ALIBABA CLOUD

阿里云

音视频通信

常见问题

文档版本：20201014

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.Android SDK

本文为您介绍了集成SDK时，集成工具报错的处理方法，帮助您快速定位问题，并集成SDK。

Android SDK

gradle中未正确配置对RTC库的引用

解决办法：

请按照正确步骤导入aar包和jar包，并在gradle中配置引用，详情请参见集成Android SDK。

隐私权限未申请

解决办法：

- 您需要添加摄像头、麦克风、网络，访问存储权限。在*AndroidManifest.xml*文件中添加权限。

```
<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"/>
```

- 您需要在代码里动态申请权限。

未在主线程初始化SDK

解决办法：

初始化 `AliRtcEngine` 实例，并注册回调。相关回调有 `AliRtcEngineEventListener` 和 `AliRtcEngineNotify` ，并且只能在主线程调用，详情请参见回调及监听。

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

弱网情况下人声有卡顿

为了保证合唱的实时性，客户端采用了低延时策略，弱网下丢包率会相应增加。

开启耳返模式下，声音外放出现回声

您需要带上耳机然后进行合唱，不能通过外放。

2.iOS SDK

本文为您介绍了集成SDK时，集成工具报错的处理方法，帮助您快速定位问题，并集成SDK。

编译时报x86或i386错误

解决办法：

iOS SDK目前暂不支持模拟器，请使用真机调试和运行。

bitcode错误

解决办法：

SDK暂不支持bitcode配置，请关闭bitcode编译选项。

image not found

解决办法：

SDK从1.7版本开始切换为动态库，将*AliRTCSdk.framework*加载到Embedded Binaries中。

隐私权限错误

解决办法：

您需要编辑*info.plist*，申请摄像头、麦克风权限。具体操作请参见集成iOS SDK。

UIDevice方法调用错误

解决办法：

添加-ObjC到链接选项里。

弱网情况下人声有卡顿

为了保证合唱的实时性，客户端采用了低延时策略，弱网下丢包率会相应增加。

开启耳返模式下，声音外放出现回声

您需要带上耳机然后进行合唱，不能通过外放。

3.Mac SDK

本文为您介绍了集成SDK时，集成工具报错的处理方法，帮助您快速定位问题，并集成SDK。

Mac

bitcode错误

解决办法：

SDK暂不支持bitcode配置，请关闭bitcode编译选项。

UTDID image not found

解决办法：

将*UTDID.framework*加载到Embedded Binaries中。具体操作请参见集成Mac SDK。

参数不符合规范

解决办法：

根据规范，请您重新对channelID、userID、name进行命名。字符内容只允许[A-Za-z0-9_-]，长度限制64字节。

4.Windows SDK

本文为您介绍了集成SDK时，集成工具报错的处理方法，帮助您快速定位问题，并集成SDK。

Windows SDK

x64编译报错

解决办法：

SDK目前只支持32位，请切换编译选项。

头文件或静态库路径设置错误

解决办法：

设置头文件和静态库路径。

动态库路径错误

解决办法：

复制*AliRTCSdk.dll*及依赖的*ffmpeg.dll*到程序的执行路径下。

5.Web SDK

本文为您介绍了集成SDK时，集成工具报错的处理方法，帮助您快速定位问题，并集成SDK。

Web SDK 常见问题

摄像头和麦克风无法使用

解决办法：

- AppServer和网页需要使用https协议。
- 检测是否禁用或者占用摄像头和麦克风设备。

Web端和其它端无法互通

解决办法：

需要在其他端调用setH5CompatibleMode，设置兼容模式。

Android设备不能进行Web通信

解决办法：

检测Android设备是否支持H264协议。

> 说明 Android设备进行Web页面通信，需要安卓设备支持H264协议。

Web SDK无法内嵌App内

Web SDK不支持内嵌App内。

Web SDK是否支持小程序

web SDK 暂不支持微信小程序、支付宝小程序。

无法显示本地摄像头

解决办法：

- AppServer和网页需要使用https协议。
- 查看设备是否被禁用。
- 检查硬件是否可用。

SDK如果只使用音频，是否可以没有摄像头

如果只使用音频，可以没有摄像头。

6.客户端入会失败常见原因

本章节为您介绍了客户端入会失败常见原因及解决办法。

客户端入会 rtc

Q：加入频道所需参数如何获取？

A：加入频道所需参数需要从AppServer获取，获取参数请参见服务端生成Token。

Q：sessionID如何获取？

A：1.9以下版本SDK中的sessionID字段可以置为空字符串，1.9及以上版本已经删除该字段。

Q：常见典型错误码有哪些？

A：

- 错误码33620485：出现这种错误码的含义是用户ID错误，导致Token计算失败，用户ID需从AppServer下发。
- 错误码33630483：出现这种错误码是因为频道已经过期，需要重新创建频道。频道过期时长为48小时，您可以调用updateChannel更新频道。更多接口请参见API概览。

7.Android切换横竖屏实现方法

当您成功集成SDK，并想实现移动端切换横竖屏进行实时音视频通信。您可以阅读本文，了解实现本地切换横竖屏的代码方法，帮助您更好的体验阿里云音视频通信服务。

横竖屏模式切换

正常情况下竖屏模式推流分辨率宽<高，例如：480*640；横屏模式推流分辨率宽>高，例如：640*480。

调用setDeviceOrientationMode方法，进行切换横竖屏：

```
//接口方法
public abstract void setDeviceOrientationMode(AliRtcEngine.AliRtcOrientationMode mode);
//示例方法
mAliRtcEngine.setDeviceOrientationMode(AliRtcOrientationModePortrait);
```

参数	类型	描述
mode	AliRtcOrientationMode	设备方向。取值： • AliRtcOrientationModePortrait（默认值）：固定竖屏模式。 • AliRtcOrientationModeLandscapeLeft：固定左横屏模式。 • AliRtcOrientationModeLandscapeRight：固定右横屏模式。 • AliRtcOrientationModeAuto：自适应模式。

> 说明
> - 当应用切换横竖屏时，调用此接口进行设备方向切换，摄像头采集会随机进行切换。
> - 竖屏模式时不需要调用此接口。
> - 1.17之前版本仅支持固定竖屏模横式，即只要当前未打开摄像头采集（未开启预览并且未开始视频推流），设置可生效。打开摄像头后再调用该接口不会生效，不支持动态横竖屏切换。

如果您的手机不支持自适应模式，而您想要设置自适应模式，您需要监听旋转的方向，然后根据角度设置当前的横竖屏。具体操作如下：

1. 设置自适应模式。

```
//设置横屏竖屏自适应模式。
 mAliRtcEngine.setDeviceOrientationMode(AliRtcOrientationModeAuto);
 setRequestedOrientation(SCREEN_ORIENTATION_UNSPECIFIED);
```

2. 设置setRequestedOrientation监听旋转角度。

```
/**
 * 监听旋转角度
 */
private OrientationEventListener mOrientationEventListener;
   @Override
```

```
protected void onResume() {
    if (null==mOrientationEventListener) {
        mOrientationEventListener = new OrientationEventListener(this) {
            @Override
            public void onOrientationChanged(int orientation) {
                if (orientation == OrientationEventListener.ORIENTATION_UNKNOWN) {
                    return; //手机平放时，检测不到有效的角度。
                }
                //备注：如果您的应用有固定横竖屏模式和自适应模式切换。请添加判断语句，只有自适应模式才根据角度设置横竖屏方向。

                //只检测是否有四个角度的改变，设置自适应模式后，只需要修改setRequestedOrientation即可。
                if (orientation > 350 || orientation < 10) { //0度，竖直。
                    setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
                } else if (orientation > 80 && orientation < 100) { //90度，右横屏。
                    setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_REVERSE_LANDSCAPE);
                } else if (orientation > 170 && orientation < 190) { //180度，倒立。
                    setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
                } else if (orientation > 260 && orientation < 280) { //270度，左横屏。
                    setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE);
                } else {
                    return;
                }
            }
        };
        mOrientationEventListener.enable();
    }
}

@Override
protected void onPause() {
    super.onPause();
    //停止监听。
    if (null != mOrientationEventListener) {
        mOrientationEventListener.disable();
        mOrientationEventListener = null;
    }
}
```

8.iOS切换横竖屏实现方法

当您成功集成SDK，并想实现移动端切换横竖屏进行实时音视频通信。您可以阅读本文，了解实现本地切换横竖屏的代码方法，帮助您更好的体验阿里云音视频通信服务。

横竖屏模式切换

正常情况下竖屏模式推流分辨率宽<高，例如：480*640；横屏模式推流分辨率宽>高，例如：640*480。

如果您想切换横竖屏，请调用setDeviceOrientationMode方法进行切换横竖屏。该方法调用成功返回0，失败返回其他。

> 说明 仅允许在推流和预览之前进行设置。

```
//接口方法
- (int)setDeviceOrientationMode:(AliRtcOrientationMode)mode;
//示例方法
[[UIDevice currentDevice] setValue:@(UIDeviceOrientationLandscapeLeft) forKey:@"orientation"];
[self.engine setDeviceOrientationMode:(AliRtcOrientationModeLandscapeLeft)];
```

参数	类型	描述
mode	AliRtcOrientationMode	设备方向。取值： • AliRtcOrientationModePortrait（默认值）：固定竖屏模式。 • AliRtcOrientationModeLandscapeLeft：固定左横屏模式。 • AliRtcOrientationModeLandscapeRight：固定右横屏模式。 • AliRtcOrientationModeAuto：自适应横竖屏模式。

> 说明
> • 当应用切换横竖屏时，调用此接口进行设备方向切换，摄像头采集会随机进行切换。
> • 1.17之前版本仅支持固定竖屏模式，不支持动态横竖屏切换，即只要当前未打开摄像头采集（未开启预览并且未开始推视频流），设置可生效；打开摄像头后再设置不生效。

竖屏模式切换推流分辨率宽高

竖屏模式推流分辨率宽>高，例如：640*480（摄像头保持竖屏采集）。

您可以调用setVideoSwapWidthAndHeight方法切换分辨率。

> 说明 请您在调用setVideoProfile和joinChannel之前进行切换。

```
- (void)setVideoSwapWidthAndHeight:(BOOL)swapWidthAndHeight forTrack:(AliRtcVideoTrack)track;
```

参数	类型	描述
swapWidthAndHeight	BOOL	是否交换宽和高，取值： YES|NO（默认值）
track	AliRtcVideoTrack	视频Track类型

说明 您可以在以下情况调用该接口进行切换宽和高：

- 竖屏模式下竖屏推流，推流分辨率需要宽>高。
- 横屏模式下横屏推流，推流分辨率需要宽<高。

9.客户端与服务端连通异常

本章节为您介绍了客户端与服务端连通异常常见原因及解决办法。

常见原因

阿里云RTC需要使用UDP端口80、443、3478、50000。如果与服务端连通出现异常，可能是因为打开了防火墙或路由器策略限制导致。

解决办法

Windows和Linux平台下可以使用netcat命令测试UDP端口是否连通。

检查防火墙或路由器是否限制了UDP端口。

10.H5纯订阅模式媒体文件播放失败问题

您可以通过阅读本文，了解H5纯订阅模式下可能会遇到浏览器页面不能自动播放媒体文件的解决办法。

常见原因

浏览器为了防止网页在用户非自愿的情况下主动播放声音，对网页上的自动播放（Autoplay）功能做了限制：浏览器在没有用户交互操作之前不允许有声音的媒体播放。

受浏览器策略影响，Chrome 70+、Safari、Firefox等浏览器新版本都不支持带声音的媒体文件自动播放，需要您在网页上手动触发才能播放媒体文件。

解决方法

您可以使用如下两种方法，解决浏览器不能播放媒体文件的问题。

- 在onError中获取到错误码10201时，会同时返回播放失败的 `userId` ，此时该用户的音频播放是静音的，在网页上手动触发事件（有用户交互）调用 `aliWebrtc.muteAllRemoteAudioPlaying(false)` 取消静音。
- 如果仅有几个人使用浏览器进行播放媒体文件，例如视频直播网站的管理员，建议您可以进行浏览器设置来播放媒体文件。

> **说明** 每次打开浏览器都需要引导用户进行单击操作，而通过浏览器设置可以一次性解决这个域名下所有页面自动播放的问题，您可以参考下文进行浏览器设置（浏览器版本不同，设置方法可能不同，本文的浏览器设置方法仅供参考，具体操作请您以实际为准）。

Chrome浏览器

您需要访问Chrome浏览器的网站设置，然后将声音项改为允许。

1. 单击浏览器的**网站设置**。

2. 将声音更改为**允许**。

Safari浏览器

在Safari浏览器的设置中，将自动播放选项改为允许全部自动播放。

1. 在网址栏单击**此网站的设置...**。

2. 将自动播放更改为**允许全部自动播放**。

Firefox浏览器

您可以在保护设置中，将自动播放修改成允许音频和视频。

1. 在地址栏单击**保护设置**。

2. 单击自动播放的**设置**。

3. 更改为允许音频和视频。

11.H5端如何实现镜像和显示手机横屏录制的视频

当您成功集成SDK，并想实现H5端镜像和显示横屏录制画面时，您可以通过本章节了解代码方法。

解决方案

> 说明 Web SDK本身不提供镜像和横竖屏接口，需要您自己实现。

- 实现镜像：通过给video标签设置css，来实现镜像功能。

```
{
    transform: rotateY(180deg);
    -webkit-transform: rotateY(180deg);
    -moz-transform: rotateY(180deg);
}
```

- 显示横屏录制画面：在video标签外增加一个div标签，并在div上设置css，来显示远端横屏录制的画面。

```
{
  transform: rotate(90deg);
  -webkit-transform: rotate(90deg);
  -moz-transform: rotate(90deg);
}
```

12.AudioSession多模块共享问题

本文为您介绍了iOS平台如何避免同一个应用中，RTC SDK和其他音频模块对AVAudioSession的竞争的解决方案。

问题描述

默认情况下，RTC SDK和App对AVAudioSession都有控制权，为了保证RTC SDK通话功能的正常使用，RTC SDK会提高对AVAudioSession使用的优先级。在某些场景下，例如：当需要暂停RTC SDK并且使用其他音频组件（音乐播放器、其他第三方音频组件等），App会希望限制RTC SDK对AVAudioSession的控制权限。

解决方案

RTC SDK为您提供以下接口方法：

```
- (int)setAudioSessionOperationRestriction:(AliRtcAudioSessionOperationRestriction)restriction;
```

该方法用来限制RTC SDK对Audio Session的管控权限，App也可以随时使用这个方法把管理权限再交还给RTC SDK。

> **说明** 如果您调用了该方法限制了RTC SDK的管理权限，则需要App自行维护以保证SDK功能正常。

13.旁路转推纯音频设置

您可以阅读本文，快速了解调用startMpuTask开始旁路转推纯音频模式的设置方法，完成旁路转推纯音频模式配置。

纯音频模式和音视频模式的旁路转推参数配置类似，区别在于LayoutIds（布局），MediaEncode（编码选项）及TaskProfile（任务计费配置）参数的配置。

在纯音频模式下以上参数配置说明如下：

- LayoutIds：根据任务计费配置参数TaskProfile决定，TaskProfile设置为*Mixed_Audio*时不限制音频源数量。
- MediaEncode：只能设置为*0*。

14.SDK使用过程中如何减少耗时

本文为您介绍在使用SDK过程中减少耗时的方法。

配置合理的自动订阅/自动推流

以下建议可以为您减少耗时：

- 如果您的场景中，入会后会默认推流，建议您采用自动推流模式。
- 如果您的场景中，入会后默认会订阅频道中的人，建议您采用自动订阅的模式。自动订阅模式在接收到远端用户回调之后，不需要再手动订阅，能更加节省时间。

通过以下接口设置自动订阅/自动推流：

setAutoPublish：设置是否自动发布，是否自动订阅。默认是自动发布和订阅，必须在加入频道之前设置。

```
- (int)setAutoPublish:(BOOL)autoPub withAutoSubscribe:(BOOL)autoSub;
```

合理的使用publish/subscribe接口

以下建议可以为您减少耗时：

- publish/subscribe接口用于手动推流、手动订阅、以及会议中推流和订阅配置的变更。
- publish接口需要配合configLocalCameraPublish、configLocalScreenPublish、configLocalAudioPublish接口使用。
- subscribe接口需要配合configRemoteCameraTrack、configRemoteScreenTrack、configRemoteAudio接口使用。
- 根据自身的需要，合理使用大小流。目前默认推小流，如果无需小流，建议关闭。
- 需要注意：每次config完之后调用一次publish接口或者subscribe接口就可以，无需重复调用。

  如：当前相机流(大流)和音频流需要调用publish接口，而屏幕共享流不需要调用。

  - 正确的调用方式如下：

```
[self.engine configLocalAudioPublish:YES];
[self.engine configLocalCameraPublish:YES];
[self.engine configLocalScreenPublish:NO];
[self.engine configLocalSimulcast:NO forTrack:AliRtcVideoTrackCamera];
[self.engine publish:^(int err) {
    // 相关UI操作。
}];
```

- 错误的调用方式如下：

```
[self.engine configLocalAudioPublish:YES];
[self.engine publish:^(int err) {
    // 相关UI操作。
}];

[self.engine configLocalCameraPublish:YES];
[self.engine publish:^(int err) {
    // 相关UI操作。
}];

[self.engine configLocalScreenPublish:NO];
[self.engine publish:^(int err) {
    // 相关UI操作。
}];

[self.engine configLocalSimulcast:NO forTrack:AliRtcVideoTrackCamera];
[self.engine publish:^(int err) {
    // 相关UI操作。
}];
```

场景中存在频繁切换频道或者频繁入离会

以下建议可以为您减少耗时：

- 维护一个SDK的示例，避免频繁的创建和销毁sdk带来的耗时。通过一个SDK实例的入离会来达到切换频道的效果。
- 可以根据需要，提前打开音频设备，节省每次入离会的耗时。比如：

  在语音直播间的场景，可以在创建SDK后调用一次startAudioPlayer接口提前打开播放设备。在销毁sdk前调用stopAudioPlayer接口关闭播放设备。中间切换频道和入离会均无需再额外调用。

  startAudioPlayer：开启音频播放。您可以控制提前打开音频播放，如果不设置，SDK会在订阅成功的时候打开音频播放。

  ```
  - (void)startAudioPlayer;
  ```

  stopAudioPlayer：关闭音频播放。您可以控制关闭音频播放。

  ```
  - (void)stopAudioPlayer;
  ```

- 音频采集设备也可以根据自身的场景决定是否提前打开。

  startAudioCapture：开启音频采集。您可以控制提前打开音频采集，如果不设置，SDK会在开始推流的时候打开音频采集。

  ```
  - (void)startAudioCapture;
  ```

stopAudioCapture：关闭音频采集。您可以控制关闭音频采集。

```
- (void)stopAudioCapture;
```

15.手动订阅和手动推流说明

15.1. Android

通过本章节，您可以了解手动推流和手动订阅的实现步骤。

背景信息

RTC SDK提供了setAutoPublish接口，支持设置自动推流或自动订阅，设置自动推流时SDK会在加入频道后自动推流，设置自动订阅时SDK会在频道中有其他用户推流时自动进行订阅拉流，无需App进行操作。

如果您需要根据具体业务场景，按需进行推流或者拉流，可以使用手动推流或手动订阅方式。

手动推流

1. 创建引擎实例后，在加入频道前，设置关闭自动推流和自动订阅。

```
// 创建引擎实例
AliRtcEngine mAliRtcEngine = AliRtcEngine.getInstance(getApplicationContext());
 mAliRtcEngine.setRtcEngineEventListener(new AliRtcEngineEventListener() {


        @Override
        public void onPublishResult(int result, String publishId) {
           //异步推流的回调
        }


       @Override
        public void onUnpublishResult(int result) {
         //异步取消推流的回调
        }


});
// 关闭自动推流和自动订阅
mAliRtcEngine.setAutoPublish(false, false);
```

2. 配置后加入频道，并根据业务需要启动推流（音频/摄像头/屏幕分享），并通过回调确认推流是否成功，如果推流失败需要进行重试或用户提示。

```
// 配置开启推送音频流
mAliRtcEngine.configLocalAudioPublish(true);

// 配置开启推送摄像头流
mAliRtcEngine.configLocalCameraPublish(true);

// 配置开启推送屏幕分享流
mAliRtcEngine.configLocalScreenPublish(true);

// 启动推流
mAliRtcEngine.publish();
```

> **说明** publish接口是异步接口，您在调用之后需要收到onPublishResult回调且result为*0*表示推流成功，接口及对应回调请参见setRtcEngineEventListener。

3. 当业务场景需要停止推流时，您需要关闭步骤2配置的媒体流并执行停止推流。

```
// 配置停止推送音频流
mAliRtcEngine.configLocalAudioPublish(false);

// 配置停止推送摄像头流
mAliRtcEngine.configLocalCameraPublish(false);

// 配置停止推送屏幕分享流
mAliRtcEngine.configLocalScreenPublish(false);

// 停止推流
mAliRtcEngine.publish();
```

手动订阅

1. 创建引擎实例后，在加入频道前，设置关闭自动订阅。

```
// 创建引擎实例
AliRtcEngine mAliRtcEngine = AliRtcEngine.getInstance(getApplicationContext());
 mAliRtcEngine.setRtcEngineEventListener(new AliRtcEngineEventListener() {

            @Override
            public void onSubscribeResult(String uid, int result, AliRtcEngine.AliRtcVideoTrack vt,
                          AliRtcEngine.AliRtcAudioTrack at) {
              //异步订阅成功或者失败的回调
            }

            @Override
           public void onUnsubscribeResult(int result, String userId) {
              Log.d(TAG, "onUnsubscribeResult result : " + result);
              //取消订阅成功或者失败的回调
           }

          @Override
         public void onRemoteTrackAvailableNotify(String uid, AliRtcEngine.AliRtcAudioTrack audioT
rack,
                                                  AliRtcEngine.AliRtcVideoTrack videoTrack) {
            //远端用户推流发生变化时回调
         }
});
// 关闭自动推流和自动订阅
mAliRtcEngine.setAutoPublish(false, false);
```

2. 配置后并成功加入频道，当频道中有用户推流发送变化时，SDK会收到onRemoteTrackAvailableNotify回调进行通知，回调中指明了推流用户及推送的媒体流。

   onRemoteTrackAvailableNotify回调中指明了推流用户ID，以及该用户发生状态变更的媒体流，您需要记录该用户已推送媒体流：

   - audioTrack为AliRtcAudioTrackMic时，表示该远端用户推送了音频流。
   - videoTrack为AliRtcVideoTrackCamera或AliRtcVideoTrackBoth时，表示该远端用户推送了摄像头视频流。
   - videoTrack为AliRtcVideoTrackScreen或AliRtcVideoTrackBoth时，表示该远端用户推送了屏幕分享视频流。
   - audioTrack为AliRtcAudioTrackNo并且videoTrack为AliRtcVideoTrackNo时，表示该远端用户已停止推流。

```
// 接收到其他用户推流变化通知
@Override
public void onRemoteTrackAvailableNotify(String uid, AliRtcEngine.AliRtcAudioTrack audioTrack,
                                         AliRtcEngine.AliRtcVideoTrack videoTrack) {


}
```

> 说明 必须在加入频道成功后，您才会接收到频道中其他用户推流的onRemoteTrackAvailableNotify回调，您需要查看加入频道结果是否成功，加入频道回调请参见onJoinChannelResult。

3. 接收到推流回调后，可根据业务需要对用户实际推送的媒体流进行配置，然后启动订阅。

```
// 配置是否订阅指定用户音频流，可根据需要设置开启/关闭
mAliRtcEngine.configRemoteAudio(userId, true|false);

// 配置是否订阅指定用户摄像头流，可根据需要设置开启/关闭
mAliRtcEngine.configRemoteCameraTrack(userId, false, true|false);

// 配置是否订阅指定用户屏幕分享流，可根据需要设置开启/关闭
mAliRtcEngine.configRemoteScreenTrack(userId, true|false);

// 启动订阅
mAliRtcEngine.subscribe(userId);
```

> 说明 subscribe接口是异步接口，需要收到onSubscribeResult回调，详情请参见onSubscribeResult。
>
> - 必须在接收到频道内用户的推流回调之后，才能对该用户进行订阅拉流，否则订阅失败；如果频道内用户状态已变为停止推流，订阅无法成功。
> - 您需要关注回调中远端用户实际推送的媒体流，对于没有推送的媒体流，订阅无法成功。
> - 如果用户状态已变为停止推流后，不能在对该用户执行订阅操作，直到再次接收到回调（该用户重新开始推送媒体流）。

4. 订阅成功后，当业务场景需要取消订阅指定用户媒体流，或远端用户已停止推流（参考步骤2中远端停止推流状态）时，配置并取消订阅对应用户。

```
// 配置是否订阅指定用户音频流，可根据需要设置关闭
mAliRtcEngine.configRemoteAudio(userId, false);

// 配置是否订阅指定用户摄像头流，可根据需要设置关闭
mAliRtcEngine.configRemoteCameraTrack(userId, false, false);

// 配置是否订阅指定用户屏幕分享流，可根据需要设置关闭
mAliRtcEngine.configRemoteScreenTrack(userId, false);

// 取消订阅
mAliRtcEngine.subscribe(userId);
```

15.2. Windows

通过阅读本文，您可以了解手动推流和手动订阅的实现步骤。

背景信息

RTC SDK提供了setAutoPublish接口，支持设置自动推流或自动订阅，设置自动推流时SDK会在加入频道后自动推流，设置自动订阅时SDK会在频道中有其他用户推流时自动进行订阅拉流，无需App进行操作。

如果您需要根据具体业务场景，按需进行推流或者拉流，可以使用手动推流或手动订阅方式。

推流端

1. 创建引擎实例后，在加入频道前，设置关闭自动推流（设置参数autoPub为*false*）。

```
// 创建引擎实例
AliRtcEngine *pEngine = AliRtcEngine::sharedInstance(this, "");

// 关闭自动推流/自动订阅
pEngine->setAutoPublishSubscribe(false, false);
```

2. 配置后加入频道，并根据业务需要启动推流（音频/摄像头/屏幕采集），并通过回调确认推流是否成功，如果推流失败需要进行重试或提示用户。

> **说明** 必须在加入频道成功后才能开始推流，否则推流会失败，您需要关注加入频道结果是否成功。

```
// 配置推送音频流
pEngine->configLocalAudioPublish(true);

// 配置推送摄像头流
pEngine->configLocalCameraPublish(true);

// 配置推送屏幕分享流
pEngine->configLocalScreenPublish(true);

// 启动推流
auto callback = [](void *opaque, int errorCode) {
   if (errorCode == 0) {
      // 推流成功
   } else {
      // 推流失败
   }
};
pEngine->publish(callback, this);
```

3. 当业务场景需要停止推流时，配置关闭上一步打开的媒体流并执行停止推流。

```
// 配置停止推送音频流
pEngine->configLocalAudioPublish(false);

// 配置停止推送摄像头流
pEngine->configLocalCameraPublish(false);

// 配置停止推送屏幕分享流
pEngine->configLocalScreenPublish(false);

// 停止推流
auto callback = [](void *opaque, int errorCode) {
};
pEngine->publish(callback, this);
```

订阅端

1. 创建引擎实例后，在加入频道前，设置关闭自动订阅（设置参数autoSub为*false*）。

```
// 创建引擎实例
AliRtcEngine *pEngine = AliRtcEngine::sharedInstance(this, "");

// 关闭自动推流/自动订阅
pEngine->setAutoPublishSubscribe(false, false);
```

2. 配置后并成功加入频道，当频道中用户推流发送变化时，SDK会回调onRemoteTrackAvailableNotify进行通知，回调中指明了推流用户及推送的媒体流。
   - audioTrack为AliRtcAudioTrackMic时，代表该远端用户推送了音频流。
   - videoTrack为AliRtcVideoTrackCamera或AliRtcVideoTrackBoth时，代表该远端用户推送了摄像头视频流。
   - videoTrack为AliRtcVideoTrackScreen或AliRtcVideoTrackBoth时，代表该远端用户推送了屏幕分享视频流。
   - audioTrack为AliRtcAudioTrackNo并且videoTrack为AliRtcVideoTrackNo时，代表该远端用户已停止推流。

```
// 接收到其他用户推流变化通知
void onRemoteTrackAvailableNotify(const AliRtc::String &uid,
    AliRtcAudioTrack audioTrack,
    AliRtcVideoTrack videoTrack)
{
    ...
}
```

> 说明
>
> - 在加入频道成功后，才会接收到频道中其他用户推流回调onRemoteTrackAvailableNotify，您需要关注加入频道结果是否成功。
> - 回调onRemoteTrackAvailableNotify中指明了推流用户ID，以及该用户发生状态变更的媒体流，您需要记录该用户已推送媒体流。

3. 接收到推流回调后，可根据远端用户实际推送的媒体流及业务需要，配置订阅推流用户的媒体流，然后启动订阅。

```
// 配置订阅指定用户音频流
pEngine->configRemoteAudio(userId, true);

// 配置订阅指定用户摄像头流
pEngine->configRemoteCameraTrack(userId, false, true);

// 配置订阅指定用户屏幕分享流
pEngine->configRemoteScreenTrack(userId, true);

// 启动订阅
auto callback = [](void *opaque, const AliRtc::String &uid, AliRtcVideoTrack vt, AliRtcAudioTrack at)
{
};
pEngine->subscribe(userId, callback, this);
```

> 说明
>
> - 在接收到远端用户的推流回调之后，您才能对该用户进行订阅拉流，否则订阅失败。
> - 您需要关注回调中远端用户实际推送的媒体流，对于没有推送的媒体流，订阅无法成功。
> - 如果远端用户状态已变为停止推流后，您不能对该用户执行订阅操作，直到再次接收到该用户重新开始推送媒体流的回调。

4. 订阅成功后，订阅用户会接收到回调onSubscribeChangedNotify，此时可设置用于显示视频流的View（视图）。
   - audioTrack为AliRtcAudioTrackMic时，代表已订阅远端用户推送了音频流。
   - videoTrack为AliRtcVideoTrackCamera或AliRtcVideoTrackBoth时，代表已订阅远端用户摄像头视频流。
   - videoTrack为AliRtcVideoTrackScreen或AliRtcVideoTrackBoth时，代表已订阅远端用户推送了屏幕分享视频流。

```
void onSubscribeChangedNotify(const AliRtc::String &uid,
    AliRtcAudioTrack audioTrack,
    AliRtcVideoTrack videoTrack) {

    if (at == AliRtcAudioTrackMic) {
        // 订阅音频流成功
    }

    if (vt == AliRtcVideoTrackCamera || vt == AliRtcVideoTrackBoth) {
        // 订阅摄像头流成功
        AliVideoCanvas cameraCanvas;
        ...
        pEngine->setRemoteViewConfig(&cameraCanvas, uid, AliRtcVideoTrackCamera);
    }

    if (vt == AliRtcVideoTrackScreen || vt == AliRtcVideoTrackBoth) {
        // 订阅屏幕流成功
        AliVideoCanvas screenCanvas;
        ...
        pEngine->setRemoteViewConfig(&screenCanvas, uid, AliRtcVideoTrackScreen);
    }
}
```

> 说明 onSubscribeChangedNotify回调中返回了实际订阅远端用户媒体流的结果，您需要关注订阅结果是否与订阅配置一致，确认订阅是否成功。

5. 当业务场景不需要订阅指定用户媒体流，或远端用户已停止推流时（参见步骤2中远端停止推流状态），配置并取消订阅对应用户。

```
// 配置取消订阅指定用户音频流
pEngine->configRemoteAudio(userId, false);

// 配置取消订阅指定用户摄像头流
pEngine->configRemoteCameraTrack(userId, false, false);

// 配置取消订阅指定用户屏幕分享流
pEngine->configRemoteScreenTrack(userId, false);

// 取消订阅
auto callback = [](void *opaque, const AliRtc::String &uid, AliRtcVideoTrack vt, AliRtcAudioTrack at)
{
};
pEngine->subscribe(userId, callback, this);
```

16.通讯模式升级至互动模式说明

您可以通过阅读本文，了解互动模式说明。

背景信息

针对部分业务场景中，单一频道存在较多用户人数，但主要时间内只有一名主播推流，少量观众需要与主播连麦互动的情况，RTC SDK加入了新的频道模式（互动模式AliRtcInteractiveLive）。

通讯模式与互动模式

通讯模式与互动模式的不同，主要体现在以下两点：

- 频道内最大支持人数不同，通信模式目前默认最多是支持50*50（即50个人同时推拉流），互动模式可以支持最多2*4000（即2个人同时推拉流，其余都是只拉流）。
- 通信模式没有用户角色的区分，互动模式下增加了用户角色的设置功能，每位加入频道的用户都具备用户角色，分别为：
  - 互动角色（AliRtcChannelProfileInteractiveLive）：允许推流，适用于频道中的主播与连麦用户。
  - 观看角色（AliRtcClientRolelive）：只能订阅拉流观看，不能推流，适用于普通观众用户。

  同时，所有观众角色只能接受到互动角色的加入频道，推流，停止推流，离开频道等状态通知，不会收到其余观众用户的信令消息。

> **说明**
>
> 同一个频道里同时只能使用一种频道模式，频道模式由最早加入频道的用户设置，后续加入的用户也需要使用对应的频道模式，否则加入频道失败。例如，最早一个用户使用互动模式进入频道成功后，后续用户如果再使用通信模式加入频道将会失败。当频道内所有用户都离开2分钟后，该频道的频道模式会被重置，将由之后首个加入频道的用户设置的频道模式重新确定。
>
> 由于前期SDK版本不支持频道模式设置，默认以通信模式加入频道。为避免出现上述因频道模式不匹配而加入失败问题，建议业务侧升级使用互动模式后，强制升级用户SDK版本，均升级完毕之后再启用，保证同一频道内用户频道模式的统一。

互动模式升级说明

从通信模式升级为互动模式时，推流端和订阅端需要作出以下接入改变：

推流端

- 主播在加入频道之前，先调用setChannelProfile接口设置频道模式为互动模式。
- 主播可以在加入频道前调用setClientRole接口，将用户角色设置为互动角色。主播入会之后自动推流，并且可以调整用户角色信息，当主播加入频道成功后保持自动推流。
- 主播也可以关闭自动推流模式，在加入频道后，切换角色为互动角色。当主播切换成功后会收到onUpdateRoleNotify回调通知，收到回调通知后可以配置需要推流媒体型，调用publish接口开始推流。

> **说明**
>
> 如果在加入频道成功前设置用户角色，默认初始角色即为设置角色，不会接收到角色变更回调。加入频道成功后再设置用户角色，设置为不同角色成功时会有回调通知。同时业务侧需要记录一下当前的角色参数值，重复设置当前角色不会收到回调。建议业务侧将角色切换和推拉流的逻辑区分开。
>
> 用户必须确保在加入频道成功的前提下，进行切换角色操作和推流动作，否则无法切换和推流成功。

订阅端

- 用户在加入频道之前，先调用setChannelProfile接口设置频道模式为互动模式。
- 用户可以在加入频道前调用setClientRole接口，将用户角色设置为观看角色，关闭自动推流，保持自动订阅。
- 当连麦观众，在加入频道成功后，如果业务场景需要连麦，将设置用户角色为互动角色，设置成功后会收到onUpdateRoleNotify回调通知，收到回调通知后可以配置需要推流媒体型，调用publish接口开始推流。

  当连麦观众下麦时，关闭推流媒体并停止推流，将用户角色设置回观众角色与普通观众一样，保持订阅拉流即可。
- 对于普通观众，只需保持观看角色，订阅观看推流媒体，无需其他操作。订阅操作可参考官网相关文档说明。