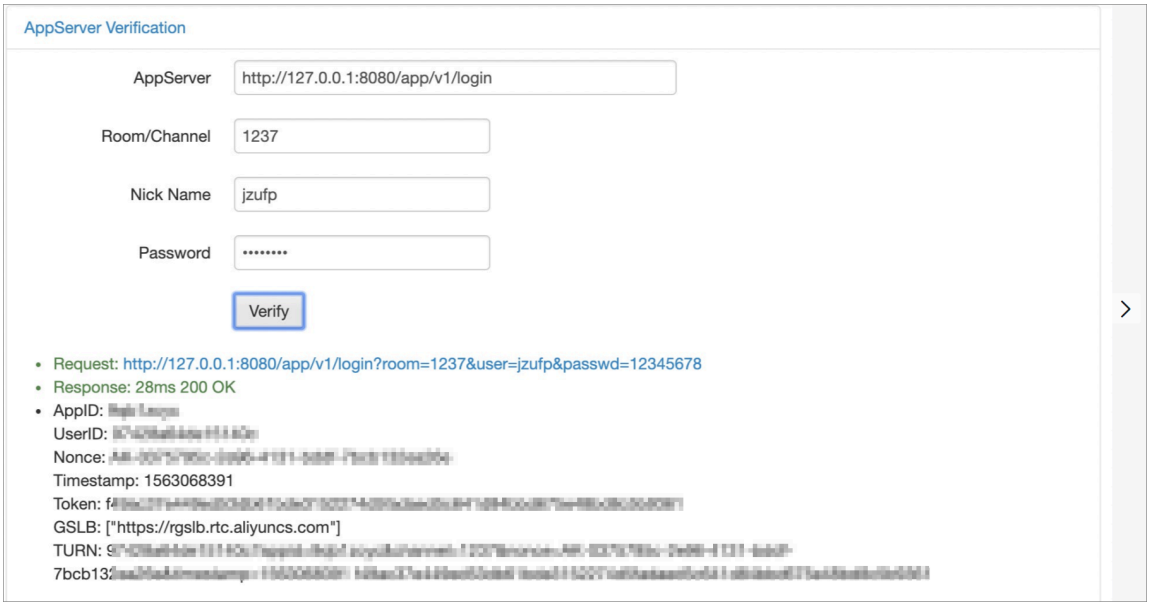
5. 单击运行。

6. 验证App Server，请单击AppServer Verification。

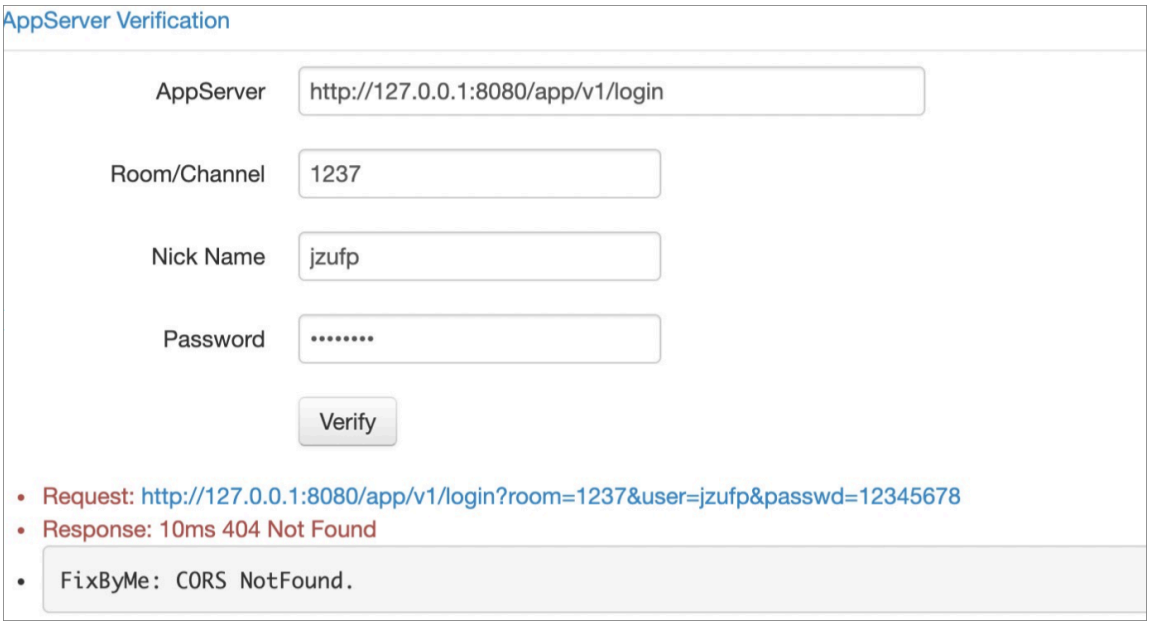
说明:

请将App Server`http://127.0.0.1:8080/app/v1/login`替换成您的AppServer地址:端口`/app/v1/login`。

- App Server启动成功。



- App Server未启动，访问失败。

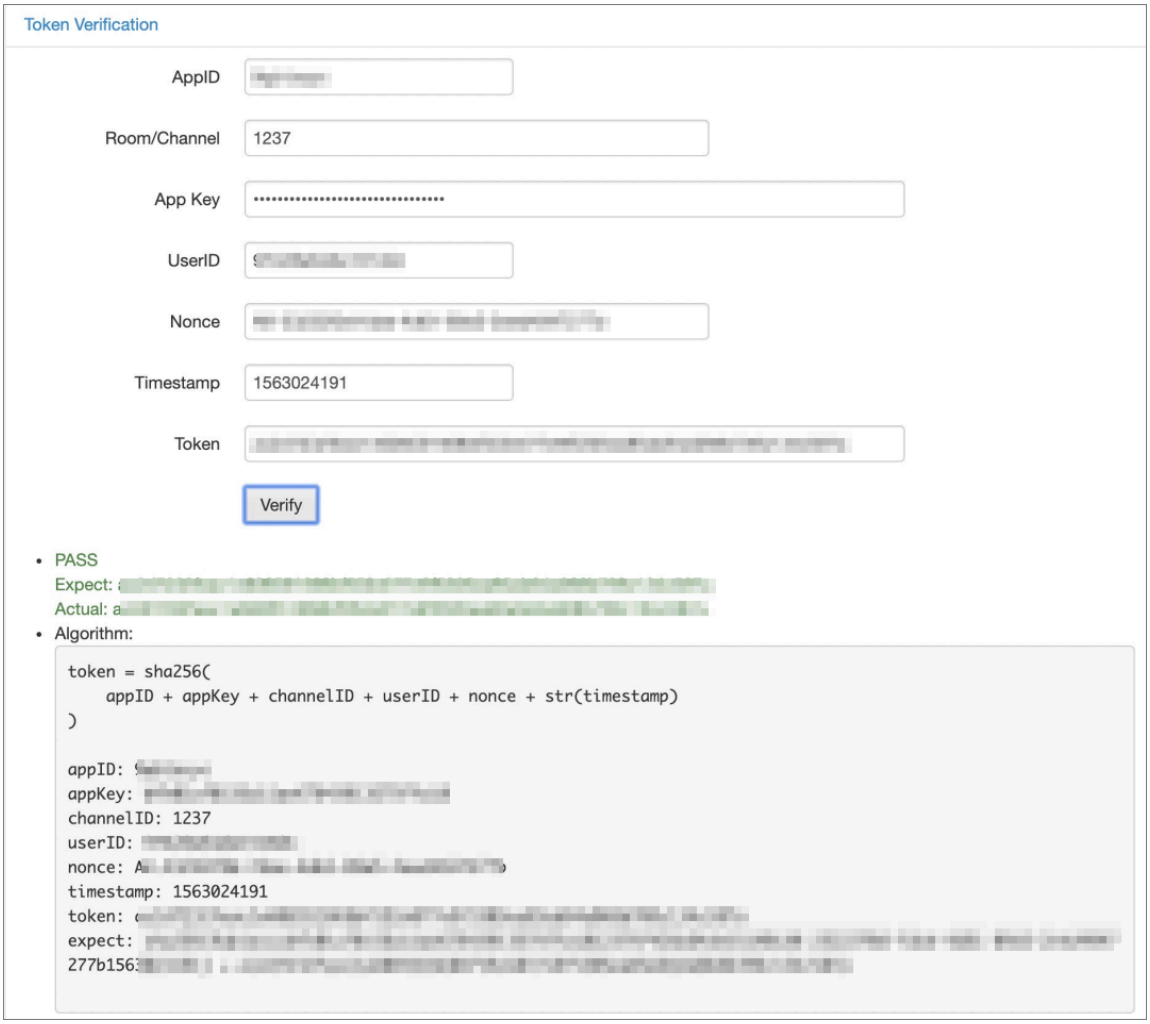


说明:

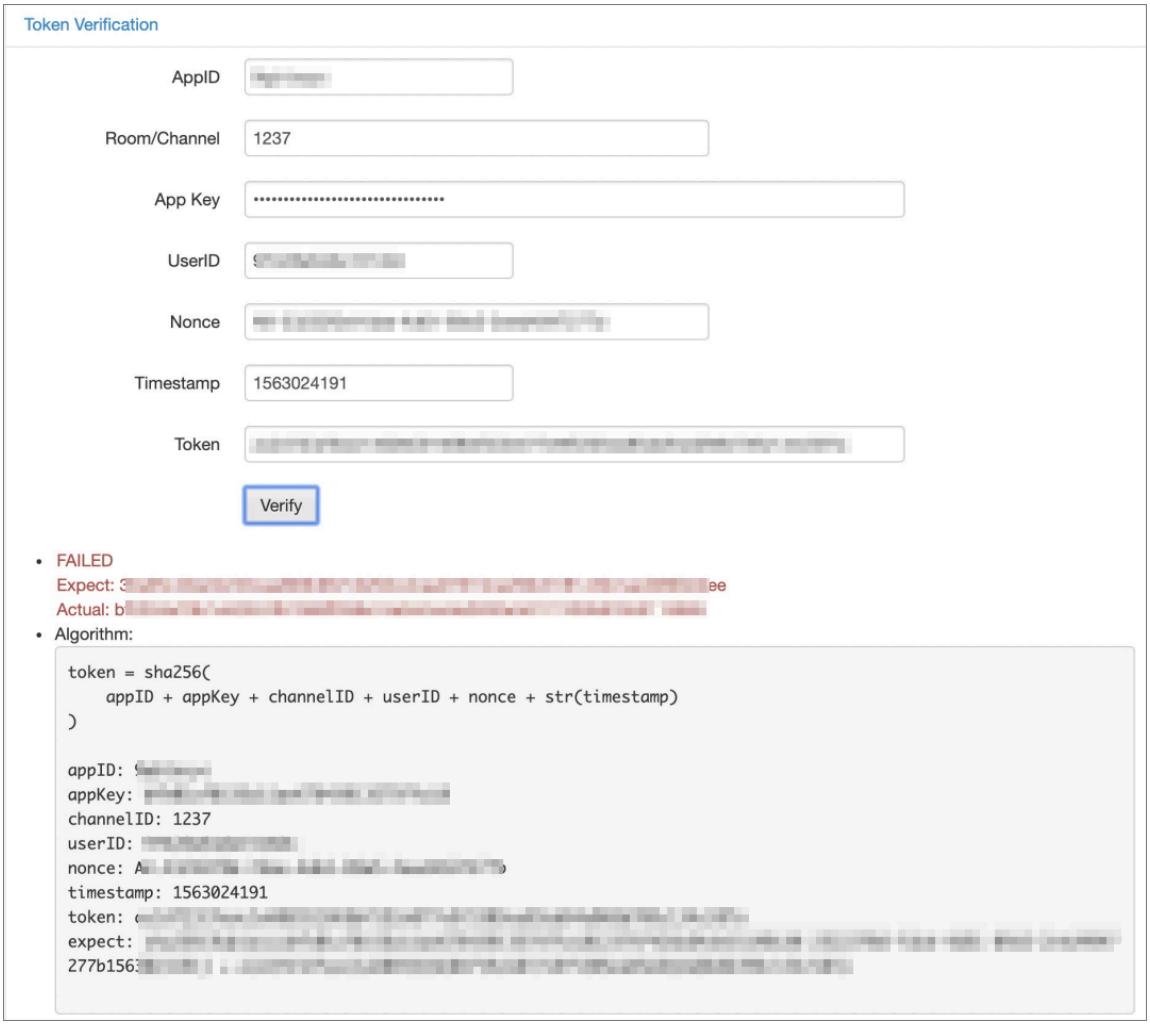
音视频通信Password并没有提供校验机制，您可以随意填写，建议默认即可。

7. 校验Token，请单击Token Verification。

   - 校验Token成功。



   - 校验Token失败。



后续步骤

阿里云音视频通信为您提供完整的频道鉴权开发流程，详情请参见#unique_14。

4 生成频道鉴权令牌

本文档为您介绍了生成频道鉴权令牌的开发流程，包括查询应用AppKey、为客户端创建令牌及下发频道信息三个步骤。

开发流程简介

生成频道鉴权令牌的流程如下所示。

- 步骤一：查询应用AppKey。在控制台的应用管理页面，查询应用的AppKey。因为涉及敏感操作，您需要短信验证授权才可以查询。
- 步骤二：为客户端创建令牌。使用应用的鉴权私钥和客户端信息，为客户端生成令牌（Token）。
- 步骤三：下发频道信息和令牌。将频道以及客户端令牌（Token）等信息，下发给客户端，作为参数传递给客户端RTC SDK。

步骤一：查询应用AppKey

> 注意:
> AppKey是应用的唯一鉴权凭证，请您妥善保管。如果不慎泄露，为避免造成严重损失，请及时提交工单，申请更新AppKey。

具体操作请参见查询AppKey。

步骤二：为客户端创建令牌

令牌（Token）是客户端的凭证，根据应用鉴权私钥（Appkey）和客户端信息生成的。加入频道时音视频通信服务会校验客户端的令牌，所以您需要为加入频道的每个客户端，生成唯一的鉴权令牌。

令牌的生成规则如下所示。

- 生成ChannelID，说明如下：
  - 频道标识，由appserver生成，保证唯一性。推荐用UUID。
  - 由字母[a-zA-Z]和数字[0-9]组成，不包含特殊字符，可以用连接号（-）分隔，最大64字节。例如：181-218-3406。
- 生成UserID，说明如下：
  - 终端用户唯一标识，由AppServer生成，为保证唯一性，推荐用UUID。
  - 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：2b9be4b25c2d38c409c376ffd2372be1。

- 生成Nonce，说明如下：
  - 令牌随机码，由用户生成，推荐用UUID。
  - 需要加上前缀AK-，以标识采用应用鉴权私钥（AppKey）方案。
  - 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。
- 生成Timestamp，说明如下：
  - 令牌过期时间戳，由用户生成指定令牌过期时间。
  - 为Unix时间格式，AppServer所运行的服务器需保持时间同步。
  - 例如：当前时间戳为1560415794（2019-06-13 16:49:54）令牌2天后过期，Timestamp设置为1560588594（2019-06-15 16:49:54）。
- 生成Token的原始字符串的拼接顺序：
  - AppID：应用ID，使用控制台创建。
  - AppKey：应用秘钥，使用控制台查询。
  - ChannelID：频道ID，AppServer生成。
  - UserID：您的唯一标识，AppServer生成。
  - Nonce：令牌随机码，AppServer生成。
  - Timestamp：令牌过期时间戳，AppServer生成。
- 使用SHA-256哈希加密算法生成字符串的摘要。

您可以参见以下不同版本的创建Token函数。

- Golang程序实例请参见Demo中的CreateToken函数，详情请参见Usage。
- Java程序实例请参见Demo中的createToken函数，详情请参见Usage。
- Python程序实例请参见Demo中的create_token函数，详情请参见Usage。
- C#程序实例请参见Demo中的CreateToken函数，详情请参见Usage。
- Nodejs程序实例请参见Demo中的CreateToken函数，详情请参见Usage。
- PHP程序实例请参见Demo中的CreateToken函数，详情请参见Usage。

步骤三：下发频道信息和令牌

将频道以及客户端令牌（Token）等信息，下发给客户端，作为参数传递给客户端RTC SDK。

参数	说明
AppID	应用ID，使用控制台创建。

参数	说明
UserID	您的唯一标识，AppServer生成。同一个UserId的用户在其他端登录，先入会的端会被后入会的端踢出房间。
ChannelID	频道ID，AppServer生成。
Nonce	频道随机码。AppServer生成。
Timestamp	频道时间戳。AppServer生成。
Token	加入频道Token，AppServer生成。
GSLB	服务地址，当前请使用：`https://rgslb.rtc.aliyuncs.com`，GSLB地址会不定期更新，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。

5 集成客户端SDK

5.1 Android

本文为您介绍了Android端集成SDK操作，帮助您快速集成SDK并能使用音视频通信基本功能。

前提条件

开发前的环境要求如下表所示，详情请参见#unique_18。

类别	说明
系统版本	支持Android 4.1及以上
API版本	不低于16
CPU架构	支持真机架构armeabi、armeabi-v7a、arm64-v8a

您需要下载SDK。解压后的文件需导入到Android Studio工程*libs*文件下，文件类型如下表所示。

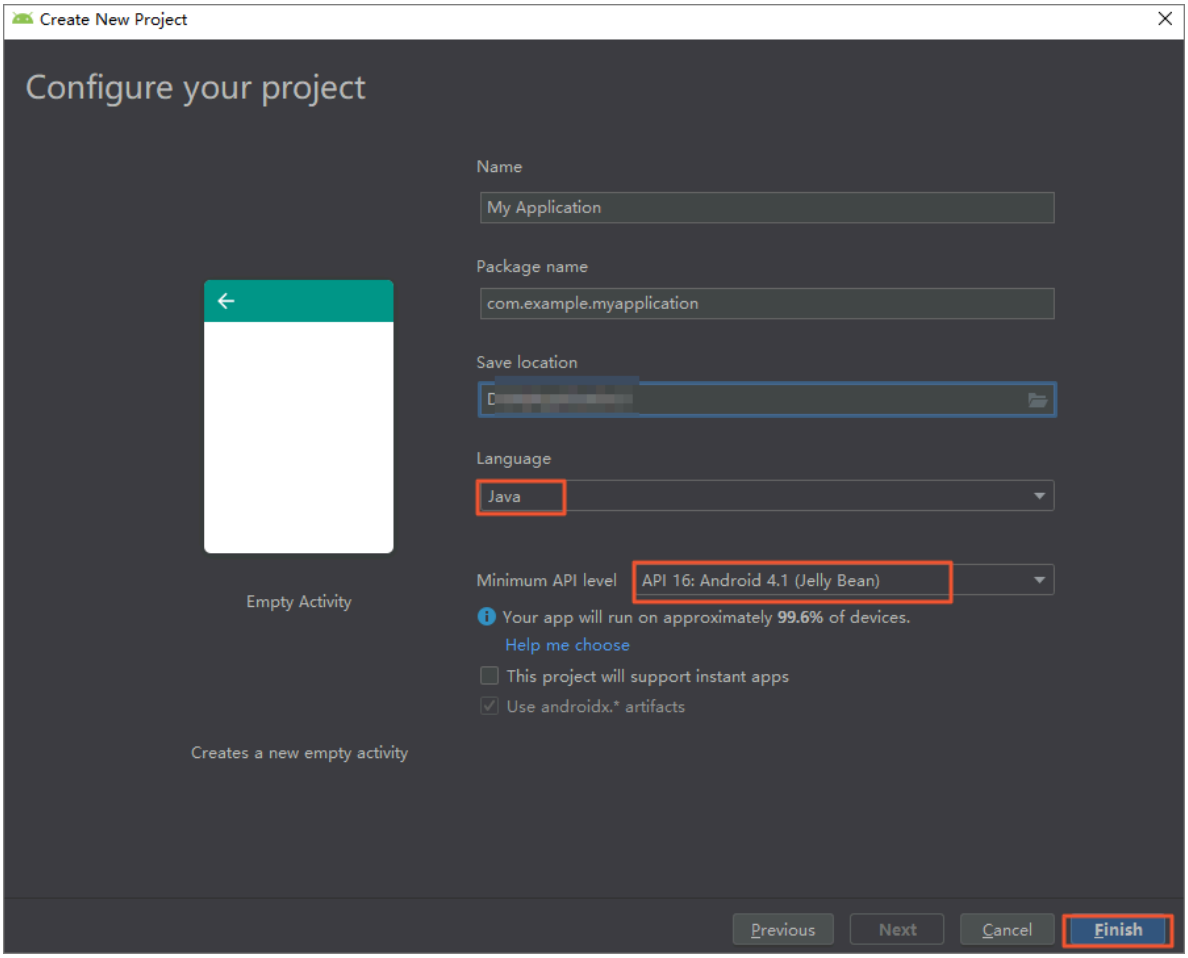
文件或文件夹名称	文件类型
AliRTCSdk	jar
utdid4all-1.5.0-proguard	jar
Sophonsdk	aar
alivc-core-rtc	aar
webrtclib	aar

操作步骤

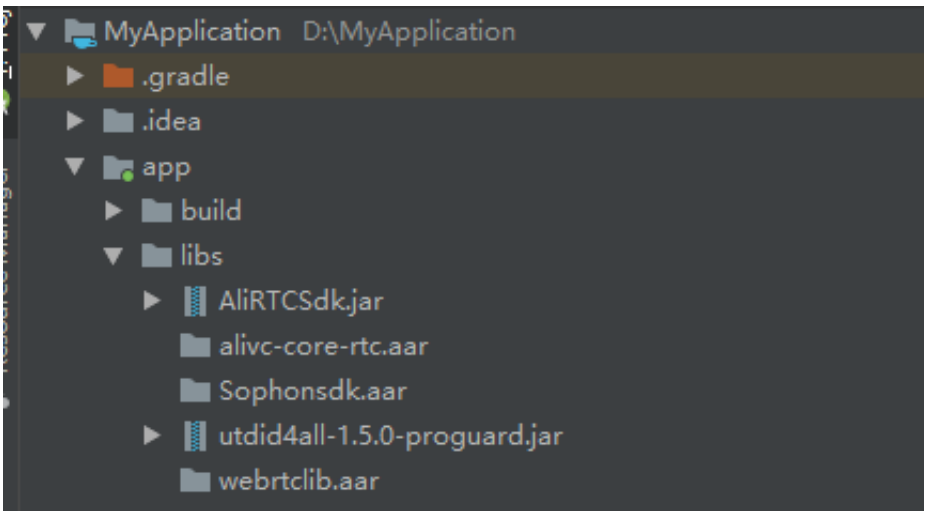
1. 使用Android Studio软件创建一个新的Empty Acitivity，并根据下图所示进行配置。

说明:

本文档的Android Studio版本为3.4.1。



2. 把解压的SDK文件导入到*app/libs*目录下。



3. 在*app/src/build.gradle*文件中添加如下配置。

```
android {
    ...
    defaultConfig{
        ...
        ndk {
```

```
            abiFilters  "armeabi", "armeabi-v7a", "arm64-v8a"
        }
         ...
    }
   ...
   repositories {
      flatDir {
        dirs 'libs'
      }
   }
}
dependencies {
implementation fileTree(dir: 'libs', include: ['*.jar','*.aar'])
...
```

4. 在*app/src/main/AndroidManifest.xml*文件中添加摄像头、麦克风、网络，访问存储权限。在代码里面需要添加动态权限申请。

```
<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_EXTE
RNAL_STORAGE"/>
<uses-permission android:name="android.permission.CHANGE_NET
WORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NET
WORK_STATE"/>
<uses-permission android:name="android.permission.MODIFY_AUD
IO_SETTINGS"/>
```

5. （可选）混淆配置。 如果您的应用设置了混淆配置，需要进行以下配置。在*proguard-rules.pro*文件中，添加`-keep`类的配置，这样可以防止混淆AliRtcSDK公共类名称。

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

6. 单击Sync Project With Gradle Files，同步项目文件，直到同步完成。

后续步骤

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

5.2 iOS

本文为您介绍了iOS端集成SDK操作，帮助您快速集成SDK并能使用音视频通信基本功能。

前提条件

开发前的环境要求如下表所示，详情请参见#unique_18。

类别	说明
iPhone设备	支持iPhone5及以上
系统版本	支持iOS 8.0及以上

类别	说明
CPU架构	支持真机架构armv7+arm64，不支持模拟器i386、x86架构
Xcode版本	9.0及以上
其他	不支持bitcode，不支持屏幕旋转

说明:

您需要持有Apple开发证书或个人账号。

CocoaPods集成

说明:

请确保您的Mac已经安装ruby环境。

1. 安装CocoaPods。在Mac终端窗口中输入如下命令。

```
sudo gem install cocoapods
```

2. 创建Podfile文件。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
```

3. 编辑Podfile文件。

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

4. 安装SDK。

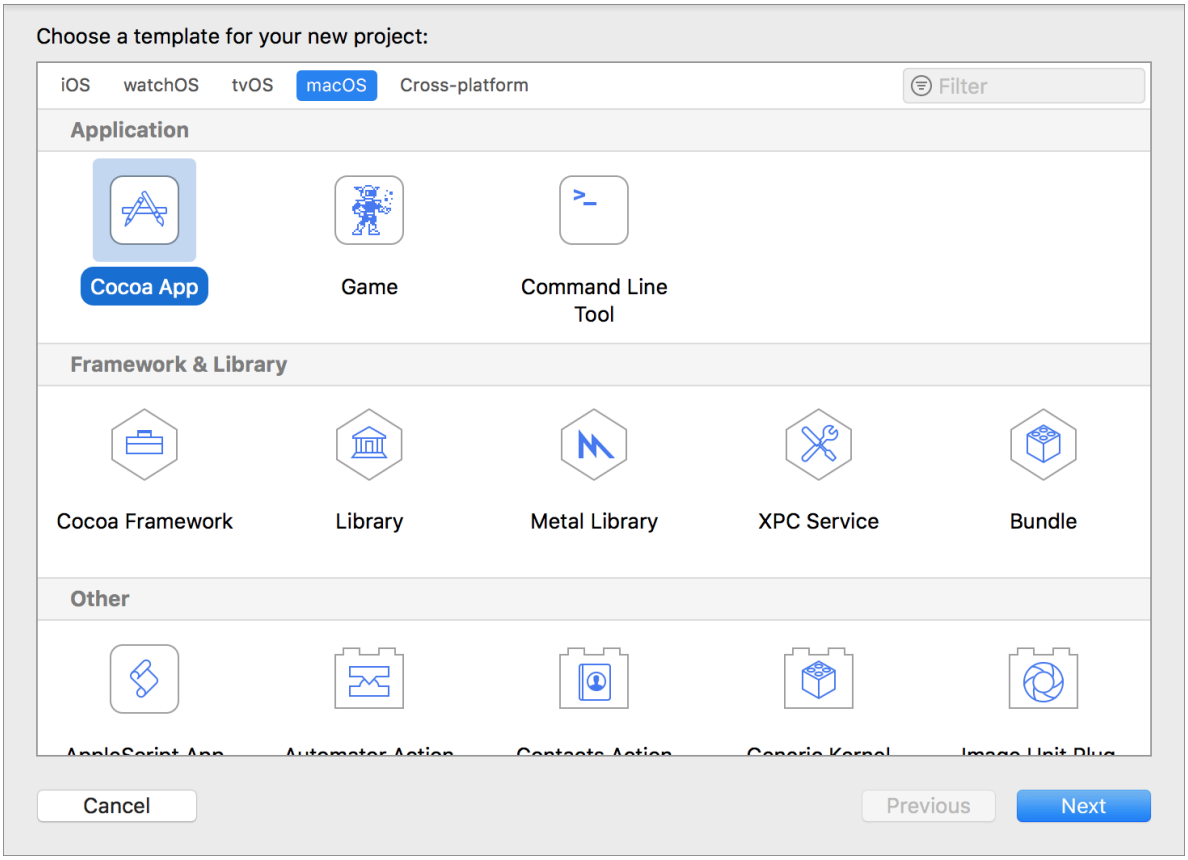
```
pod install
```

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SDK。

2. 使用XCode工具创建一个新的iOS工程，并把SDK包拷贝到您的工程中。

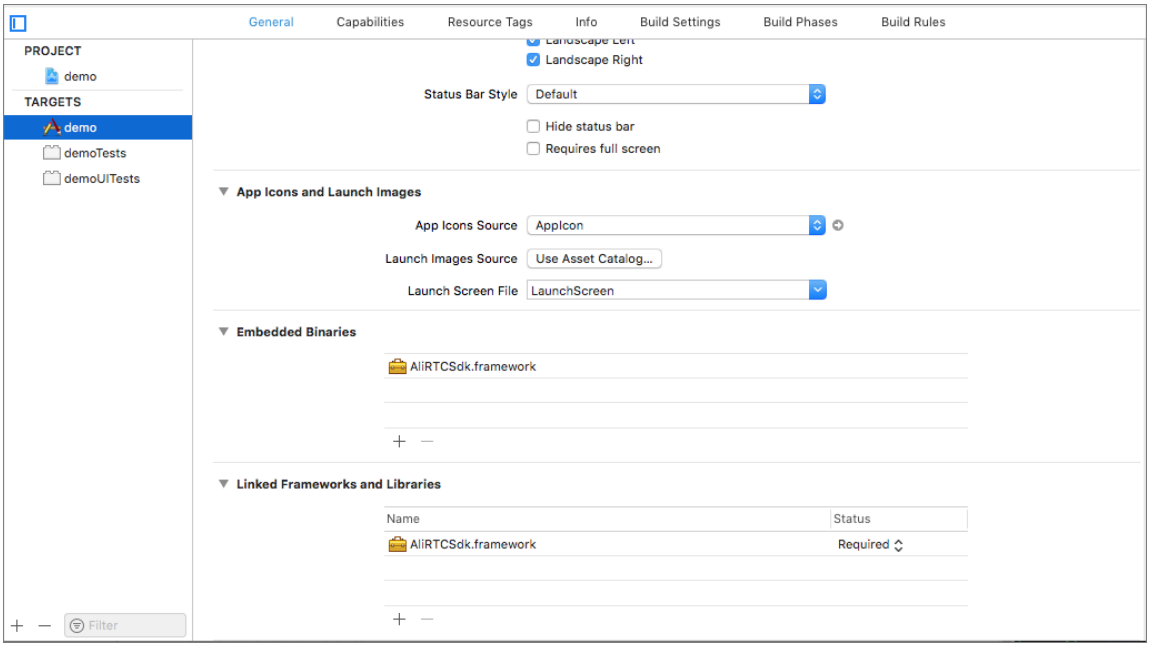


3. 添加文件。

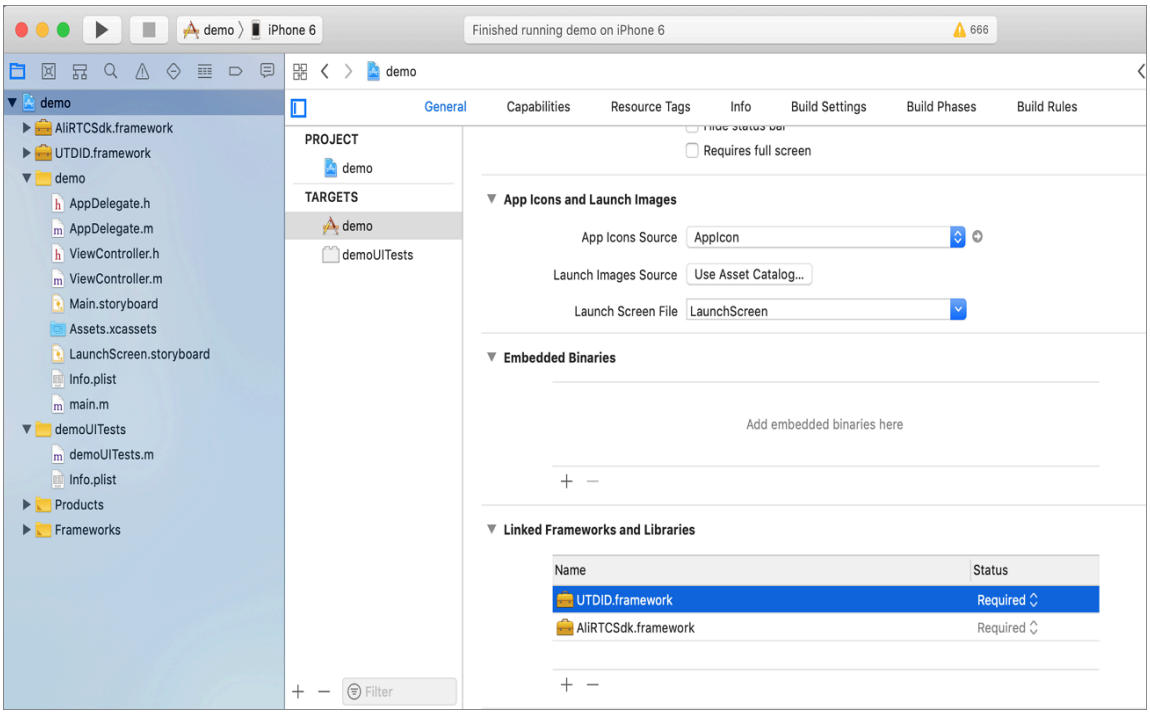
a) 在General页面，将SDK中*AliRTCSdk.framework*加入到工程。

说明:

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



b) 在General页面，将SDK中*UTDID.framework*加入到工程。

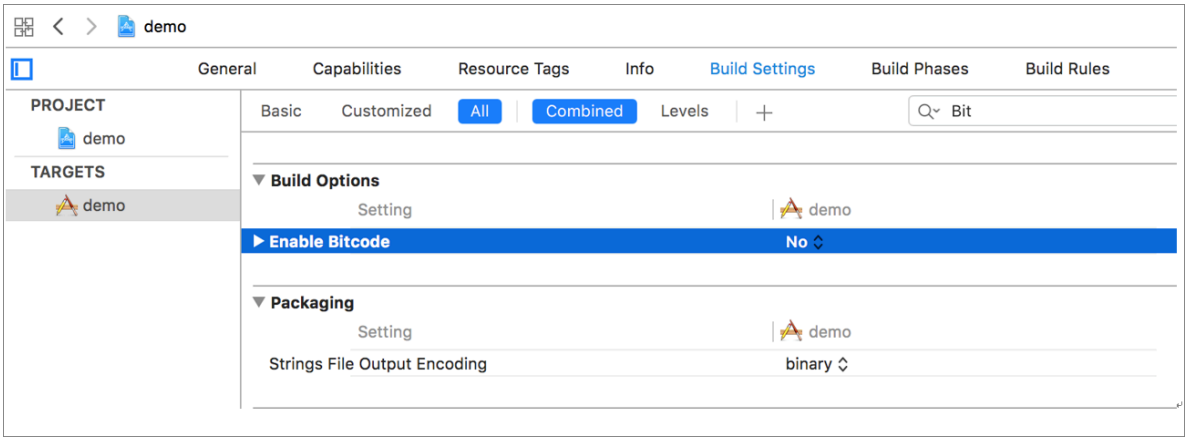


4. 在Build Phases页面，添加系统依赖。

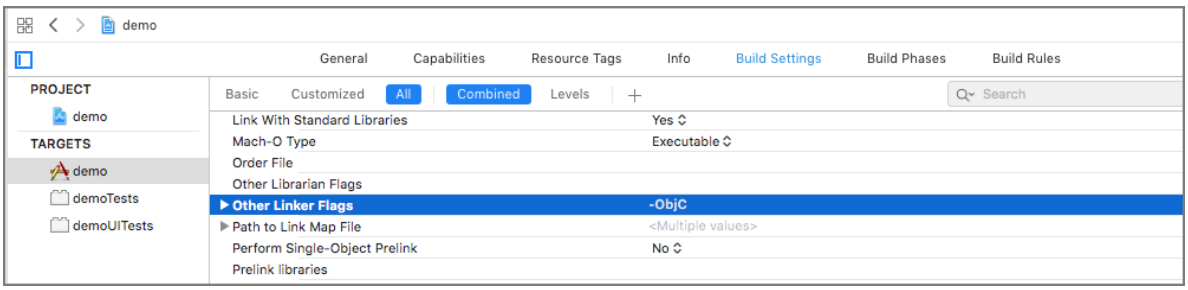
   系统依赖如下所示。

   - libc++.tbd
   - CoreMedia.framework
   - AVFoundation.framework
   - libz.tbd
   - libresolv.tbd
   - AudioToolbox.framework
   - VideoToolbox.framework

5. 在Build Settings页面，设置Enable Bitcode为No。



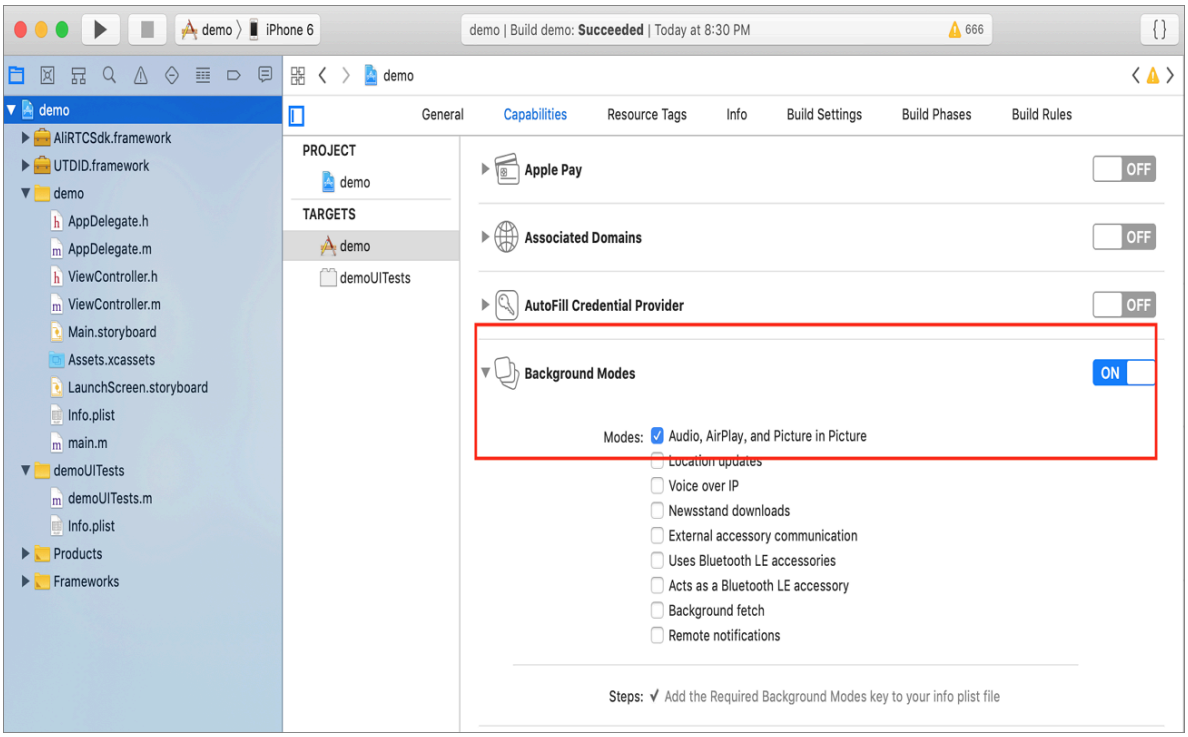
6. 在Build Settings页面，添加-ObjC链接选项。



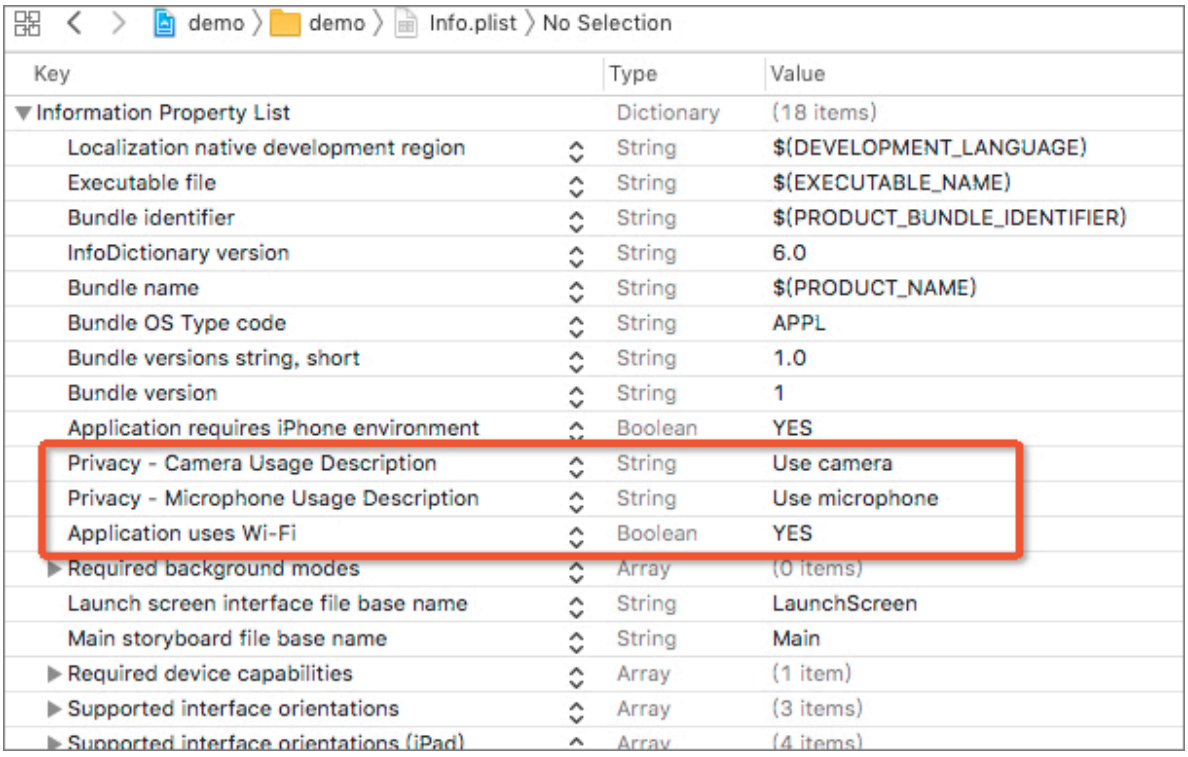
7. 在Capabilities页面，打开后台音频权限。

说明:

为保障APP退入手机后台之后，通话可以保持不中断，建议开启后台音频权限，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iPhone，执行编译Commond +B，提示Build Success，表示SDK集成成功。

后续步骤

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

5.3 Mac

本文为您介绍了Mac端集成SDK操作，帮助您快速集成SDK并能使用音视频通信基本功能。

前提条件

开发前的环境要求如下表所示，详情请参见#unique_18。

类别	说明
Mac设备	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
系统版本	支持macOS 10.12及以上
CPU架构	支持真机架构armv7+arm64，不支持模拟器i386、x86架构
Xcode版本	9.0及以上
其他	不支持屏幕旋转

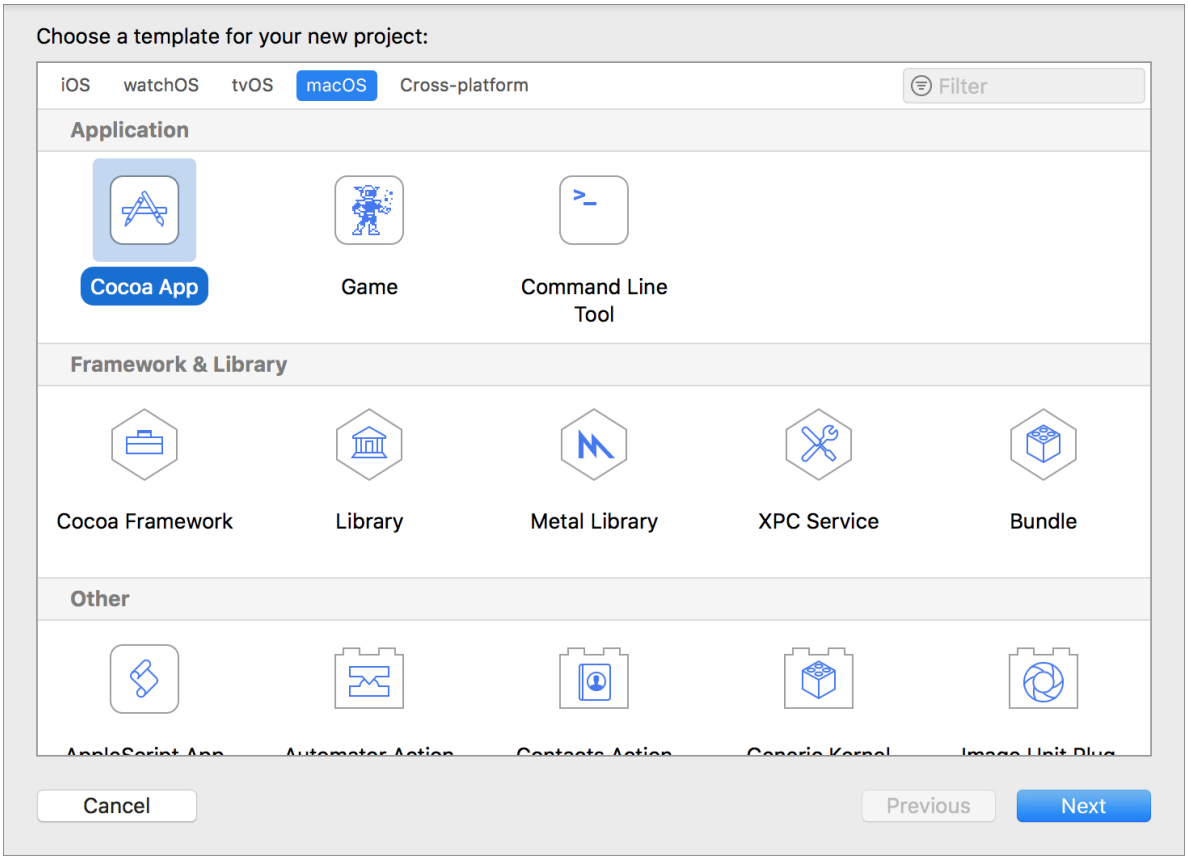
说明:

您需要持有Apple开发证书或个人账号。

操作步骤

1. 下载SDK。

2. 使用XCode工具创建一个新的iOS工程，并把SDK包拷贝到您的工程中。



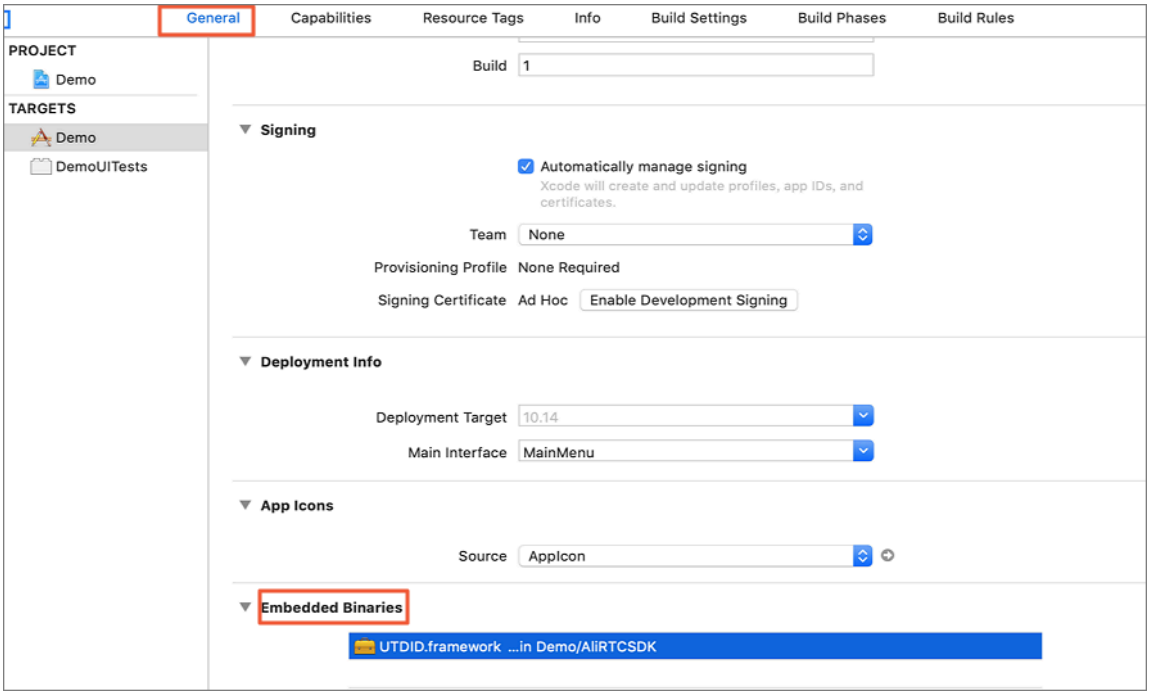
3. 添加文件。

a) 选择Build Phases > Link Binary With Libraries，将*AliRTCSdk.framework*和*UTDID.framework*加入到Link Binary With Libraries。

b) 在General页面，添加*UTDID.framework*到Embedded Binaries中。

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。

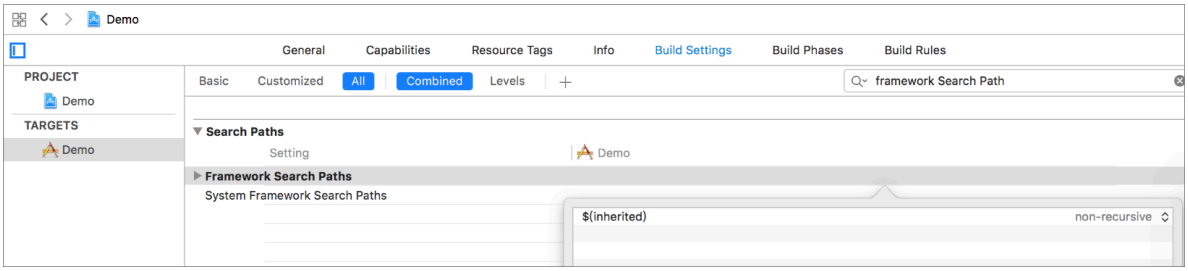


4. 在Build Phases页面，添加系统依赖。

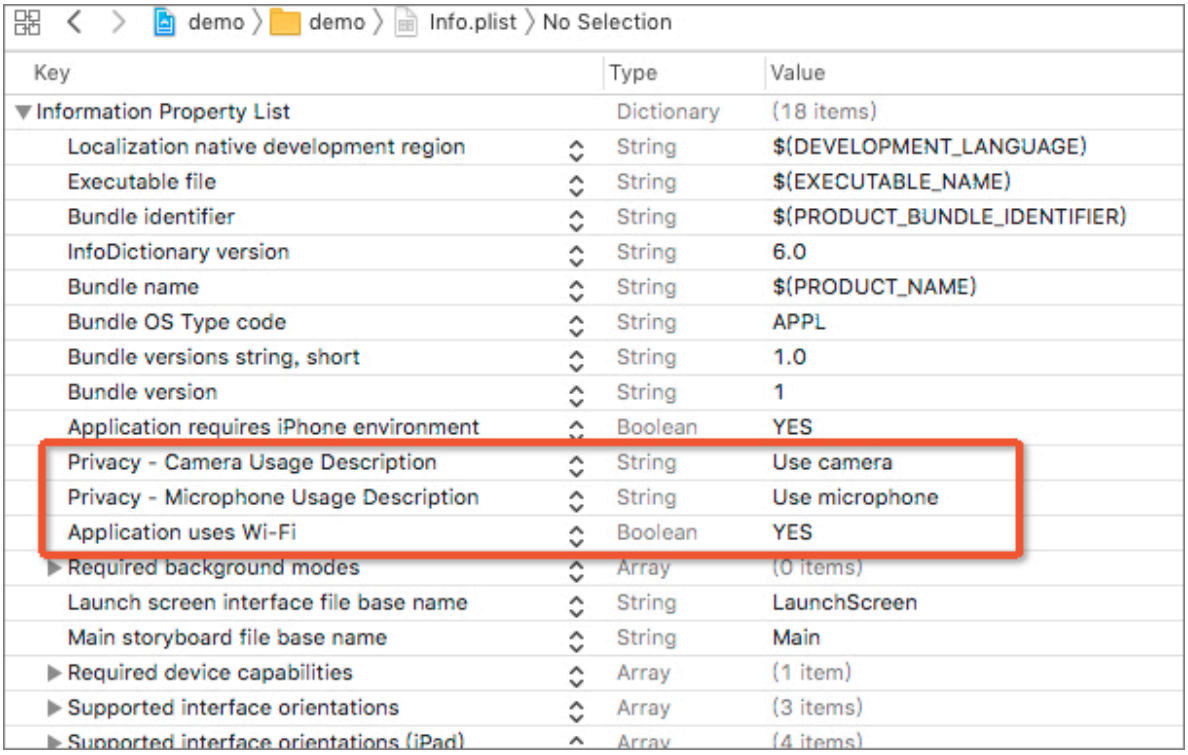
   相关系统库如下所示。

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

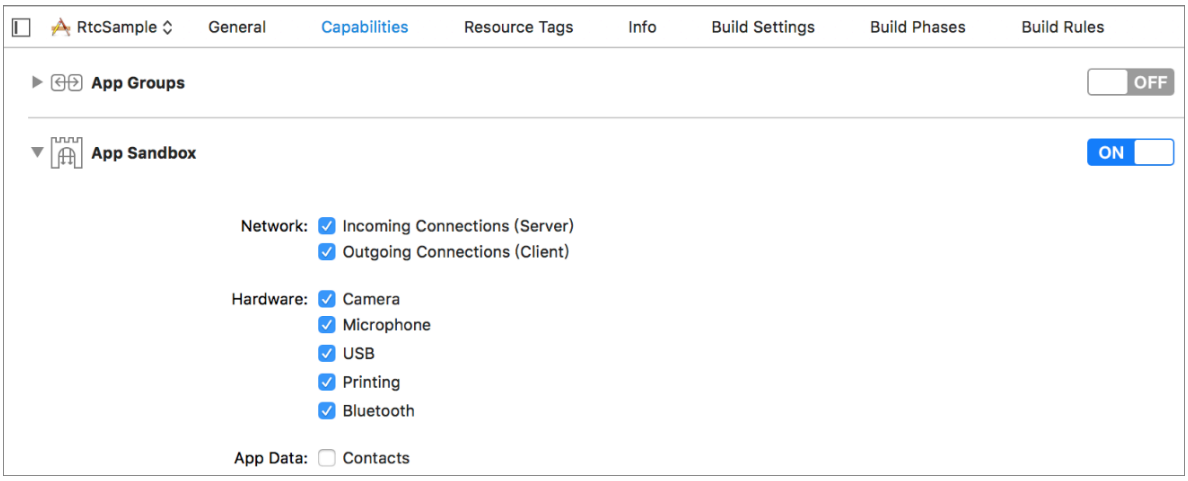
5. 选择Build Settings > Framework Search Path，将*AliRTCSDK.framework*文件夹拖入弹出框内。



6. 编辑*info.plist*文件，添加权限。



7. 在Capabilities页面，设置权限。



8. 执行编译Commond +B，提示Build Success，表示SDK集成成功。

后续步骤

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

5.4 Windows

本文为您介绍了Windows端集成SDK操作，帮助您快速集成SDK并能使用音视频通信基本功能。

前提条件

开发前的环境要求如下表所示，详情请参见#unique_18。

类别	说明
系统版本	支持Windows XP（SP3）、Windows 7、Windows 8.X、Windows 10
系统位数	只支持32位
开发环境	支持Visual Studio 2010及以上

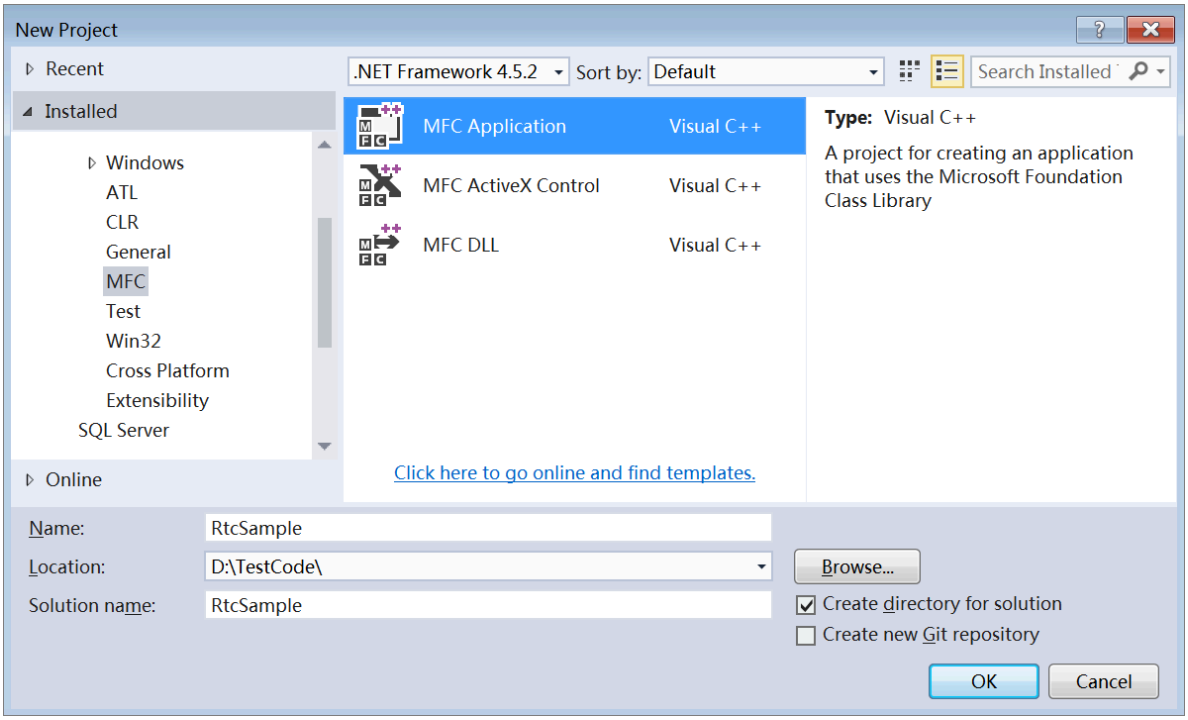
说明:

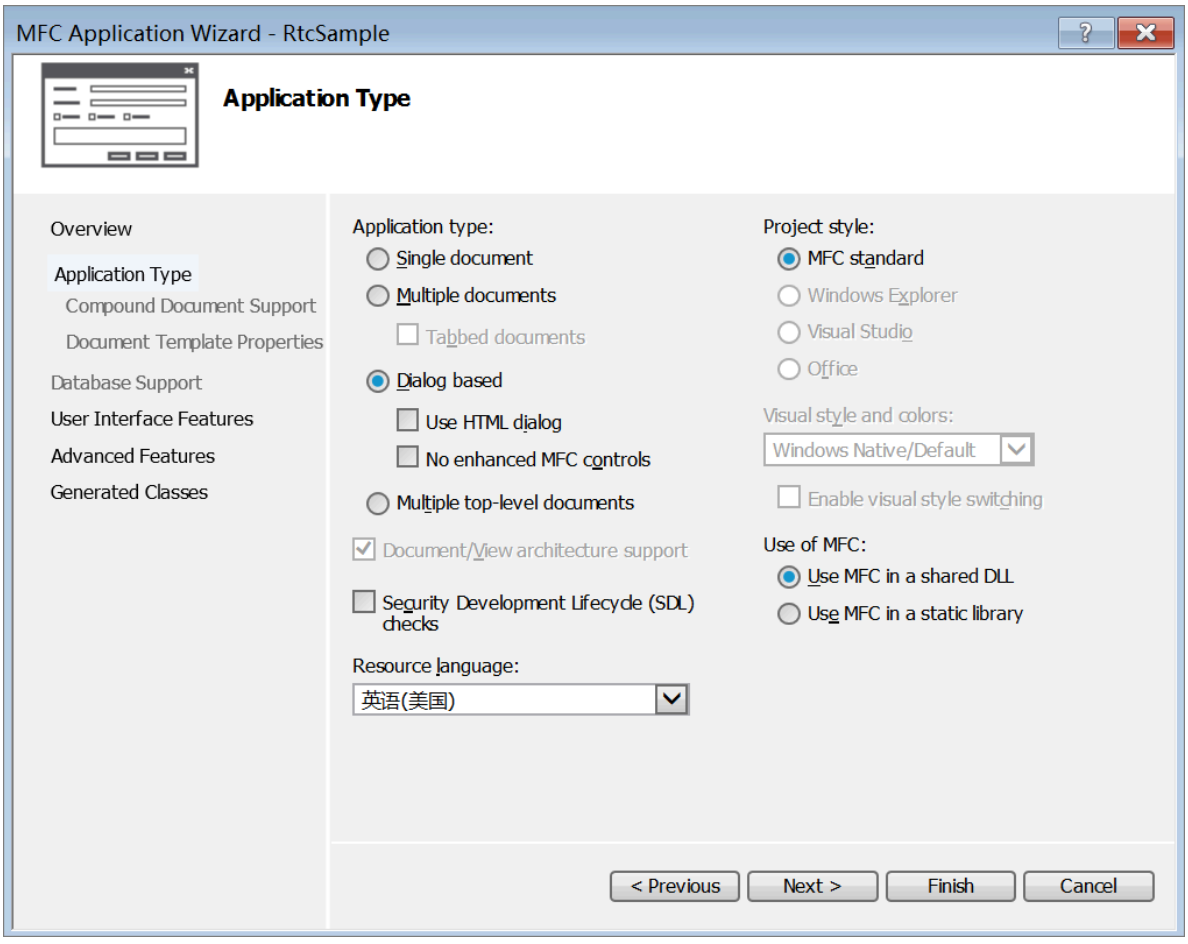
本文以MFC的工程为例，您也可以选择其他UI框架。

操作步骤

1. 下载SDK。

2. 使用Visual Studio创建一个MFC Dialog based类型的工程，工程命名为RTCSample。



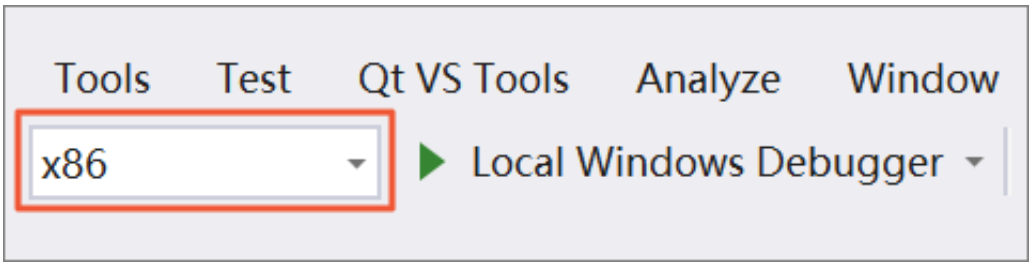


3. 将*RTC EngineSDK*的相关文件移动到*.sln*文件所在的目录。

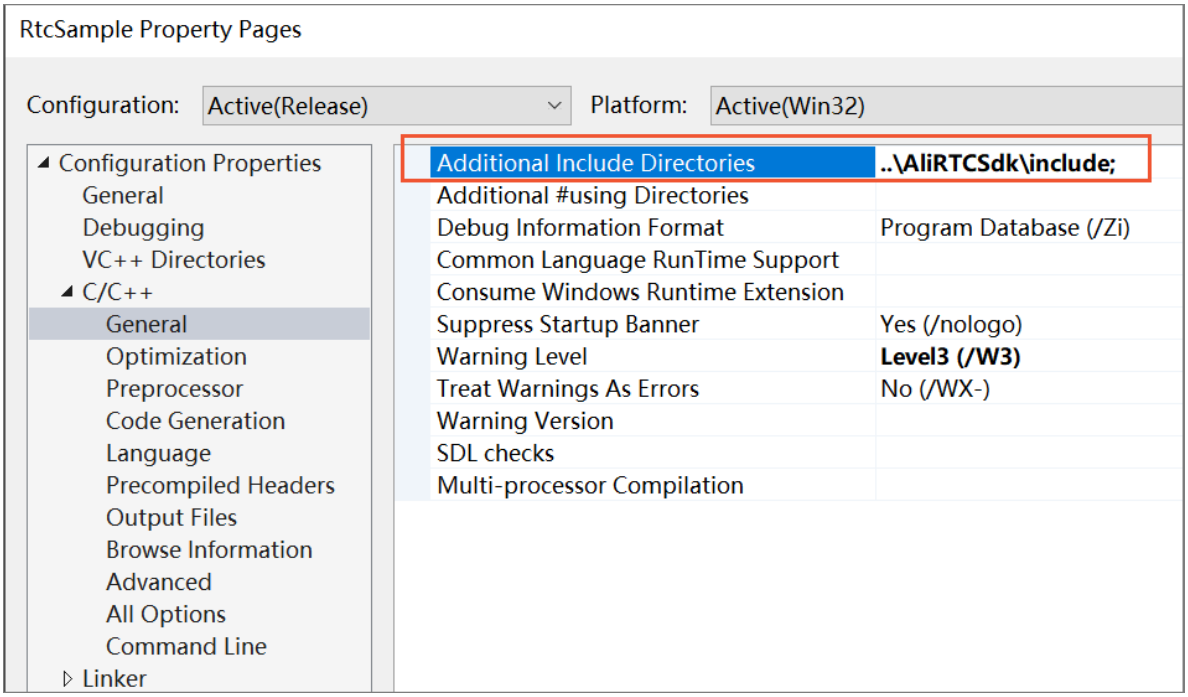
4. 配置RtcSample工程的属性，添加RTC EngineSDK库及头文件的路径。

> **说明:**
> 当前版本SDK只支持Release x86版。

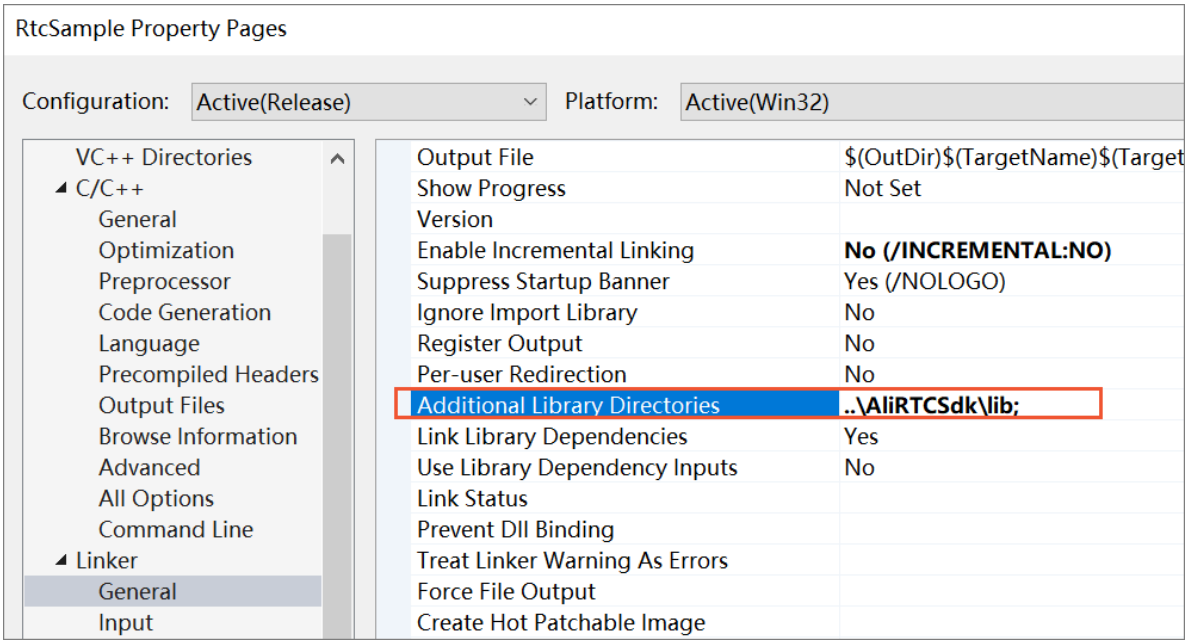
5. 设置32位工程属性。



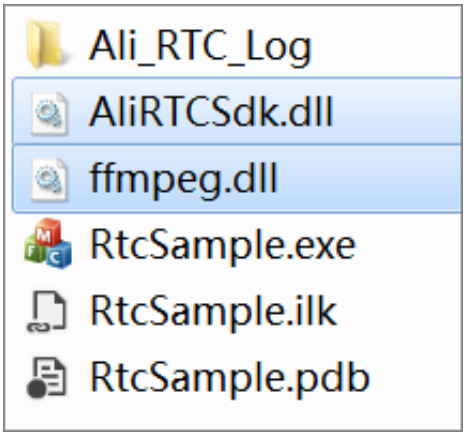
6. 在工程属性页面，设置依赖头文件的路径。



7. 设置静态库的路径。



8. 复制*AliRTCSdk.dll*文件及依赖的*ffmpeg.dll*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，您可以实现音视频通信的基本功能，详情请参见Windows实现基本功能。

5.5 Web

本文为您介绍了Web端集成SDK操作，帮助您快速集成SDK并能使用音视频通信基本功能。

背景信息

开发前的环境要求如下表所示，详情请参见#unique_18。

支持平台	浏览器	浏览器版本	备注
iOS	Safari	不低于11.1.2	—
Android	Chrome	不低于63	需要手机支持H264
Mac	Chrome	不低于60	—
Mac	Safari	不低于11	—
Windows	Chrome	不低于60	—

说明:

Web SDK目前仅对部分受邀用户开放，如果您有业务需求，请单击音视频通信WEB SDK使用申请。

操作步骤

1. 从官网申请获得AliWebRtcSDK包保存到本地项目下。
2. 在项目相应的前端页面文件中，对*aliyun-webrtc-sdk.js*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:

本图片仅供参考，具体版本号和引入样式请您以实际为准。

后续步骤

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

6 实现基本功能

6.1 Android

阿里云音视频通信的基本功能包含初始化SDK、加入频道、本地发布和订阅远端、离开频道等。当您成功初始化SDK，您可以进行本地预览视频功能，进行简单的预览和测试，您也可以设置手动或者自动模式。

前提条件

在实现基本功能前，请您确保下载最新SDK，请参见#unique_19。

说明:

本文中的实现方法为主要功能方法，仅供参考，您可以根据您的业务需求进行实际开发。

操作步骤

1. 初始化SDK。

   在*app/src/main/java/come.xample.myapplication/MainActivity*文件中，您需要调用onCreate方法创建AliRtcEngine实例，并注册回调。相关回调有AliRtcEngineEventListener和AliRtcEngineNotify。具体回调及监听请参见#unique_30。

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

   说明:

   只能在主线程调用，暂不支持多实例。

   mEventListener：用户操作回调监听（回调接口都在子线程）。

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

   mEngineNotify：SDK事件通知（回调接口都在子线程）。

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

```
};
```

a) 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas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.AliVideoCa
nvas();
//SDK内部提供进行播放的view
SophonSurfaceView surfaceView = new SophonSurfaceView(this);
surfaceView.setZOrderOnTop(true);
surfaceView.setZOrderMediaOverlay(true);
mSurfaceContainer.addView(surfaceView,new ViewGroup.LayoutPara
ms(ViewGroup.LayoutParams.MATCH_PARENT,ViewGroup.LayoutParams.
MATCH_PARENT));
/* 预览窗口的view */
canvas.view = surfaceView;
//renderMode提供四种模式：Auto、Stretch、Fill、Crop，建议使用Auto模式。
canvas.renderMode = AliRtcRenderModeAuto;
mEngine.setLocalViewConfig(canvas, AliRtcVideoTrackCamera);
mEngine.startPreview();
//mSurfaceContainer为包裹mCanvas的父view
mSurfaceContainer.getChildAt(0).setVisibility(View.VISIBLE);
```

建议您把mAliVideoCanvas之上的所有view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。

mirrorMode提供三种模式：AliRtcRenderMirrorModeOnlyFront、AliRtcRenderMirrorModeAllEnabled、AliRtcRenderMirrorModeAllDisable。

b) 设置自动或者手动模式。

说明:

阿里云音视频通信Android端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Publish模式：如果您打开自动Publish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Publish模式，则需要您调用publish接口之后才会发布音视频流。
- 自动Subscribe模式：如果您打开自动Subscribe模式，加入频道之后，SDK将会自动订阅当前频道内其他用户的音视频流；如果关闭自动Subscribe模式，则需要您调用subscribe接口之后才会订阅其他用户的音视频流。

```
/**
*设置自动发布和订阅，只能在joinChannel之前设置
*@param autoPub    true表示自动发布；false表示手动发布
*@param autoSub    true表示自动订阅；false表示手动订阅
*/
```

```
mEngine.setAutoPublish(true, true);
```

2. 加入频道。

> 说明:
> AliRtcAuthInfo的各项参数均需要App Server通过API来获取，然后App Server下发至客户端，客户端将各项参数赋值后，就可以加入频道。
>
> - name：无需APP Server下发。
> - channelID 、userID、name命名要求：字符内容只允许[A-Za-z0-9_-]，长度限制64字节，非法命名系统将拒绝提供服务。

```
AliRtcAuthInfo userInfo = new AliRtcAuthInfo();
userInfo.setConferenceId(/* 用户的channelID */);
userInfo.setAppid(/* 用户的appid */);
userInfo.setNonce(/* 用户的nonce */);
userInfo.setTimestamp(/* 用户的timestamp */);
userInfo.setUserId(/* 用户的userid */);
userInfo.setGslb(/* 用户的gslb */);
userInfo.setToken(/* 用户的token */);
if(mEngine != null) {
mEngine.joinChannel(userInfo, /* name */);
}
```

3. 发布或取消发布本地流。

发布本地流。

- 自动pub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。
- 非自动pub模式下：加入成功后，可通过以下接口发布本地流。

如果publish过程中需要变更配置或者停止publish，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.
AliRtcVideoTrackCamera);
mEngine.publish();
```

取消发布本地流。

```
mEngine.configLocalAudioPublish(false);
mEngine.configLocalCameraPublish(false);
mEngine.configLocalScreenPublish(false);
mEngine.configLocalSimulcast(false, AliRtcEngine.AliRtcVideoTrack.
AliRtcVideoTrackCamera);
```

```
mEngine.publish();
```

发布和取消发布本地流回调代码如下所示。

```
private AliRtcEngineEventListener mEventListener = new AliRtcEngi
neEventListener() {
@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sub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。
   - 非自动sub模式下：加入频道成功后，可通过以下接口订阅远端流。

   如果subscribe过程中需要变更配置或者停止subscribe，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.
AliRtcAudioTrackaudioTrack,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,AliRtcAudioTrack audioTrack) {
//订阅流回调，可以做UI及数据的更新
}
}
```

5. 离开频道。

   - 对于版本号大于1.7的SDK，请调用如下接口。

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

   - 对于版本号小于等于1.7的SDK，请增加timeout参数，一般建议设置为1000，表示该接口的调用超时时间为1秒，建议在Activity的onDestroy中调用。调用leavechannel后请不要再操作AliRtcEngine实例。

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

   您可以下载示例代码，快速跑通Demo，实现频道内和其他用户进行实时音视频通话，详情请参见#unique_31。

6.2 iOS

阿里云音视频通信的基本功能包含初始化SDK、加入频道、本地发布和订阅远端、离开频道等。当您成功初始化SDK，您可以进行本地预览视频功能，进行简单的预览和测试，您也可以设置手动或者自动模式。

前提条件

在实现基本功能前，请您确保下载最新SDK，请参见#unique_19。

> 说明:
> 本文中的实现方法为主要功能方法，仅供参考，您可以根据您的业务需求进行实际开发。

操作步骤

1. 请您先初始化SDK。

   首先您需要创建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
```

创建SDK实例并注册delegate。

iOS回调详情请参见#unique_32。

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

说明:

目前SDK暂不支持多实例。

a) 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas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 = AliRtcRenderMirrorModeOnlyFrontCameraPre
viewEnabled;
[self.engine setLocalViewConfig:canvas forTrack:AliRtcVide
oTrackCamera];
[self.engine startPreview];
```

您也可以取消本地预览。

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

说明:

- renderMode提供四种模式：Auto、Stretch、Fill、Crop，建议使用Auto模式。
- view必须是AliRenderView或者其子类。
- mirrorMode在本地或远端均可设置镜像模式。
- mirrorMode提供三种模式：OnlyFrontCameraPreviewEnabled、AllEnabled、AllDisabled。

b) 设置自动或者手动模式。

说明:

阿里云音视频通信iOS端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Publish模式： 如果您打开自动Publish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Publish模式，则需要您调用publish接口之后才会发布音视频流。
- 自动Subscribe模式： 如果您打开自动Subscribe模式，加入频道之后，SDK将会自动订阅当前频道内其他用户的音视频流；如果关闭自动Subscribe模式，则需要您调用subscribe接口之后才会订阅其他用户的音视频流。

```
/*
设置自动发布和订阅，只能在joinChannel之前配置
autoPublish: YES表示自动发布，NO表示手动发布
autoSubscribe: YES表示自动订阅，NO表示手动订阅
*/
[self.engine setAutoPublish:YES withAutoSubscribe:YES];
```

2. 加入频道。

说明:

AliRtcAuthInfo的各项参数均需要App Server通过API来获取，然后App Server下发至客户端，客户端将各项参数赋值后，就可以加入频道。

- name：无需APP Server下发。
- channelID 、userID、name命名要求：字符内容只允许[A-Za-z0-9_-]，长度限制64字节，非法命名系统将拒绝提供服务。

```
AliRtcAuthInfo *authinfo = [[AliRtcAuthInfo alloc]init];
authinfo.channel   = /* 您的channelId */;
authinfo.appid     = /* 您的Appid */;
authinfo.token     = /* 您的token */;
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
```

```
}];
```

3. 发布或取消发布本地流。

   发布本地流。

   - 自动pub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。
   - 非自动pub模式下：加入成功后，可通过以下接口发布本地流。

   如果publish过程中需要变更配置或者停止publish，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:AliRtcVideoTrackCame
ra];
[self.engine publish:^(int err) {
    // publish相关UI操作
}];
```

   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) {
    // publish取消相关UI操作
}];
```

4. 订阅或取消订阅远程流。

   订阅远程流。

   - 自动sub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。
   - 非自动sub模式下：加入频道成功后，可通过以下接口订阅远端流。

   如果subscribe过程中需要变更配置或者停止subscribe，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)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 = (AliRenderView *)view;/* 渲染view */
        [self.engine setRemoteViewConfig:canvas uid:uid forTrack:
AliRtcVideoTrackCamera];
      }
    });
}
```

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)audioTrack 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];
```

您可以下载示例代码，快速跑通Demo，实现频道内和其他用户进行实时音视频通话，详情请参见#unique_33。

6.3 Mac

阿里云音视频通信的基本功能包含初始化SDK、加入频道、本地发布和订阅远端、离开频道等。当您成功初始化SDK，您可以进行本地预览视频功能，进行简单的预览和测试，您也可以设置手动或者自动模式。

前提条件

在实现基本功能前，请您确保下载最新SDK，请参见#unique_19。

说明:

本文中的实现方法为主要功能方法，仅供参考，您可以根据您的业务需求进行实际开发。

操作步骤

1. 请您先初始化SDK。

   首先您需要创建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
   ```

   创建SDK实例并注册delegate。

   Mac回调详情请参见#unique_35。

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

   说明:

   目前SDK暂不支持多实例。

   a) 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas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 = AliRtcRenderMirrorModeOnlyFrontCameraPre
   viewEnabled;
   [self.engine setLocalViewConfig:canvas forTrack:AliRtcVide
   oTrackCamera];
   [self.engine startPreview];
   ```

   说明:

   · renderMode提供四种模式：Auto、Stretch、Fill、Crop，建议使用Auto模式。

- view必须是AliRenderView或者其子类。
- mirrorMode提供两种模式：AliRtcRenderMirrorModeAllEnabled、AliRtcRenderMirrorModeAllDisabled。

您也可以取消本地预览。

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

b) 设置自动或者手动模式。

阿里云音视频通信Mac端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Publish模式： 如果您打开自动Publish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Publish模式，则需要您调用publish接口之后才会发布音视频流。
- 自动Subscribe模式： 如果您打开自动Subscribe模式，加入频道之后，SDK将会自动订阅当前频道内其他用户的音视频流；如果关闭自动Subscribe模式，则需要您调用subscribe接口之后才会订阅其他用户的音视频流。

```
/*
设置自动发布和订阅，只能在joinChannel之前配置
autoPublish: YES表示自动发布，NO表示手动发布
autoSubscribe: YES表示自动订阅，NO表示手动订阅
*/
[self.engine setAutoPublish:YES withAutoSubscribe:YES];
```

2. 加入频道。

 说明:

AliRtcAuthInfo的各项参数均需要App Server通过API来获取，然后App Server下发至客户端，客户端将各项参数赋值后，就可以加入频道。

- name：无需APP Server下发。
- channelID 、userID、name命名要求：字符内容只允许[A-Za-z0-9_-]，长度限制64字节，非法命名系统将拒绝提供服务。

```
AliRtcAuthInfo *authinfo = [[AliRtcAuthInfo alloc]init];
authinfo.channel   = /* 您的channelId */;
authinfo.appid     = /* 您的Appid */;
authinfo.token     = /* 您的token */;
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
```

```
}];
```

3. 发布或取消发布本地流。

   发布本地流。

   · 自动pub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。

   · 非自动pub模式下：加入成功后，可通过以下接口发布本地流。

   如果publish过程中需要变更配置或者停止publish，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:AliRtcVideoTrackCame
ra];
[self.engine publish:^(int err) {
    // publish相关UI操作
}];
```

   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) {
    // publish取消相关UI操作
}];
```

4. 订阅或取消订阅远程流。

   订阅远程流。

   · 自动sub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。

   · 非自动sub模式下：加入频道成功后，可通过以下接口订阅远端流。

   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)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 = (AliRenderView *)view;/* 渲染view */
        [self.engine setRemoteViewConfig:canvas uid:uid forTrack:
AliRtcVideoTrackCamera];
      }
```

```
    });
}
```

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)audioTrack 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];
```

您可以下载示例代码，快速跑通Demo，实现频道内和其他用户进行实时音视频通话，详情请参见#unique_36。

6.4 Windows

阿里云音视频通信的基本功能包含初始化SDK、加入频道、本地发布和订阅远端、离开频道等。当您成功初始化SDK，您可以进行本地预览视频功能，进行简单的预览和测试，您也可以设置手动或者自动模式。

前提条件

在实现基本功能前，请您确保下载最新SDK，请参见#unique_19。

说明:

本文中的实现方法为主要功能方法，仅供参考，您可以根据您的业务需求进行实际开发。

操作步骤

1. 初始化SDK。

如果您在CRtcSampleDlg中持有AliRTCEngine实例。代码如下所示。

```
class CRtcSampleDlg : public AliRtcEventListener
{
...
```

```
    AliRtcEngine *m_pEngine;
...
}
```

您需要创建AliRTCEngine实例，并注册AliRtcEventListener监听相关回调，详细回调方法请参见#unique_37。

- 第一个参数是您实现的AliRtcEventListener对象，用于接收SDK的回调。
- 第二个参数是SDK初始化配置，当前版本请使用空字符串。

```
m_pEngine = AliRtcEngine::sharedInstance(/*AliRtcEventListener实例*/, /* 配置参数*/);
```

说明:

暂不支持多实例

a) 本地预览。在创建完AliRtcEngine实例后，您可以创建canvas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: 镜像画面 false: 正常画面;//
```

说明:

renderMode提供四种模式：Auto、Stretch、Fill、Crop，建议您使用Auto模式。hWnd必须是预览窗口句柄。

停止本地预览。

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

a) 设置自动或者手动模式。

- 自动Publish模式： 如果您打开自动Publish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Publish模式，则需要您调用publish接口之后才会发布音视频流。
- 自动Subscribe模式： 如果您打开自动Subscribe模式，加入频道之后，SDK将会自动订阅当前频道内其他用户的音视频流；如果关闭自动Subscribe模式，则需要您调用subscribe接口之后才会订阅其他用户的音视频流。

```
/*
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(true, true);
```

2. 加入频道。

说明:

AliRtcAuthInfo的各项参数均需要App Server通过API来获取，然后App Server下发至客户端，客户端将各项参数赋值后，就可以加入频道。

- name：无需APP Server下发。
- channelID 、userID、name命名要求：字符内容只允许[A-Za-z0-9_-]，长度限制64字节，非法命名系统将拒绝提供服务。

```
AliRtcAuthInfo authinfo;
authinfo.channel   = /* 用户的channelId */;
authinfo.appid     = /* 用户的Appid */;
authinfo.token     = /* 用户的token */;
authinfo.nonce     = /* 用户的nonce */;
authinfo.user_id   = /* 用户的userId */;
authinfo.token     = /* 用户的token */;
authinfo.timestamp = /* 用户的timestamp */;
authinfo.gslb      = /* 用户的gslb */;
...
// joinChannel是一个异步接口，设置回调函数
auto onJoinResult = [](void *opaque, int errCode) {
    // join成功
    if (errCode == 0) {
        // 加入频道成功
    } else {
        //加入频道失败
    }
};
m_pEngine->joinChannel(authinfo, /* name */, onJoinResult, /*
UserData，传递给onJoinResult回调函数*/);
```

3. 发布或取消发布本地流。

发布本地流。

- 自动pub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。
- 非自动pub模式下：加入成功后，可通过以下接口发布本地流。

如果publish过程中需要变更配置或者停止publish，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
m_pEngine->configLocalSimulcast(true, AliRtcVideoTrack::AliRtcVide
oTrackCamera); //子码流
//屏幕共享流不支持子码流，因此第二个参数只能为AliRtcVideoTrack::AliRtcVide
oTrackCamera。
// 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::AliRtcVide
oTrackCamera);
// Call back when publish finished
auto onPubResult = [](void *opaque, int errCode) {
    if (errCode == 0) {
        // 取消发布成功
    } else {
        // 取消发布失败
    }
};
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sub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。
   - 非自动sub模式下：加入频道成功后，可通过以下接口订阅远端流。

   如果subscribe过程中需要变更配置或者停止subscribe，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：true表示优先订阅大流
//第三个参数：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, AliRtcAudioTrack at) {
};
```

```
m_pEngine->subscribe(/* remoteUserID */, onSubResult, /*UserData*/);
```

不论自动模式还是非自动模式下，订阅成功后，通过订阅的callback，然后您可以进行相关UI操作或逻辑处理

```
auto onSubResult = [](void *opaque, const AliRtc::String &uid,
AliRtcVideoTrack vt, AliRtcAudioTrack at) {
    ...
    // 设置视频流的窗口
    if (vt == AliRtcVideoTrack::AliRtcVideoTrackCamera || vt ==
AliRtcVideoTrack::AliRtcVideoTrackBoth) {
        /*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subscribe操作，或者更新UI等。

```
 void onRemoteTrackAvailableNotify(const AliRtc::String & uid,
AliRtcAudioTrack audioTrack, 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, AliRtcAudioTrack at) {
};
m_pEngine->subscribe(/* remoteUserID */, onSubResult, /*UserData*/);
```

5. 离开频道。

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

您可以下载示例代码，快速跑通Demo，实现频道内和其他用户进行实时音视频通话，详情请参见#unique_38。

6.5 Web

阿里云音视频通信的基本功能包含创建实例、加入频道、本地发布和订阅远端、离开频道等。当您成功创建实例，您可以进行本地预览视频功能，进行简单的预览和测试。

前提条件

您需要集成WebSDK，详情请参见集成WebSDK。

操作步骤

1. 创建AliRtcEngine。

```
var aliWebrtc = new AliRtcEngine();     // 不支持多实例
```

a) 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，只能由数字、字母、下划线组成
    channel,         // 频道
    appid,           // 应用id
    nonce,           // nonce
    timestamp,       // 时间戳
    gslb,            // gslb
},displayName).then((obj)=>{
    // 入会成功
} ,(error)=>{
    // 入会失败，这里console下error内容，可以看到失败原因
    console.log(error.message);
});
```

说明:

joinChannel需要传递两个参数。第一个是频道鉴权令牌信息，需要您搭建App Server调用音视频通信的API去获取；第二个参数是频道里显示的用户名。

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，需要调用publish接口，发布本地流，远程会接收到onPublisher事件。

```
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事件。

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

```
});
```

4. 订阅onPublisher事件或onUnPublisher事件。

   · 订阅onPublisher事件。当远程人员推流时，在SDK里会触发onPublisher事件， 通过订阅这个事件，能够得到频道里已经推流的人员。

```
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。
    console.log(publisher.streamConfigs);
  });
```

   · 订阅onUnPublisher事件。当远程用户结束推流时，会触发这个事件。

```
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事件只有加入频道以后才会触发。

5. 订阅或取消订阅远程流。

   · 订阅远程流。userId是用户Id。

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

   成功订阅远程流以后。 订阅成功到和远程人员建立链接是一个异步的过程，需要订阅onMediaStream事件，得到远程流的stream对象，通过video元素播放。

```
aliwebrtc.on('onMediaStream',(subscriber, stream)=>{
    var video = document.getElementByTag('video');
    aliwebrtc.setDisplayRemoteVideo(
        subscriber,         // onMediaStream中返回的参数
        video,              // html中用于显示stream对象的video元素
        stream              // onMediaStream中返回的参数
    )
});
```

   · 显示远程流。在onMediaStream消息中，调用setDisplayRemoteVideo来显示远程流。

```
aliwebrtc.setDisplayRemoteVideo(
    subscriber,         // onMediaStream中返回的参数
    video,                // html中用于显示stream对象的video元素
    stream              // onMediaStream中返回的参数
```

```
)
```

- 取消订阅。通过unSubscribe可以取消订阅远程流。userId是用户Id。

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

6. 离开频道。

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

获取示例代码，您可以通过单击音视频通信WEB SDK使用申请。