实时音视频

在线直播（含 UI）

腾讯云

文档目录

在线直播（含 UI）
集成 TUILiveRoom (Android)

最近更新时间：2024-05-09 21:48:12

组件介绍

TUILiveRoom 是一个开源的视频直播 UI 组件，通过在项目中集成 TUILiveRoom 组件，您只需要编写几行代码就可以为您的 App 添加“视频互动直播”场景。TUILiveRoom 包含 Android、iOS 等平台的源代码，基本功能如下图所示：

> **说明**
> TUIKit 系列组件同时使用了腾讯云 实时音视频 TRTC 和 即时通信 IM 两个基础 PaaS 服务，开通实时音视频后会同步开通即时通信IM服务。即时通信 IM 服务详细计费规则请参见 即时通信 – 价格说明，TRTC 开通会默认关联开通 IM SDK 的体验版，仅支持100个 DAU。

组件集成

步骤一：下载并导入 TUILiveRoom 组件

单击进入 Github ，选择克隆/下载代码，然后拷贝 Android/debug ， Android/tuiaudioeffect ， Android/tuibarrage ， Android/tuibeauty ， Android/tuigift 和 Android/tuiliveroom 目录到您的工程中，并完成如下导入动作：

- 在 setting.gradle 中完成导入，参考如下：

```
include ':debug'
include ':tuibeauty'
include ':tuibarrage'
include ':tuiaudioeffect'
include ':tuigift'
include ':tuiliveroom'
```

- 在 app 的 build.gradle 文件中添加对 tuiliveroom 的依赖：

```
api project(":tuiliveroom")
```

- 在根目录的 build.gradle 文件中添加 TRTC SDK 和 IM SDK 的依赖：

```
ext {
 liteavSdk = "com.tencent.liteav:LiteAVSDK_TRTC:latest.release"
 imSdk = "com.tencent.imsdk:imsdk-plus:latest.release"
}
```

步骤二：配置权限及混淆规则

1. 在 AndroidManifest.xml 中配置 App 的权限，SDK 需要以下权限（6.0以上的 Android 系统需要动态申请相机、读取存储权限）：

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
/>
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
<uses-permission android:name="android.permission.BLUETOOTH" />
```

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-feature android:name="android.hardware.camera"/>
<uses-feature android:name="android.hardware.camera.autofocus" />
```

2. 在 proguard-rules.pro 文件，将 SDK 相关类加入不混淆名单：

```
-keep class com.tencent.** { *; }
```

步骤三：初始化并登录组件

```
// 1.添加事件监听及登录
TUILogin.addLoginListener(new TUILoginListener() {
    @Override
    public void onConnecting() {      // 正在连接中
        super.onConnecting();
    }
    @Override
    public void onConnectSuccess() {  // 连接成功通知
        super.onConnectSuccess();
    }
    @Override
    public void onConnectFailed(int errorCode, String errorMsg) {  // 连接失败通知
        super.onConnectFailed(errorCode, errorMsg);
    }
    @Override
    public void onKickedOffline() {  // 登录被踢下线通知（示例：账号在其他设备登录）
        super.onKickedOffline();
    }
    @Override
    public void onUserSigExpired() { // userSig过期通知
        super.onUserSigExpired();
    }
});
TUILogin.login(mContext, "Your SDKAppID", "Your userId", "Your userSig", new
TUICallback() {
    @Override
    public void onSuccess() {
    }
    @Override
    public void onError(int errorCode, String errorMsg) {
        Log.d(TAG, "errorCode: " + errorCode + " errorMsg:" + errorMsg);
    }
});
```

```
// 2.初始化TUILiveRoom组件
TUILiveRoom mLiveRoom = TUILiveRoom.sharedInstance(mContext);
```

参数说明：

- **SDKAppID：TRTC 应用 ID**，如果您未开通腾讯云 TRTC 服务，可进入 腾讯云实时音视频控制台，创建一个新的 TRTC 应用后，单击**应用信息**，SDKAppID 信息如下图所示：

- **Secretkey：TRTC 应用密钥**和 SDKAppId 对应，进入 TRTC 应用管理 后，SecretKey 信息如上图所示。
- **userId**：当前用户的 ID，字符串类型，只允许包含英文字母（a-z 和 A-Z）、数字（0-9）、连词符（-）和下划线（_）。建议结合业务实际账号体系自行设置。
- **userSig**：根据 SDKAppId、userId，Secretkey等信息计算得到的安全保护签名，您可以单击 这里 直接在线生成一个调试的userSig，也可以参照我们的 示例工程 自行计算，更多信息请参见 如何计算及使用 UserSig 。

步骤四：实现视频互动直播间

1. **主播端开播**

```
mLiveRoom.createRoom(int roomId, String roomName, String coverUrl);
```

2. **观众端观看**

```
mLiveRoom.enterRoom(roomId);
```

3. 观众与主播连麦 TRTCLiveRoom#requestJoinAnchor

```
// 1.观众端发起连麦请求
// LINK_MIC_TIMEOUT为超时时间
TRTCLiveRoom mTRTCLiveRoom=TRTCLiveRoom.sharedInstance(mContext);
mTRTCLiveRoom.requestJoinAnchor(mSelfUserId + "请求和您连麦",
LINK_MIC_TIMEOUT
 new TRTCLiveRoomCallback.ActionCallback() {
 @Override
 public void onCallback(int code, String msg) {
    if (code == 0) {
        // 主播接受了观众的请求
        TXCloudVideoView view = new TXCloudVideoView(context);
        parentView.add(view);
        // 观众启动预览，开启推流
        mTRTCLiveRoom.startCameraPreview(true, view, null);
        mTRTCLiveRoom.startPublish(mSelfUserId + "_stream", null);
    }
 }
});

// 2.主播端收到连麦请求
mTRTCLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
 @Override
 public void onRequestJoinAnchor(final TRTCLiveRoomDef.TRTCLiveUserInfo userInfo,
    String reason, final int timeout) {
    // 同意对方的连麦请求
    mTRTCLiveRoom.responseJoinAnchor(userInfo.userId, true, "同意连麦");
 }

   @Override
 public void onAnchorEnter(final String userId) {
    // 主播收到连麦观众的上麦通知
    TXCloudVideoView view = new TXCloudVideoView(context);
    parentView.add(view);
    // 主播播放观众画面
    mTRTCLiveRoom.startPlay(userId, view, null);
 }
});
```

4. 主播与主播 PK TRTCLiveRoom#requestRoomPK

```
// 主播 A 创建12345的房间
mLiveRoom.createRoom(12345, "roomA", "Your coverUrl");
// 主播 B 创建54321的房间
mLiveRoom.createRoom(54321, "roomB", "Your coverUrl");

// 主播 A:
TRTCLiveRoom mTRTCLiveRoom=TRTCLiveRoom.sharedInstance(mContext);

// 1.主播 A 向主播 B 发起 PK 请求
mTRTCLiveRoom.requestRoomPK(54321, "B",
 new TRTCLiveRoomCallback.ActionCallback() {
 @Override
 public void onCallback(int code, String msg) {
    // 5.收到是否同意的回调
    if (code == 0) {
       // 用户接受
    } else {
       // 用户拒绝
    }
 }
});

mTRTCLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
 @Override
 public void onAnchorEnter(final String userId) {
    // 6.收到 B 进房的通知
    mTRTCLiveRoom.startPlay(userId, mTXCloudVideoView, null);
 }
});

// 主播 B:
// 2.主播 B 收到主播 A 的消息
mTRTCLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
 @Override
 public void onRequestRoomPK(
   final TRTCLiveRoomDef.TRTCLiveUserInfo userInfo, final int timeout) {
    // 3.主播 B 回复主播 A 接受请求
    mTRTCLiveRoom.responseRoomPK(userInfo.userId, true, "");
 }
 @Override
 public void onAnchorEnter(final String userId) {
    // 4.主播 B 收到主播 A 进房的通知，播放主播 A 的画面
    mTRTCLiveRoom.startPlay(userId, mTXCloudVideoView, null);
 }
});
```

步骤五：美颜特效（可选）

TUILiveRoom 美颜使用了 腾讯特效 SDK，在使用美颜功能时，需要先设置 XMagic License，XMagic License 申请请参见 XMagic License 申请指引 。

```
TUIBeautyView.getBeautyService().setLicense(context, “XMagicLicenseURL”, “XMagicLicenseKey”);
```

交流与反馈

- 如果您是开发者，欢迎您加入我们的技术交流 QQ 群：**770645461**，进行技术交流和产品沟通。

集成 TUILiveRoom (iOS)

最近更新时间：2024-05-09 21:48:12

组件介绍

TUILiveRoom 是一个开源的视频直播 UI 组件，通过在项目中集成 TUILiveRoom 组件，您只需要编写几行代码就可以为您的 App 添加“视频互动直播”场景。TUILiveRoom包含 Android、iOS 等平台的源代码，基本功能如下图所示：

> **说明**
> TUIKit 系列组件同时使用了腾讯云 实时音视频 TRTC 和 即时通信 IM 两个基础 PaaS 服务，开通实时音视频后会同步开通即时通信IM服务。即时通信 IM 服务详细计费规则请参见 即时通信 – 价格说明，TRTC 开通会默认关联开通 IM SDK 的体验版，仅支持100个 DAU。

组件集成

步骤一：导入 TUILiveRoom 组件

通过 cocoapods 导入组件，具体步骤如下：

1. 在您的工程 `Podfile` 文件同一级目录下创建 `TUILiveRoom` 文件夹。
2. 单击进入 Github/TUILiveRoom ，选择克隆/下载代码，然后将 TUILiveRoom/iOS/ 目录下的 `Source` 、`Resources` 、`TUIBeauty` 、`TUIAudioEffect` 、`TUIBarrage` 、`TUIGift` 、`TUIKitCommon` 文件夹 和 `TUILiveRoom.podspec` 文件拷贝到您在 `步骤1` 创建的 TUILiveRoom 文件夹下。
3. 在您的 Podfile 文件中添加以下依赖，之后执行 `pod install` 命令，完成导入。

```
# :path => "指向TUILiveRoom.podspec的相对路径"
pod 'TUILiveRoom', :path => "./TUILiveRoom/TUILiveRoom.podspec", :subspecs =>
["TRTC"]
# :path => "指向TUIKitCommon.podspec的相对路径"
pod 'TUIKitCommon', :path => "./TUILiveRoom/TUIKitCommon/"
# :path => "指向TUIBeauty.podspec的相对路径"
pod 'TUIBeauty', :path => "./TUILiveRoom/TUIBeauty/"
# :path => "指向TUIAudioEffect.podspec的相对路径"
pod 'TUIAudioEffect', :path => "./TUILiveRoom/TUIAudioEffect/", :subspecs =>
["TRTC"]
# :path => "指向TUIBarrage.podspec的相对路径"
pod 'TUIBarrage', :path => "./TUILiveRoom/TUIBarrage/"
# :path => "指向TUIGift.podspec的相对路径"
pod 'TUIGift', :path => "./TUILiveRoom/TUIGift/"
```

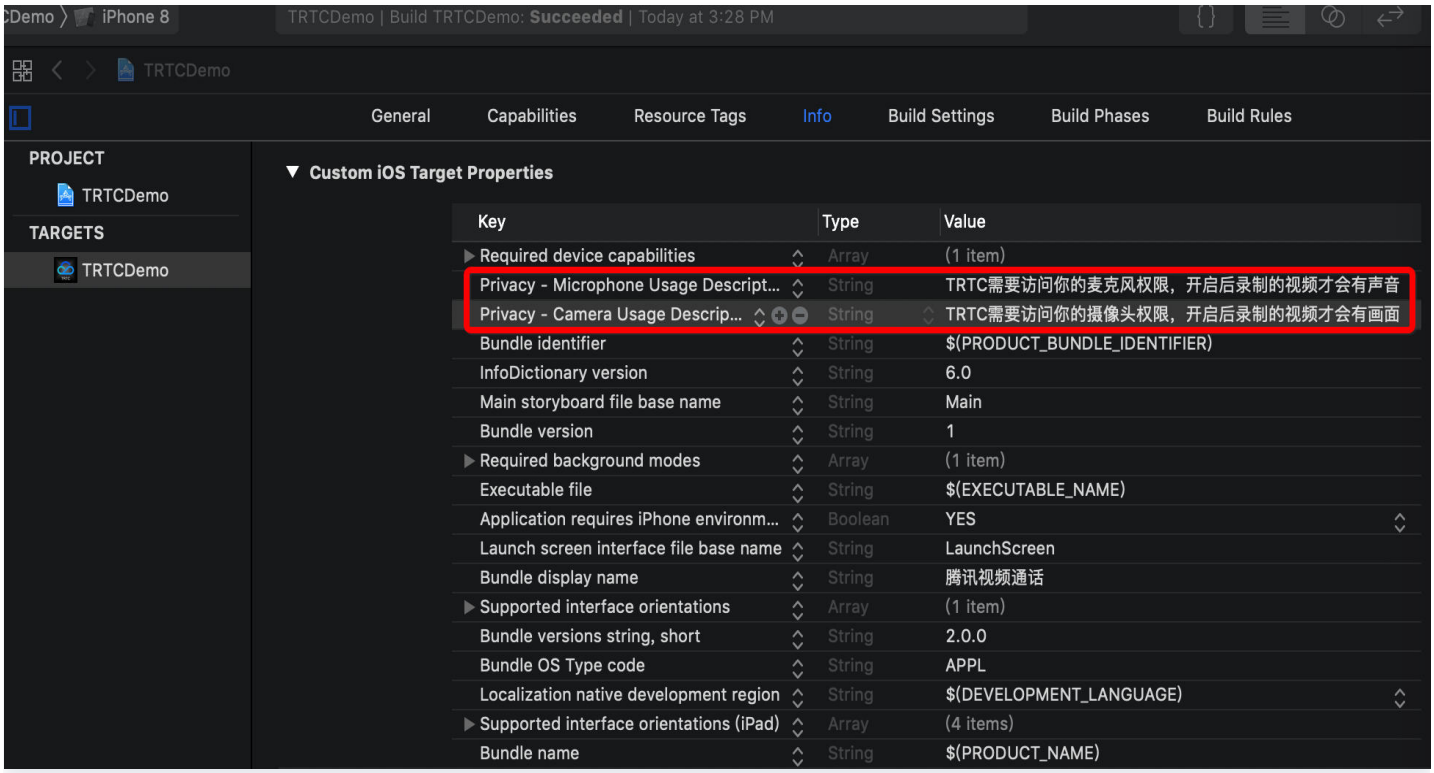
⚠ 注意：

- `Source` 、`Resources` 文件夹和 `TUILiveRoom.podspec` 文件必需在同一目录下。
- TUIKitCommon.podspec 在 TUIKitCommon 文件夹下。

步骤二：配置权限

使用音视频功能，需要授权麦克风和摄像头的使用权限。在 App 的 Info.plist 中添加以下两项，分别对应麦克风和摄像头在系统弹出授权对话框时的提示信息。

```
<key>NSCameraUsageDescription</key>
<string>RoomApp需要访问您的相机权限，开启后录制的视频才会有画面</string>
<key>NSMicrophoneUsageDescription</key>
<string>RoomApp需要访问您的麦克风权限，开启后录制的视频才会有声音</string>
```



步骤三：初始化并登录组件

Objective-C

```
@import TUILiveRoom;
@import TUICore;

// 1.组件登录
[TUILogin login:@"您的SDKAppID" userID:@"您的UserID" userSig:@"您的UserSig"
succ:^{

} fail:^(int code, NSString *msg) {

}];
// 2.初始化TUILiveRoom实例
TUILiveRoom *mLiveRoom = [TUILiveRoom sharedInstance];
```

Swift

```
import TUILiveRoom
import TUICore
```

```
// 1.组件登录
TUILogin.login("您的SDKAppID", userID: "您的UserID", userSig: "您的UserSig") {

} fail: { code, msg in

}
// 2.初始化TUILiveRoom实例
let mLiveRoom = TUILiveRoom.sharedInstance
```

参数说明：

- **SDKAppID：TRTC 应用 ID**，如果您未开通腾讯云 TRTC 服务，可进入 腾讯云实时音视频控制台，创建一个新的 TRTC 应用后，单击**应用信息**，SDKAppID 信息如下图所示：

- **SecretKey：TRTC 应用密钥**，和 SDKAppID 对应，进入 TRTC 应用管理 后，SecretKey 信息如上图所示。
- **UserID**：当前用户的 ID，字符串类型，长度不超过32字节，不支持使用特殊字符，建议使用英文或数字，可结合业务实际账号体系自行设置。
- **UserSig**：根据 SDKAppId、UserID，SecretKey 等信息计算得到的安全保护签名，您可以单击 这里 直接

在线生成一个调试的 UserSig，也可以参照我们的 TUILiveRoom 示例工程 自行计算，更多信息见 如何计算及使用 UserSig 。

步骤四：实现视频互动直播间

1. 主播端开播

Objective-C

```
[mLiveRoom createRoomWithRoomId:123 roomName:@"test room" coverUrl:@""];
```

Swift

```
mLiveRoom.createRoom(roomId: 123, roomName: "test room", coverUrl:"")
```

2. 观众端观看

Objective-C

```
[mLiveRoom enterRoomWithRoomId:123];
```

Swift

```
mLiveRoom.createRoom(roomId: 123)
```

3. 观众与主播连麦 TRTCLiveRoom#requestJoinAnchor

Objective-C

```
// 1.观众端发起连麦请求
[TRTCLiveRoom shareInstance].delegate = self;
// @param mSelfUserId String 当前用户id
```

```
NSString *mSelfUserId = @"1314";
[[TRTCLiveRoom shareInstance] requestJoinAnchor:[NSString stringWithFormat:@"%@
请求和你连麦", mSelfUserId] timeout:30 responseCallback:^(BOOL agreed, NSString *
_Nullable reason) {
    if (agreed) {
        // 主播接受了观众的请求
      UIView *playView = [UIView new];
          [self.view addSubView:playView];
        // 观众启动预览，开启推流
      [[TRTCLiveRoom shareInstance] startCameraPreviewWithFrontCamera:YES
view:playView callback:nil];
      [[TRTCLiveRoom shareInstance] startPublishWithStreamID:[NSString
stringWithFormat:@"%@_stream", mSelfUserId] callback:nil];
    }
}];

// 2.主播端收到连麦请求
#pragma mark - TRTCLiveRoomDelegate
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onRequestJoinAnchor:
(TRTCLiveUserInfo *)user reason:(NSString *)reason {
    // 同意对方的连麦请求
    [[TRTCLiveRoom shareInstance] responseJoinAnchor:user.userId agree:YES
reason:@"同意连麦"];
}

- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onAnchorEnter:(NSString *)userID
{
    // 主播收到连麦观众的上麦通知
    UIView *playView = [UIView new];
    [self.view addSubview:playView];
    // 主播播放观众画面
    [[TRTCLiveRoom shareInstance] startPlayWithUserID:userID view:playView
callback:nil];
}
```

Swift

```
// 1.观众端发起连麦请求
TRTCLiveRoom.shareInstance().delegate = self
let mSelfUserId = "1314"
TRTCLiveRoom.shareInstance().requestJoinAnchor(reason: mSelfUserId + "请求和您连
麦", timeout: 30) {  [weak self] (agree, msg) in
    guard let self = self else { return }
  if agree {
        // 主播接受了观众的请求
```

```
        let playView = UIView()
        self.view.addSubView(playView)
        // 观众启动预览，开启推流
        TRTCLiveRoom.shareInstance().startCameraPreview(frontCamera: true, view:
playView)
        TRTCLiveRoom.shareInstance().startPublish(streamID: mSelfUserId + "_stream")
    }
}

// 2.主播端收到连麦请求
extension ViewController: TRTCLiveRoomDelegate {

    func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onRequestJoinAnchor user:
TRTCLiveUserInfo, reason: String?) {
        // 同意对方的连麦请求
        TRTCLiveRoom.shareInstance().responseRoomPK(userID: user.userId, agree: true,
reason: "同意连麦")
    }

    func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onAnchorEnter userID: String) {
        // 主播收到连麦观众的上麦通知
        let playView = UIView()
        view.addSubview(playView)
        // 主播播放观众画面
        TRTCLiveRoom.shareInstance().startPlay(userID: userID, view: playView);
    }
}
```

4. 主播与主播 PK TRTCLiveRoom#requestRoomPK

Objective-C

```
// 主播 A 创建12345的房间
[[TUILiveRoom sharedInstance] createRoomWithRoomId:12345 roomName:@"roomA"
coverUrl:@"roomA coverUrl"];
// 主播 B 创建54321的房间
[[TUILiveRoom sharedInstance] createRoomWithRoomId:54321 roomName:@"roomB"
coverUrl:@"roomB coverUrl"];

// 主播 A
// 主播 A 向 主播 B 发起PK请求
```

```
[[TRTCLiveRoom shareInstance] requestRoomPKWithRoomID:543321 userID:@"roomB
userId" timeout:30 responseCallback:^(BOOL agreed, NSString * _Nullable reason) {
    if (agreed) {
        // 用户B接受
    } else {
        // 用户B拒绝
    }
}];

// 主播 B：
// 2.主播 B 收到主播 A 的消息
#pragma mark - TRTCLiveRoomDelegate
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onRequestRoomPK:
(TRTCLiveUserInfo *)user {
    // 3.主播 B 回复主播 A 接受请求
    [[TRTCLiveRoom shareInstance] responseRoomPKWithUserID:user.userId agree:YES
reason:@""];
}

- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onAnchorEnter:(NSString *)userID
{
    // 4.主播 B 收到主播 A 进房的通知，播放主播 A 的画面
    [[TRTCLiveRoom shareInstance] startPlayWithUserID:userID view:playAView
callback:nil];
}
```

Swift

```
// 主播 A 创建12345的房间
TUILiveRoom.sharedInstance.createRoom(roomId: 12345, roomName: "roomA")
// 主播 B 创建54321的房间
TUILiveRoom.sharedInstance.createRoom(roomId: 54321, roomName: "roomB")

// 主播 A
// 主播 A 向 主播 B 发起PK请求
TRTCLiveRoom.shareInstance().requestRoomPK(roomID: 543321, userID: "roomB
userId", timeout: 30) { [weak self] (agreed, msg) in
    guard let self = self else { return }
    if agreed {
        // 用户B接受
    } else {
        // 用户B拒绝
    }
}
```

```
// 主播 B：
// 2.主播 B 收到主播 A 的消息
extension ViewController: TRTCLiveRoomDelegate {
    func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onRequestRoomPK user:
TRTCLiveUserInfo) {
        // 3.主播 B 回复主播 A 接受请求
        TRTCLiveRoom.shareInstance().responseRoomPK(userID: user.userId, agree: true,
reason: "")
    }

    func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onAnchorEnter userID: String) {
        // 4.主播 B 收到主播 A 进房的通知，播放主播 A 的画面
        TRTCLiveRoom.shareInstance().startPlay(userID: userID, view: playAView);
    }
}
```

步骤五：美颜特效（可选）

TUILiveRoom 美颜使用了 腾讯特效 SDK，在使用美颜功能时，需要先设置 XMagic License，XMagic License 申请请参见 XMagic License 申请指引 。

Objective-C

```
@import TUIBeauty;

- (void)setXMagicLicense {
    [[TUIBeautyView getBeautyService] setLicenseUrl:@"XMagicLicenseURL"
key:@"XMagicLicenseKey" completion:^(NSInteger authResult, NSString * _Nonnull
errorMsg) {

    }];
}
```

Swift

```
import TUIBeauty

func setXMagicLicence() {
    // [Option] Tencent Effect: XMagic Beauty License
```

```
    TUIBeautyView.getBeautyService().setLicenseUrl(XMagicLicenseURL, key:
XMagicLicenseKey) { code, msg in
        debugPrint("auth result code:\(code) msg:\(msg)")
    }
}
```

交流与反馈

如果您是开发者，欢迎您加入我们的技术交流 QQ 群：**770645461**，进行技术交流和产品沟通。

集成 TUIPusher&TUIPlayer (Web)

最近更新时间：2024-05-09 09:19:31

组件介绍

TUIPusher & TUIPlayer 是 Web 端开源的含 UI 直播互动组件。TUIPusher & TUIPlayer 集成 实时音视频 TRTC 、 即时通信 IM 等基础 SDK，为企业直播、电商带货、行业培训、远程教学等多种直播场景提供快速上线 Web 端直播推拉流工具的解决方案。

> **说明**
> TUIKit 系列组件同时使用了腾讯云 实时音视频 TRTC 和 即时通信 IM 两个基础 PaaS 服务，开通实时音视频后会同步开通即时通信IM服务。即时通信 IM 服务详细计费规则请参见 即时通信 – 价格说明，TRTC 开通会默认关联开通 IM SDK 的体验版，仅支持100个 DAU。

> **注意：**
> Web 端 TUIPusher&TUIPlayer 与其他各端 TUILiveRoom 数据不互通，仅作为 Web 端直播场景应用示例。

TUIPusher & TUIPlayer 的优势：

- 贴合直播场景需求，提供了含 UI 的直播场景通用解决方案，覆盖了直播场景常见功能（如设备选择、美颜、直播推流、观众拉流、聊天等），助力业务快速上线。
- 直接接入腾讯云实时音视频 TRTC、腾讯云即时通信 IM 以及腾讯云超级播放器 TCPlayer 等基础 SDK，方便客户灵活扩展业务功能。
- Web 端易于用户使用，易于功能迭代的天然优势。

快速体验

为了方便您快速体验 TUIPusher & TUIPlayer 的功能，我们结合用户管理系统和房间管理系统提供了 TUIPusher 体验链接 及 TUIPlayer 体验链接 。

> **注意**
> 同时体验 TUIPusher 和 TUIPlayer 需要使用两个不同的账号登录。

TUIPusher 推流组件功能介绍

- 支持采集摄像头和麦克风的流并推流
  - 可根据需求设置视频参数（帧率，分辨率，码率）
  - 支持开启美颜并设置视频美颜参数
- 支持采集屏幕分享流并推流
- 支持推流到腾讯云实时音视频后台，推流到腾讯云 CDN
- 支持在线聊天室，和在线观众进行聊天互动
- 支持获取观众列表，对在线观众进行禁言操作

TUIPlayer 拉流组件功能介绍

- 支持同时播放音视频流和屏幕分享流
- 支持在线聊天室，和主播及其他观众进行聊天互动

- 支持超低延时直播（300ms 延时），快直播（1000ms 以内延时）以及标准直播（支持超高并发观看）三种拉流线路
- 兼容桌面浏览器及移动端浏览器，支持移动端浏览器横屏观看

> **注意**
> 部分浏览器不支持 WebRTC，只能使用标准直播线路观看，如需体验其他线路，请尝试更换浏览器。

组件集成

步骤一：开通腾讯云服务

> **注意**
> - TUIPusher & TUIPlayer 基于腾讯云实时音视频和即时通信服务进行开发。实时音视频 TRTC 应用与 即时通信 IM 应用的 SDKAppID 一致，才能复用账号与鉴权。
> - 即时通信 IM 应用针对文本消息，提供基础版本的 安全打击 能力，如果希望使用自定义不雅词功能，可以单击 **升级** 或在 购买页 购买 **安全打击-高级版** 服务。
> - 本地计算 UserSig 的方式仅用于本地开发调试，请勿直接发布到线上，一旦 SECRETKEY 泄露，攻击者就可以盗用您的腾讯云流量。正确的 UserSig 签发方式是将 UserSig 的计算代码集成到您的服务端，并提供面向 App 的接口，在需要 UserSig 时由您的 App 向业务服务器发起请求获取动态 UserSig。更多详情请参见 服务端生成 UserSig 。

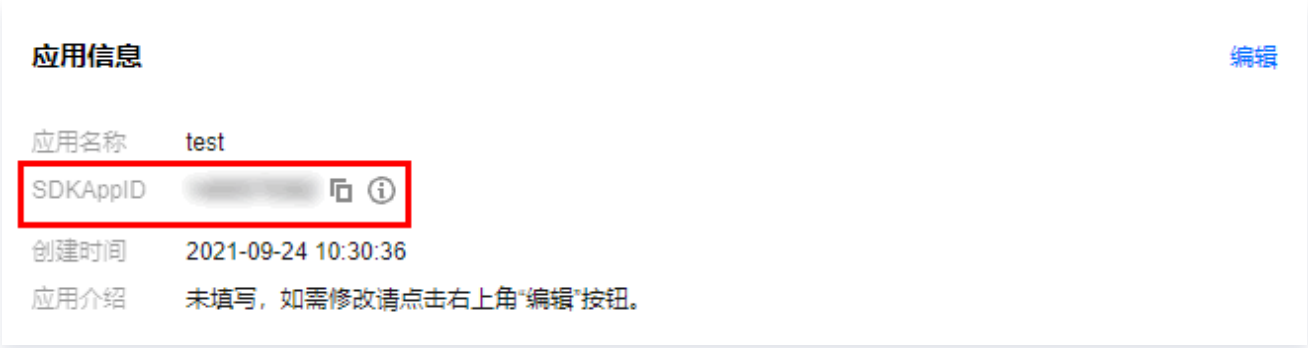
方式1：基于实时音视频

步骤1：创建实时音视频 TRTC 应用

1. 注册腾讯云账号 并开通 实时音视频 和 即时通信 服务。
2. 在 实时音视频控制台 单击 **应用管理 > 创建应用** 创建新应用。

步骤2: 获取 TRTC 密钥信息

1. 在 **应用管理 > 应用信息** 中获取 SDKAppID 信息。



2. 在 **应用管理 > 快速上手** 中获取应用的 secretKey 信息。

> **说明**

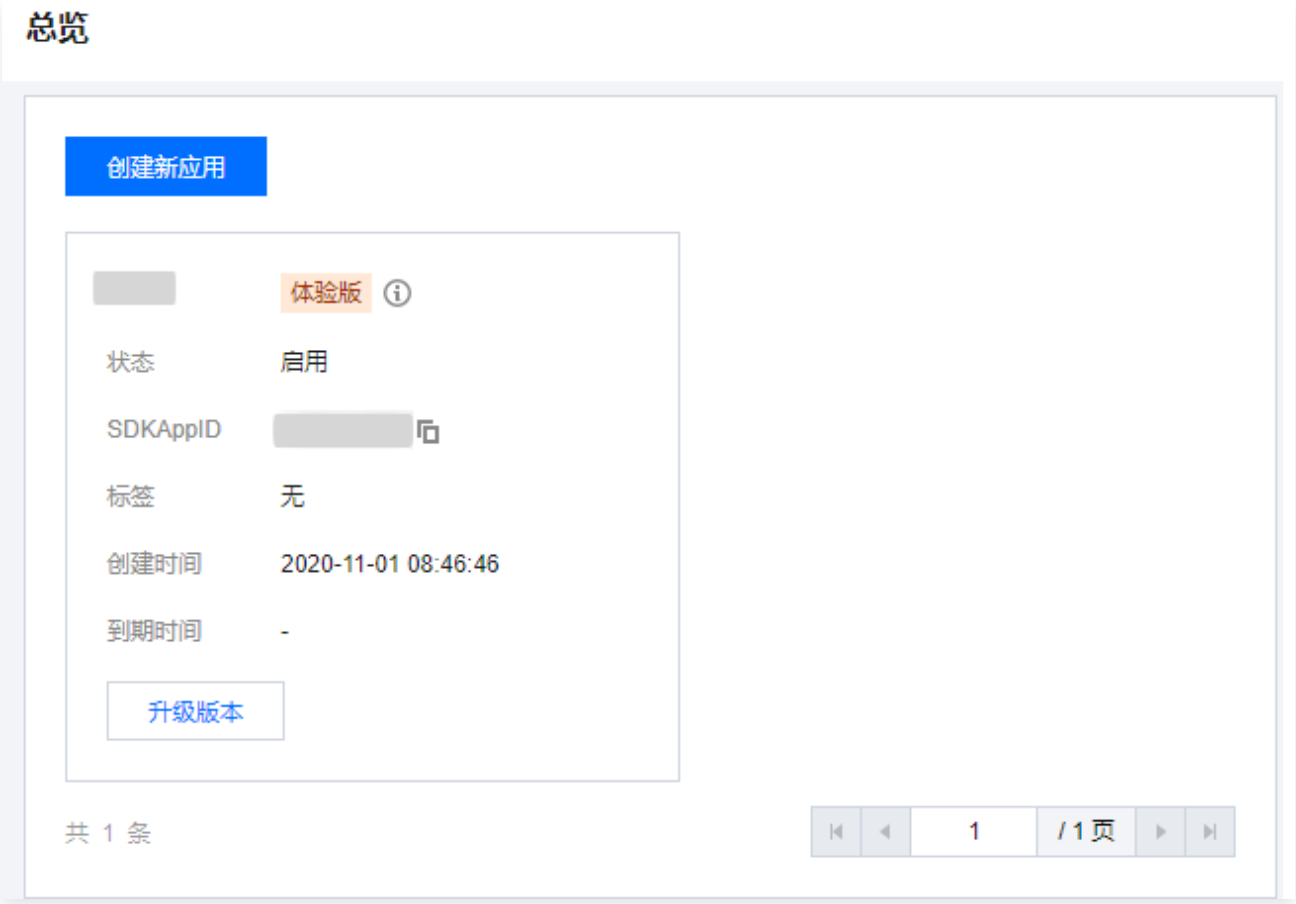
- 首次创建实时音视频应用的腾讯云账号，可获赠一个10000分钟的音视频资源免费试用包。
- 创建实时音视频应用之后会自动创建一个 SDKAppID 相同的即时通信 IM 应用，可在 即时通信控制台 配置该应用的套餐信息。

方式2：基于即时通信 IM

步骤1：创建即时通信 IM 应用

1. 登录 即时通信 IM 控制台，单击 **创建新应用** 将弹出对话框。

2. 输入您的应用名称，单击 **确认** 即可完成创建。



3. 您可在 即时通信 IM 控制台 总览页面查看新建应用的状态、业务版本、SDKAppID、创建时间以及到期时间。请记录 SDKAppID 信息。

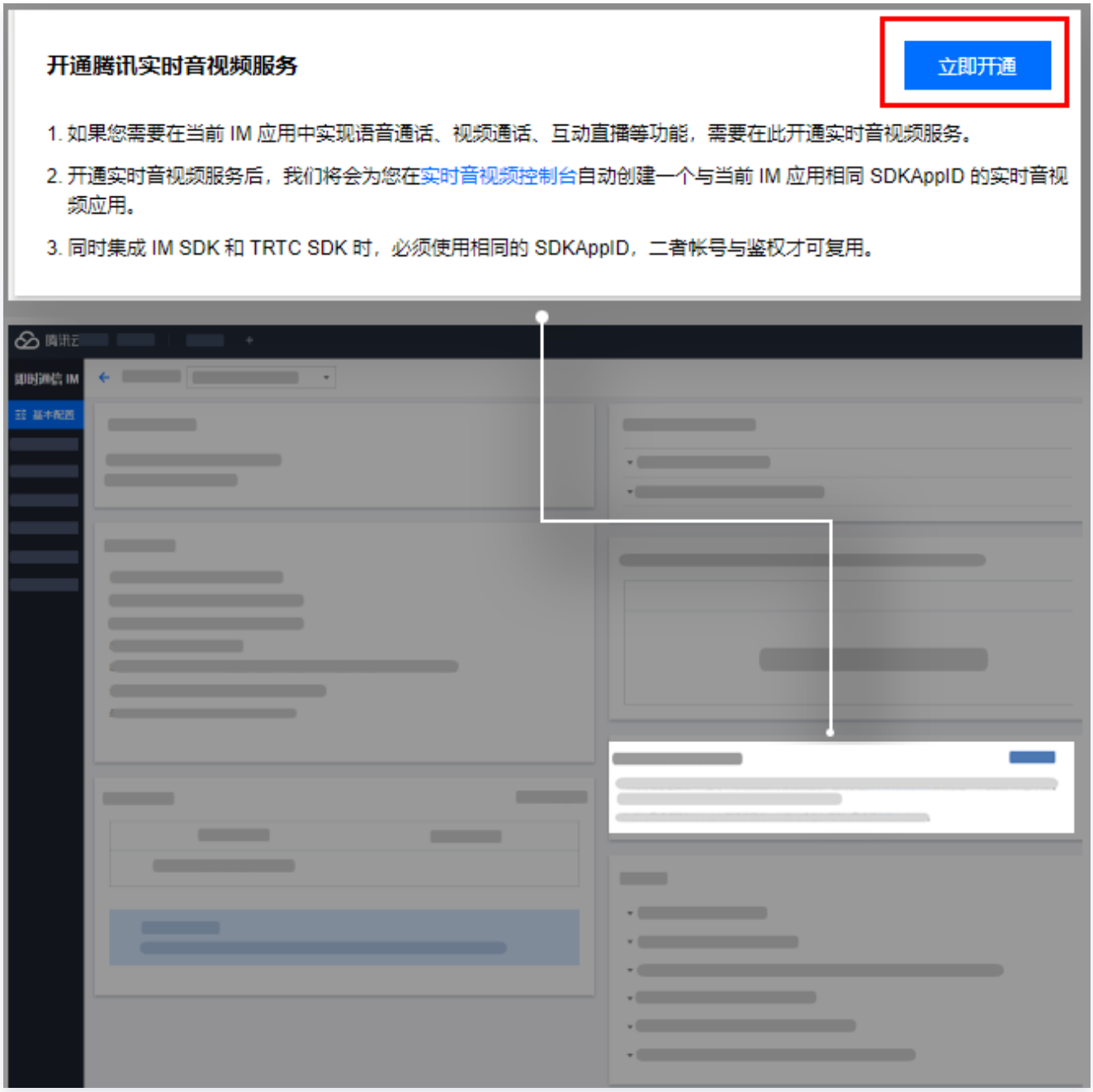
步骤2：获取 IM 密钥并开通实时音视频服务

1. 在 即时通信 IM 控制台 总览页单击您创建完成的即时通信 IM 应用，随即跳转至该应用的基础配置页。在**基本信息** 区域，单击 **显示密钥**，复制并保存密钥信息。

注意

请妥善保管密钥信息，谨防泄露。

2. 在该应用的基础配置页，开通腾讯云实时音视频服务。



步骤二：项目准备

开发环境要求：

- Vue 2
- node（v14.16.0）
- npm（版本请与 node 版本匹配）

1. 在 GitHub 下载 TUIPusher & TUIPlayer 代码。
2. 为 TUIPusher & TUIPlayer 安装依赖。

```
cd Web/TUIPusher
npm install

cd Web/TUIPlayer
npm install
```

3. 将 sdkAppId 和 secretKey 填入 TUIPusher/src/config/basic-info-config.js 及 TUIPlayer/src/config/basic-info-config.js 配置文件中。

```
/**
 * 腾讯云 SDKAppId，需要替换为您自己账号下的 SDKAppId。
 *
 * 进入腾讯云实时音视频[控制台](https://console.cloud.tencent.com/rav ) 创建应用，即可看到 SDKAppId，
 * 它是腾讯云用于区分客户的唯一标识。
 */
export const sdkAppId = 0;
/**
 * 签名过期时间，建议不要设置的过短
 * <p>
 * 时间单位：秒
 * 默认时间：7 x 24 x 60 x 60 = 604800 = 7 天
 */
export const expireTime = 604800;

/**
 * 计算签名用的加密密钥，获取步骤如下：
 *
 * step1. 进入腾讯云实时音视频[控制台](https://console.cloud.tencent.com/rav )，如果还没有应用就创建一
 * step2. 单击“应用配置”进入基础配置页面，并进一步找到“帐号体系集成”部分。
 * step3. 点击“查看密钥”按钮，就可以看到计算 UserSig 使用的加密的密钥了，请将其拷贝并复制到如下的变量中
 *
 * 注意：该方案仅适用于调试Demo，正式上线前请将 UserSig 计算代码和密钥迁移到您的后台服务器上，以避免加密密钥泄
 * 文档：https://cloud.tencent.com/document/product/647/17275#Server
 */
export const secretKey = '';
```

4. 本地开发环境运行 TUIPusher & TUIPlayer。

```
cd Web/TUIPusher
npm run serve

cd Web/TUIPlayer
npm run serve
```

5. 可打开 http://localhost:8080 和 http://localhost:8081 体验 TUIPusher 和 TUIPlayer 功能。
6. 可更改 TUIPusher/src/config/basic-info-config.js 及 TUIPlayer/src/config/basic-info-config.js 配置文件中的房间，主播及观众等信息，**注意保持 TUIPusher 和 TUIPlayer 的房间信息，主播信息一致**。

> **注意**
> - 完成以上配置，您可以使用 TUIPusher & TUIPlayer 进行超低延时直播，如您需要支持快直播和标准直播，请继续阅读 步骤三：旁路直播 。
> - 本地计算 UserSig 的方式仅用于本地开发调试，请勿直接发布到线上，一旦您的 SECRETKEY 泄露，攻击者就可以盗用您的腾讯云流量。
> - 正确的 UserSig 签发方式是将 UserSig 的计算代码集成到您的服务端，并提供面向 App 的接口，在需要 UserSig 时由您的 App 向业务服务器发起请求获取动态 UserSig。更多详情请参见 服务端生成 UserSig 。

步骤三：旁路直播

TUIPusher & TUIPlayer 实现的快直播和标准直播依托于腾讯云 云直播服务 ，因此支持快直播和标准直播线路需要您开启旁路推流功能。

1. 在 实时音视频控制台 中为您正在使用的应用开启旁路推流配置，可按需开启指定流旁路或全局自动旁路。

2. 请在 域名管理 页面添加自有播放域名，具体请参见 添加自有域名 。
3. 在 TUIPlayer/src/config/basic-info-config.js 配置文件中配置播放域名。

完成以上配置，您可以体验 TUIPusher & TUIPlayer 支持超低延时直播，快直播以及标准直播的所有功能。

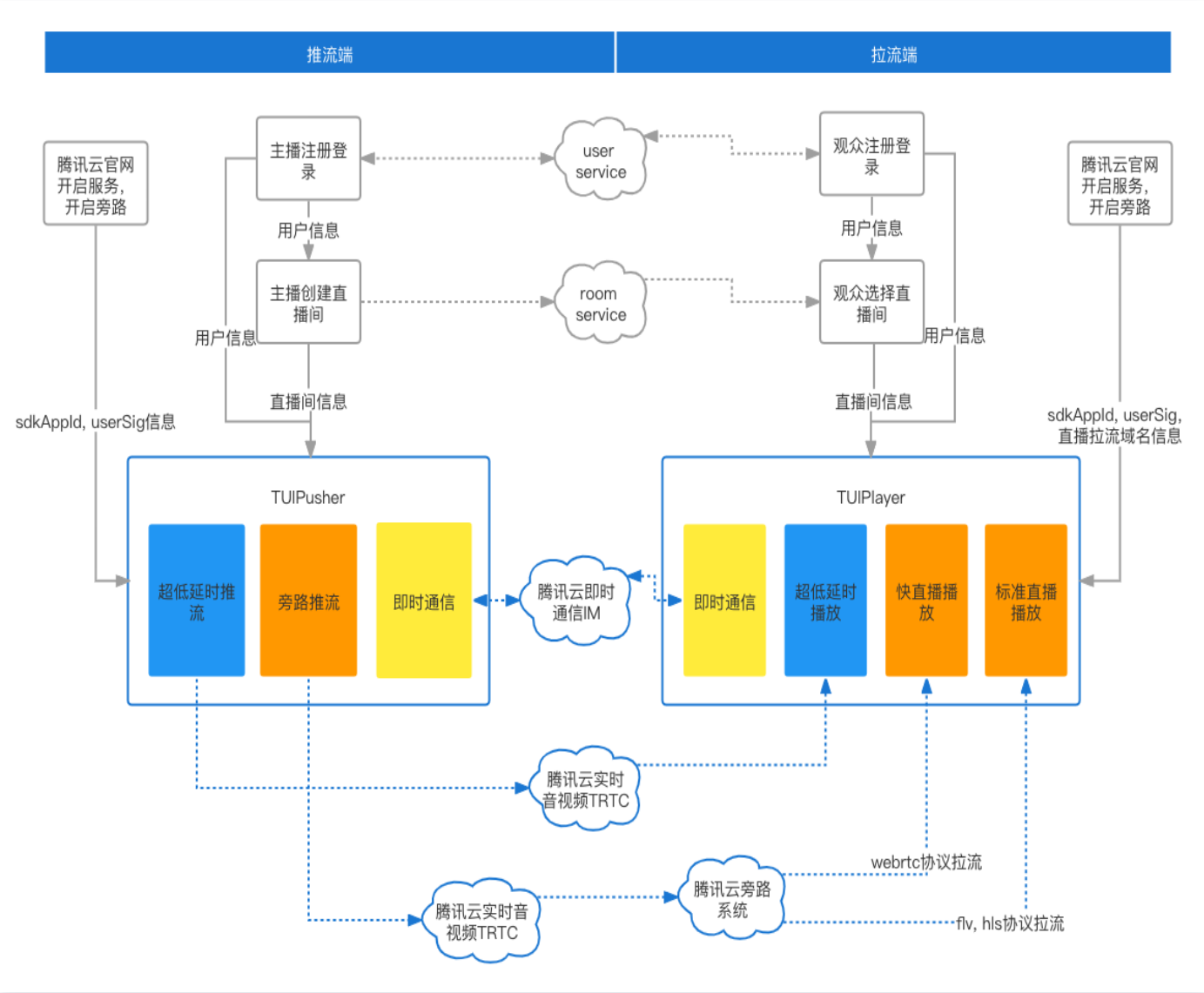
步骤四：生产环境应用

当您将 TUIPusher & TUIPlayer 用于生产应用时，在接入 TUIPusher & TUIPlayer 之外，您需要：

- 创建用户管理系统，用于管理产品用户信息，包括但不限于用户 ID，用户名，用户头像等。
- 创建房间管理系统，用于管理产品直播间信息，包括但不限于直播间 ID、直播间名称，直播间主播信息等。
- 服务端生成 UserSig。

> **注意**
> - 本文生成 UserSig 的方式，是在客户端根据您填入的 sdkAppId 及 secretKey 生成 userSig，该方式的 secretKey 很容易被反编译逆向破解，一旦您的密钥泄露，攻击者就可以盗用您的腾讯云流量，因此**该方法仅适合本地跑通 TUIPusher & TUIPlayer 进行功能调试**。
> - 正确的 UserSig 签发方式是将 UserSig 的计算代码集成到您的服务端，并提供面向 App 的接口，在需要 UserSig 时由您的应用向业务服务器发起请求获取动态 UserSig。更多详情请参见 服务端生成 UserSig 。

- 参考 `TUIPusher/src/pusher.vue` 及 `TUIPlayer/src/player.vue` 文件，将用户信息、直播间信息、SDKAppId 及 UserSig 等账号信息提交到 vuex 的 store 进行全局存储，您就可以跑通推拉流两个客户端的所有功能。详细业务流程参见下图：



相关问题

Web 端如何实现美颜功能?

请参见 开启美颜 。

Web 端如何实现屏幕共享?

请参见 屏幕分享 。

Web 端如何实现云端录制?

1. 开启**云端录制**功能，具体操作请参见 实现云端录制与回放 。

2. 开启**云端录制> 指定用户录制**之后，Web 端可通过在调用 TRTC.createClient 接口时传入 userDefineRecordId 参数开启录制。

Web 端如何实现推流到 CDN？

Web 端推流到 CDN 请参见 实现推流到 CDN 。

Web 端如何实现快直播拉流?

实现快直播拉流的方式是通过 Web SDK 推流到 CDN 之后使用 WebRTC 协议拉流，具体请参见 快直播拉流 > Web（H5）播放器 。

Web 端如何实现标准直播拉流?

实现标准直播拉流，请参见 标准直播拉流 > Web（H5）播放器 。

注意事项

平台支持

操作系统	浏览器类型	浏览器最低版本要求	TUIPlayer	TUIPusher	TUIPusher 屏幕分享
Mac OS	桌面版 Safari 浏览器	11+	支持	支持	支持（需要 Safari13+ 版本）
Mac OS	桌面版 Chrome 浏览器	56+	支持	支持	支持（需要 Chrome72+ 版本）
Mac OS	桌面版 Firefox 浏览器	56+	支持	支持	支持（需要 Firefox66+ 版本）
Mac OS	桌面版 Edge 浏览器	80+	支持	支持	支持
Mac OS	桌面版微信内嵌网页	-	支持	不支持	不支持
Mac OS	桌面版企业微信内嵌网页	-	支持	不支持	不支持
Windows	桌面版 Chrome 浏览器	56+	支持	支持	支持（需要 Chrome72+ 版本）
Windows	桌面版 QQ 浏览器（极速内核）	10.4+	支持	支持	不支持

Windows	桌面版 Firefox 浏览器	56+	支持	支持	支持（需要 Firefox66+ 版本）
Windows	桌面版 Edge 浏览器	80+	支持	支持	支持
Windows	桌面版微信内嵌网页	-	支持	不支持	不支持
Windows	桌面版企业微信内嵌网页	-	支持	不支持	不支持
iOS	微信内嵌浏览器	-	支持	不支持	不支持
iOS	企业微信内嵌浏览器	-	支持	不支持	不支持
iOS	移动版 Safari 浏览器	-	支持	不支持	不支持
iOS	移动版 Chrome 浏览器	-	支持	不支持	不支持
Android	微信内嵌浏览器	-	支持	不支持	不支持
Android	企业微信浏览器	-	支持	不支持	不支持
Android	移动版 Chrome 浏览器	-	支持	不支持	不支持
Android	移动版 QQ 浏览器	-	支持	不支持	不支持
Android	移动版 Firefox 浏览器	-	支持	不支持	不支持
Android	移动端 UC 浏览器	-	支持（仅支持标准直播观看）	不支持	不支持

域名要求

出于对用户安全、隐私等问题的考虑，浏览器限制网页在 HTTPS 协议下才能正常使用 TUIPusher & TUIPlayer 的全部功能。为确保生产环境用户顺畅接入和体验 TUIPusher & TUIPlayer 的全部功能，请使用 HTTPS 协议访问音视频应用页面。

> **注意**
> 本地开发可以通过 `http://localhost` 协议进行访问。

URL 域名及协议支持情况请参考如下表格：

应用场景	协议	TUIPlayer	TUIPusher	TUIPusher 屏幕分享	备注
生产环境	HTTPS 协议	支持	支持	支持	推荐
生产环境	HTTP 协议	支持	不支持	不支持	-
本地开发环境	`http://localhost`	支持	支持	支持	推荐
本地开发环境	`http://127.0.0.1`	支持	支持	支持	-
本地开发环境	`http://[本机IP]`	支持	不支持	不支持	-

防火墙限制

TUIPusher & TUIPlayer 依赖以下端口进行数据传输，请将其加入防火墙白名单。

- TCP 端口：8687
- UDP 端口：8000，8080，8800，843，443，16285
- 域名：qcloud.rtc.qq.com

结语

在后续的迭代中, TRTC Web 端推拉流组件会逐渐与 iOS、Andriod 等各端连通，并在 Web 端实现观众连麦、高级美颜、自定义布局、转推多平台、上传图片文字音乐等能力，欢迎大家多多使用、提出您的宝贵意见。

如果有任何需要或者反馈，可扫描下方二维码，或者单击 反馈链接 同步给我们。

此外，我们欢迎加入 TUIRoomKit 技术交流 QQ 群（群号：770645461）进行技术交流和问题反馈。

TUILiveRoom API 查询
TRTCLiveRoom API (iOS)

最近更新时间：2023-08-24 16:22:41

TRTCLiveRoom 是基于腾讯云实时音视频（TRTC）和即时通信 IM 服务组合而成的，支持以下功能：

- 主播创建新的直播间开播，观众进入直播间观看。
- 主播和观众进行视频连麦互动。
- 两个不同房间的主播 PK 互动。
- 支持发送各种文本消息和自定义消息，自定义消息可用于实现弹幕、点赞和礼物。

> **说明**
> TUIKit 系列组件同时使用了腾讯云 实时音视频 TRTC 和 即时通信 IM 两个基础 PaaS 服务，开通实时音视频后会同步开通即时通信IM服务。即时通信 IM 服务详细计费规则请参见 即时通信 – 价格说明，TRTC 开通会默认关联开通 IM SDK 的体验版，仅支持100个 DAU。

TRTCLiveRoom 是一个开源的 Class，依赖腾讯云的两个闭源 SDK，具体的实现过程请参见 视频连麦直播（iOS）。

- TRTC SDK：使用 TRTC SDK 作为低延时直播组件。
- IM SDK：使用 IM SDK 的 AVChatroom 实现直播聊天室的功能，同时，通过 IM 消息串联主播间的连麦流程。

TRTCLiveRoom API 概览

SDK 基础函数

API	描述
delegate	设置事件回调。
login	登录。
logout	登出。
setSelfProfile	修改个人信息。

房间相关接口函数

API	描述
createRoom	创建房间（主播调用），若房间不存在，系统将自动创建一个新房间。

destroyRoom	销毁房间（主播调用）。
enterRoom	进入房间（观众调用）。
exitRoom	离开房间（观众调用）。
getRoomInfos	获取房间列表的详细信息。
getAnchorList	获取房间内所有的主播列表，`enterRoom()` 成功后调用才有效。
getAudienceList	获取房间内所有的观众信息，`enterRoom()` 成功后调用才有效。

推拉流相关接口函数

API	描述
startCameraPreview	开启本地视频的预览画面。
stopCameraPreview	停止本地视频采集及预览。
startPublish	开始直播（推流）。
stopPublish	停止直播（推流）。
startPlay	播放远端视频画面，可以在普通观看和连麦场景中调用。
stopPlay	停止渲染远端视频画面。

主播和观众连麦

API	描述
requestJoinAnchor	观众请求连麦。
responseJoinAnchor	主播处理连麦请求。
kickoutJoinAnchor	主播移除连麦观众。

主播跨房间 PK

API	描述

requestRoomPK	主播请求跨房 PK。
responseRoomPK	主播响应跨房 PK 请求。
quitRoomPK	退出跨房 PK。

音视频控制相关接口函数

API	描述
switchCamera	切换前后摄像头。
setMirror	设置是否镜像展示。
muteLocalAudio	静音本地音频。
muteRemoteAudio	静音远端音频。
muteAllRemoteAudio	静音所有远端音频。

背景音乐音效相关接口函数

API	描述
getAudioEffectManager	获取背景音乐音效管理对象 TXAudioEffectManager 。

美颜滤镜相关接口函数

API	描述
getBeautyManager	获取美颜管理对象 TXBeautyManager 。

消息发送相关接口函数

API	描述
sendRoomTextMsg	在房间中广播文本消息，一般用于弹幕聊天。
sendRoomCustomMsg	发送自定义文本消息。

调试相关接口函数

API	描述
showVideoDebugLog	是否在界面中展示 Debug 信息。

TRTCLiveRoomDelegate API 概览

通用事件回调

API	描述
onError	错误回调。
onWarning	警告回调。
onDebugLog	Log 回调。

房间事件回调

API	描述
onRoomDestroy	房间被销毁的回调。
onRoomInfoChange	直播房间信息变更回调。

主播和观众进出事件回调

API	描述
onAnchorEnter	收到新主播进房通知。
onAnchorExit	收到主播退房通知。
onAudienceEnter	收到观众进房通知。
onAudienceExit	收到观众退房通知。

主播和观众连麦事件回调

API	描述
onRequestJoinAnchor	主播收到观众连麦请求时的回调。
onKickoutJoinAnchor	连麦观众收到被移出连麦的通知。

主播 PK 事件回调

API	描述
onRequestRoomPK	收到请求跨房 PK 通知。
onQuitRoomPK	收到断开跨房 PK 通知。

消息事件回调

API	描述
onRecvRoomTextMsg	收到文本消息。
onRecvRoomCustomMsg	收到自定义消息。

SDK 基础函数

delegate

TRTCLiveRoom 事件回调，您可以通过 TRTCLiveRoomDelegate 获得 TRTCLiveRoom 的各种状态通知。

```
@property(nonatomic, weak)id<TRTCLiveRoomDelegate> delegate;
```

> **说明**
> delegate 是 TRTCLiveRoom 的代理回调。

login

登录。

```
/// 登录到组件系统
/// - Parameters:
///   - sdkAppID: 您可以在实时音视频控制台 > 【[应用管理](https://console.cloud.tencent.com/trtc/app)】> 应用信息中查看 SDKAppID。
///   - userID: 当前用户的 ID，字符串类型，只允许包含英文字母（a-z 和 A-Z）、数字（0-9）、连词符（-）和下划线（\_）。
///   - userSig: 腾讯云设计的一种安全保护签名，获取方式请参见 [如何计算及使用 UserSig](https://cloud.tencent.com/document/product/647/17275)。
```

```
///   - config: 全局配置信息，请在登录时初始化，登录之后不可变更。 isAttachedTUIKit 项目中是否引入并使用TUIKit
///   - callback: 登录回调，成功时 code 为0。
/// - Note:
///   - userSig 建议设定 7 天，能够有效规避 usersign 过期导致的 IM 收发消息失败、TRTC 连麦失败等情况
- (void)loginWithSdkAppID:(int)sdkAppID
                   userID:(NSString *)userID
                  userSig:(NSString *)userSig
                   config:(TRTCLiveRoomConfig *)config
                 callback:(Callback _Nullable)callback
NS_SWIFT_NAME(login(sdkAppID:userID:userSig:config:callback:));
```

参数如下表所示：

参数	类型	含义
sdkAppID	Int	您可以在实时音视频控制台 >应用管理> 应用信息中查看 SDKAppID。
userID	String	当前用户的 ID，字符串类型，只允许包含英文字母（a-z 和 A-Z）、数字（0-9）、连词符（-）和下划线（_）。
userSig	String	腾讯云设计的一种安全保护签名，获取方式请参见 如何计算及使用 UserSig。
config	TRTCLiveRoomConfig	全局配置信息，请在登录时初始化，登录之后不可变更。 • useCDNFirst 属性：用于设置观众观看方式。true 表示普通观众通过 CDN 观看，计费便宜但延时较高。false 表示普通观众通过低延时观看，计费价格介于 CDN 和连麦之间，但延迟可控制在1s以内。 • CDNPlayDomain 属性：在 useCDNFirst 设置为 true 时才会生效，用于指定 CDN 观看的播放域名，您可以登录直播控制台 > 域名管理页面中进行设置。
callback	(_ code: Int, _ message: String?) -> Void	登录回调，成功时 code 为0。

logout

登出。

```
// 退出登录
/// - Parameter callback:  登出回调，成功时 code 为0
- (void)logout:(Callback _Nullable)callback
NS_SWIFT_NAME(logout(_:));
```

参数如下表所示：

参数	类型	含义
callback	(_ code: Int, _ message: String?) -> Void	登出回调，成功时 code 为0。

setSelfProfile

修改个人信息。

```
/// 设置用户信息，您设置的用户信息会被存储于腾讯云 IM 云服务中。
/// - Parameters:
///   - name: 用户昵称
///   - avatarURL: 用户头像地址
///   - callback: 个人信息设置回调，成功时 code 为0
- (void)setSelfProfileWithName:(NSString *)name
                     avatarURL:(NSString * _Nullable)avatarURL
                      callback:(Callback _Nullable)callback
NS_SWIFT_NAME(setSelfProfile(name:avatarURL:callback:));
```

参数如下表所示：

参数	类型	含义
name	String	昵称。
avatarURL	String	头像地址。
callback	(_ code: Int, _ message: String?) -> Void	个人信息设置回调，成功时 code 为0。

房间相关接口函数

createRoom

创建房间（主播调用）。

```
/// 创建房间（主播调用），若房间不存在，系统将自动创建一个新房间。
/// 主播开播的正常调用流程是：
/// 1.【主播】调用 startCameraPreview() 打开摄像头预览，此时可以调整美颜参数。
/// 2.【主播】调用 createRoom() 创建直播间，房间创建成功与否会通过 callback 通知给主播。
/// 3.【主播】调用 startPublish() 开始推流。
/// - Parameters:
```

```
///   - roomID: 房间标识，需要由您分配并进行统一管理。多个 roomid 可以汇总成一个直播间列表，腾讯云暂不提供直播间列表的管理服务，请自行管理您的直播间列表。
///   - roomParam: TRTCCreateRoomParam | 房间信息，用于房间描述的信息，例如房间名称，封面信息等。如果房间列表和房间信息都由您自行管理，可忽略该参数。
///   - callback:  进入房间的结果回调，成功时 code 为0。
/// - Note:
///   - 主播开始直播的时候调用，可重复创建自己已创建过的房间。
- (void)createRoomWithRoomID:(UInt32)roomID
                  roomParam:(TRTCCreateRoomParam *)roomParam
                   callback:(Callback _Nullable)callback
NS_SWIFT_NAME(createRoom(roomID:roomParam:callback:));
```

参数如下表所示：

参数	类型	含义
roomID	UInt32	房间标识，需要由您分配并进行统一管理。多个 roomID 可以汇总成一个直播间列表，腾讯云暂不提供直播间列表的管理服务，请自行管理您的直播间列表。
roomParam	TRTCCreateRoomParam	房间信息，用于房间描述的信息，例如房间名称，封面信息等。如果房间列表和房间信息都由您自行管理，可忽略该参数。
callback	(_ code: Int, _ message: String?) -> Void	创建房间的结果回调，成功时 code 为0。

主播开播的正常调用流程如下：

1. **主播调用** `startCameraPreview()` **打开摄像头预览，此时可以调整美颜参数。**
2. **主播调用** `createRoom()` **创建直播间，房间创建成功与否会通过 callback 通知给主播。**
3. **主播调用** `starPublish()` **开始推流。**

destroyRoom

销毁房间（主播调用）。主播在创建房间后，可以调用该函数来销毁房间。

```
/// 销毁房间（主播调用）
/// 主播在创建房间后，可以调用这个函数来销毁房间。
/// - Parameter callback: 销毁房间的结果回调，成功时 code 为0。
/// - Note:
///   - 主播在创建房间后，可以调用该函数来销毁房间。
- (void)destroyRoom:(Callback _Nullable)callback
NS_SWIFT_NAME(destroyRoom(callback:));
```

参数如下表所示：

参数	类型	含义
callback	(_ code: Int, _ message: String?) -> Void	销毁房间的结果回调，成功时 code 为0。

enterRoom

进入房间（观众调用）。

```
/// 进入房间（观众调用）
/// 观众观看直播的正常调用流程是：
/// 1.【观众】向您的服务端获取最新的直播间列表，其中有多个直播间的 roomid 和房间信息。
/// 2.【观众】观众选择一个直播间以后，调用 enterRoom() 进入该房间。
/// 3.【观众】如果您的服务器所管理的房间列表中包含每一个房间的主播 userID，则可以直接在 enterRoom() 成功后调用 startPlay(userID) 即可播放主播的画面。
/// 如果您管理的房间列表只有 roomid 也没有关系，观众在 enterRoom() 成功后很快会收到来自 TRTCLiveRoomDelegate 中的 onAnchorEnter(userID) 回调。
/// 此时使用回调中的 userID 调用 startPlay(userID) 即可播放主播的画面。
/// - Parameters:
///   - roomID: 房间标识。
///   - useCDNFirst: 是否优先使用CDN播放
///   - cdnDomain: CDN域名
///   - callback: 进入房间的结果回调，成功时 code 为0。
/// - Note:
///   - 观众进入直播房间的时候调用
///   - 主播不可调用这个接口进入自己已创建的房间，而要用createRoom
- (void)enterRoomWithRoomID:(UInt32)roomID
                useCDNFirst:(BOOL)useCDNFirst
                  cdnDomain:(NSString * _Nullable)cdnDomain
                   callback:(Callback)callback
NS_SWIFT_NAME(enterRoom(roomID:useCDNFirst:cdnDomain:callback:));
```

参数如下表所示：

参数	类型	含义
roomID	UInt32	房间标识。
callback	(_ code: Int, _ message: String?) -> Void	进入房间的结果回调，成功时 code 为0。

观众观看直播的正常调用流程如下：

1. 观众向您的服务端获取最新的直播间列表，可能包含多个直播间的 roomID 和房间信息。

2. 观众选择一个直播间，并调用 enterRoom() 进入该房间。
3. 观众调用 startPlay(userID) 并传入主播的 userID 开始播放。
   - 若直播间列表已包含主播端的 userID 信息，观众端可直接调用 startPlay(userID) 即可开始播放。
   - 若在进房前暂未获取主播的 userID，观众端在进房后会收到 TRTCLiveRoomDelegate 中的 onAnchorEnter(userID) 的事件回调，该回调中携带主播的 userID 信息，再调用 startPlay(userID) 即可播放。

exitRoom

离开房间。

```
/// 退出房间（观众调用）
/// - Parameter callback: 退出房间的结果回调，成功时 code 为0。
/// - Note:
///   - 观众离开直播房间的时候调用
///   - 主播不可调用这个接口离开房间

- (void)exitRoom:(Callback _Nullable)callback
NS_SWIFT_NAME(exitRoom(callback:));
```

参数如下表所示：

参数	类型	含义
callback	(_ code: Int, _ message: String?) -> Void	退出房间的结果回调，成功时 code 为0。

getRoomInfos

获取房间列表的详细信息，房间信息是主播在创建 createRoom() 时通过 roomInfo 设置的。

> **说明**
> 如果房间列表和房间信息都由您自行管理，可忽略该函数。

```
/// 获取房间列表的详细信息
/// 其中的信息是主播在创建 createRoom() 时通过 roomInfo 设置进来的，如果房间列表和房间信息都由您自行管理，可忽略该函数。
/// - Parameter roomIDs: 房间号列表
/// - Parameter callback: 房间详细信息回调
- (void)getRoomInfosWithRoomIDs:(NSArray<NSNumber *> *)roomIDs
                       callback:(RoomInfoCallback _Nullable)callback
NS_SWIFT_NAME(getRoomInfos(roomIDs:callback:));
```

参数如下表所示：

参数	类型	含义
roomIDs	[UInt32]	房间号列表。
callback	(_ code: Int, _ message: String?, _ roomList: [TRTCLiveRoomInfo]) -> Void	房间详细信息回调。

getAnchorList

获取房间内所有的主播列表，`enterRoom()` 成功后调用才有效。

```
/// 获取房间内所有的主播列表，enterRoom() 成功后调用才有效。
/// - Parameter callback: 用户详细信息回调
- (void)getAnchorList:(UserListCallback _Nullable)callback
NS_SWIFT_NAME(getAnchorList(callback:));
```

参数如下表所示：

参数	类型	含义
callback	(_ code: Int, _ message: String, _ userList: [TRTCLiveUserInfo]) -> Void	用户详细信息回调。

getAudienceList

获取房间内所有的观众信息，`enterRoom()` 成功后调用才有效。

```
/// 获取房间内所有的观众信息，enterRoom() 成功后调用才有效。
/// - Parameter callback: 用户详细信息回调
- (void)getAudienceList:(UserListCallback _Nullable)callback
NS_SWIFT_NAME(getAudienceList(callback:));
```

参数如下表所示：

参数	类型	含义
callback	(_ code: Int, _ message: String, _ userList: [TRTCLiveUserInfo]) -> Void	用户详细信息回调。

推拉流相关接口函数

startCameraPreview

开启本地视频的预览画面。

```
/// 开启本地视频的预览画面
/// - Parameters:
///   - frontCamera: true：前置摄像头；false：后置摄像头。
///   - view: 承载视频画面的控件。
///   - callback: 操作回调。
- (void)startCameraPreviewWithFrontCamera:(BOOL)frontCamera
                                     view:(UIView *)view
                                 callback:(Callback _Nullable)callback
NS_SWIFT_NAME(startCameraPreview(frontCamera:view:callback:));
```

参数如下表所示：

参数	类型	含义
frontCamera	Bool	true：前置摄像头；false：后置摄像头。
view	UIView	承载视频画面的控件。
callback	(_ code: Int, _ message: String?) -> Void	操作回调。

stopCameraPreview

停止本地视频采集及预览。

```
/// 停止本地视频采集及预览
- (void)stopCameraPreview;
```

startPublish

开始直播（推流），适用于以下场景：

- 主播开播的时候调用
- 观众开始连麦时调用

```
/// 开始直播（推流），适用于如下两种场景：
/// 1. 主播开播的时候调用
/// 2. 观众开始连麦时调用
/// - Parameters:
```

```
///   - streamID: 用于绑定直播 CDN 的 streamId，如果您希望您的观众通过直播 CDN 进行观看，需要指定当前主播的直播 streamId。
///   - callback: 操作回调
- (void)startPublishWithStreamID:(NSString *)streamID
                        callback:(Callback _Nullable)callback
NS_SWIFT_NAME(startPublish(streamID:callback:));
```

参数如下表所示：

参数	类型	含义
streamID	String	用于绑定直播 CDN 的 streamID，如果您希望观众通过直播 CDN 进行观看，需要指定当前主播的直播 streamID。
callback	(_ code: Int, _ message: String?) -> Void	操作回调。

stopPublish

停止直播（推流），适用于以下场景：

- 主播结束直播时调用。
- 观众结束连麦时调用。

```
/// 停止直播（推流），适用于如下两种场景：
/// 1. 主播结束直播时调用
/// 2. 观众结束连麦时调用
/// - Parameter callback: 操作回调。
- (void)stopPublish:(Callback _Nullable)callback
NS_SWIFT_NAME(stopPublish(callback:));
```

参数如下表所示：

参数	类型	含义
callback	(_ code: Int, _ message: String?) -> Void	操作回调。

startPlay

播放远端视频画面，可以在普通观看和连麦场景中调用。

```
/// 播放远端视频画面，可以在普通观看和连麦场景中调用
/// 【普通观看场景】
```

```
/// 1. 如果您的服务器所管理的房间列表中包含每一个房间的主播 userID，则可以直接在
enterRoom() 成功后调用 startPlay(userID) 即可播放主播的画面。
/// 2. 如果您管理的房间列表只有 roomid 也没有关系，观众在 enterRoom() 成功后很快会收到来
自 TRTCLiveRoomDelegate 中的 onAnchorEnter(userID) 回调。
/// 此时使用回调中的 userID 调用 startPlay(userID) 即可播放主播的画面。
///【直播连麦场景】
/// 发起连麦后，主播会收到来自 TRTCLiveRoomDelegate 中的 onAnchorEnter(userID) 回调，
此时使用回调中的 userID 调用 startPlay(userID) 即可播放连麦画面。
/// - Parameters:
///   - userID: 需要观看的用户 ID。
///   - view: 承载视频画面的 view 控件。
///   - callback: 操作回调。
- (void)startPlayWithUserID:(NSString *)userID
                       view:(UIView *)view
                   callback:(Callback _Nullable)callback
NS_SWIFT_NAME(startPlay(userID:view:callback:));
```

参数如下表所示：

参数	类型	含义
userID	String	需要观看的用户 ID。
view	UIView	承载视频画面的 view 控件。
callback	(_ code: Int, _ message: String?) -> Void	操作回调。

- **普通观看场景**
  - 若直播间列表已包含主播端的 userID 信息，观众端可以直接在 `enterRoom()` 成功后调用 `startPlay(userID)` 播放主播的画面。
  - 若在进房前暂未获取主播的 userID，观众端在进房后会收到 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userID)` 的事件回调，该回调中携带主播的 userID 信息，再调用 `startPlay(userID)` 即可播放主播的画面。
- **直播连麦场景**
  发起连麦后，主播会收到来自 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userID)` 回调，此时使用回调中的 userID 调用 `startPlay(userID)` 即可播放连麦画面。

stopPlay

停止渲染远端视频画面。需在 `onAnchorExit()` 回调时，调用该接口。

```
/// 停止渲染远端视频画面
/// - Parameters:
```

```
///   - userID: 对方的用户信息。
///   - callback: 操作回调。
/// - Note:
///   - 在 onAnchorExit 回调时，调用这个接口
- (void)stopPlayWithUserID:(NSString *)userID
                  callback:(Callback _Nullable)callback
NS_SWIFT_NAME(stopPlay(userID:callback:));
```

参数如下表所示：

参数	类型	含义
userID	String	对方的用户信息。
callback	(_ code: Int, _ message: String?) -> Void	操作回调。

主播和观众连麦

requestJoinAnchor

观众请求连麦。

```
/// 观众端请求连麦
/// - Parameters:
///   - reason: 连麦请求原因。
///   - responseCallback: 请求连麦的回调。
/// - Note: 观众发起请求后，主播端会收到`onRequestJoinAnchor`回调
- (void)requestJoinAnchor:(NSString *)reason
                  timeout:(double)timeout
          responseCallback:(ResponseCallback _Nullable)responseCallback
NS_SWIFT_NAME(requestJoinAnchor(reason:timeout:responseCallback:));
```

参数如下表所示：

参数	类型	含义
reason	String	连麦原因。
timeout	long	主播响应回调。
responseCallback	(_ agreed: Bool, _ reason: String?) -> Void	主播响应回调。

主播和观众的连麦流程如下：

1. 观众调用 requestJoinAnchor() 向主播发起连麦请求。
2. 主播会收到 TRTCLiveRoomDelegate 的 onRequestJoinAnchor() 回调通知。
3. 主播调用 responseJoinAnchor() 决定是否接受来自观众的连麦请求。
4. 观众会收到 responseCallback 回调通知，该通知会携带主播的处理结果。
5. 观众如果请求被同意，则调用 startCameraPreview() 开启本地摄像头。
6. 观众调用 startPublish() 正式进入推流状态。
7. 主播一旦观众进入连麦状态，主播会收到 TRTCLiveRoomDelegate 的 onAnchorEnter() 通知。
8. 主播调用 startPlay() 即可看到连麦观众的视频画面。
9. 观众如果直播间里已有其他观众正在跟主播连麦，新加入的连麦观众会收到 onAnchorEnter() 通知，调用 startPlay() 播放其他连麦者的视频画面。

responseJoinAnchor

主播处理连麦请求。主播在收到 TRTCLiveRoomDelegate 的 onRequestJoinAnchor() 回调后，需要调用该接口来处理观众的连麦请求。

```
/// 主播回复观众连麦请求
/// - Parameters:
///   - user: 观众 ID。
///   - agree: true：同意；false：拒绝。
///   - reason: 同意/拒绝连麦的原因描述。
/// - Note: 主播回复后，观众端会收到`requestJoinAnchor`传入的`responseCallback`回调
- (void)responseJoinAnchor:(NSString *)userID
                     agree:(BOOL)agree
                    reason:(NSString *)reason
NS_SWIFT_NAME(responseJoinAnchor(userID:agree:reason:));
```

参数如下表所示：

参数	类型	含义
userID	String	观众 ID。
agree	Bool	true：同意；false：拒绝。
reason	String?	同意/拒绝连麦的原因描述。

kickoutJoinAnchor

主播移除连麦观众。主播调用此接口移除连麦观众后，被移连麦观众会收到 TRTCLiveRoomDelegate 的 onKickoutJoinAnchor() 回调通知。

```
/// 主播移除连麦观众
/// - Parameters:
///   - userID: 连麦观众 ID。
///   - callback: 操作回调。
/// - Note: 主播调用此接口移除连麦观众后，被移连麦观众会收到
trtcLiveRoomOnKickoutJoinAnchor() 回调通知
- (void)kickoutJoinAnchor:(NSString *)userID
                 callback:(Callback _Nullable)callback
NS_SWIFT_NAME(kickoutJoinAnchor(userID:callback:));
```

参数如下表所示：

参数	类型	含义
userID	String	连麦观众 ID。
callback	(_ code: Int, _ message: String?) -> Void	操作回调。

主播跨房间 PK

requestRoomPK

主播请求跨房 PK。

```
/// 主播请求跨房 PK
/// - Parameters:
///   - roomID: 被邀约房间 ID。
///   - userID: 被邀约主播 ID。
///   - responseCallback: 请求跨房 PK 的结果回调。
/// - Note: 发起请求后，对方主播会收到 `onRequestRoomPK` 回调
- (void)requestRoomPKWithRoomID:(UInt32)roomID
                         userID:(NSString *)userID
                responseCallback:(ResponseCallback _Nullable)responseCallback
NS_SWIFT_NAME(requestRoomPK(roomID:userID:responseCallback:));
```

参数如下表所示：

参数	类型	含义
roomID	UInt32	被邀约房间 ID。
userID	String	被邀约主播 ID。
responseCallba	(_ agreed: Bool, _ reason:	请求跨房 PK 的结果回调。

ck	String?) -> Void	

主播和主播之间可以跨房间 PK，两个正在直播中的主播 A 和 B 之间的跨房 PK 流程如下：

1. 主播 A调用 `requestRoomPK()` 向主播 B 发起连麦请求。
2. 主播 B会收到 `TRTCLiveRoomDelegate` 的 `onRequestRoomPK()` 回调通知。
3. 主播 B调用 `responseRoomPK()` 决定是否接受主播 A 的 PK 请求。
4. 主播 B如果接受主播 A 的请求，等待 `TRTCLiveRoomDelegate` 的 `onAnchorEnter()` 通知，然后调用 `startPlay()` 来显示主播 A 的视频画面。
5. 主播 A会收到 `responseCallback` 回调通知，该通知会携带来自主播 B 的处理结果。
6. 主播 A如果请求被同意，等待 `TRTCLiveRoomDelegate` 的 `onAnchorEnter()` 通知，然后调用 `startPlay()` 显示主播 B 的视频画面。

responseRoomPK

主播响应跨房 PK 请求。主播响应后，对方主播会收到 `requestRoomPK` 传入的 `responseCallback` 回调。

```
/// 响应跨房 PK 请求
/// 主播响应其他房间主播的 PK 请求。
/// - Parameters:
///   - user: 发起 PK 请求的主播 ID
///   - agree: true：同意；false：拒绝
///   - reason: 同意/拒绝 PK 的原因描述
/// - Note: 主播回复后，对方主播会收到 `requestRoomPK` 传入的 `responseCallback` 回调
- (void)responseRoomPKWithUserID:(NSString *)userID
                           agree:(BOOL)agree
                          reason:(NSString *)reason
NS_SWIFT_NAME(responseRoomPK(userID:agree:reason:));
```

参数如下表所示：

参数	类型	含义
userID	String	发起 PK 请求的主播 ID。
agree	Bool	true：同意；false：拒绝。
reason	String?	同意/拒绝 PK 的原因描述。

quitRoomPK

退出跨房 PK。PK 中的任何一个主播退出跨房 PK 状态后，另一个主播会收到 `TRTCLiveRoomDelegate` 的 `trtcLiveRoomOnQuitRoomPK()` 回调通知。

```
/// 主播退出跨房 PK
/// - Parameter callback: 退出跨房 PK 的结果回调
/// - Note: 当两个主播中的任何一个退出跨房 PK 状态后，另一个主播会收到
`trtcLiveRoomOnQuitRoomPK` 回调通知。
- (void)quitRoomPK:(Callback _Nullable)callback
NS_SWIFT_NAME(quitRoomPK(callback:));
```

参数如下表所示：

参数	类型	含义
callback	(_ code: Int, _ message: String?) -> Void	操作回调。

音视频控制相关接口函数

switchCamera

切换前后摄像头。

```
/// 切换前后摄像头
- (void)switchCamera;
```

setMirror

设置是否镜像展示。

```
/// 设置是否镜像展示
/// - Parameter isMirror: 开启/关闭镜像。
- (void)setMirror:(BOOL)isMirror
NS_SWIFT_NAME(setMirror(isMirror:));
```

参数如下表所示：

参数	类型	含义
isMirror	Bool	开启/关闭镜像。

muteLocalAudio

静音本地音频。

```
/// 静音本地音频。
```

```
/// - Parameter isMuted: true：开启静音；false：关闭静音。
- (void)muteLocalAudio:(BOOL)isMuted
NS_SWIFT_NAME(muteLocalAudio(isMuted:));
```

参数如下表所示：

参数	类型	含义
isMuted	Bool	true：开启静音；false：关闭静音。

muteRemoteAudio

静音远端音频。

```
/// 静音远端音频
/// - Parameters:
///   - userID: 远端的用户ID。
///   - isMuted: true：开启静音；false：关闭静音。
- (void)muteRemoteAudioWithUserID:(NSString *)userID isMuted:(BOOL)isMuted
NS_SWIFT_NAME(muteRemoteAudio(userID:isMuted:));
```

参数如下表所示：

参数	类型	含义
userID	String	远端的用户 ID。
isMuted	Bool	true：开启静音；false：关闭静音。

muteAllRemoteAudio

静音所有远端音频。

```
/// 静音所有远端音频
/// - Parameter isMuted: true：开启静音；false：关闭静音。
- (void)muteAllRemoteAudio:(BOOL)isMuted
NS_SWIFT_NAME(muteAllRemoteAudio(_:));
```

参数如下表所示：

参数	类型	含义
isMuted	Bool	true：开启静音；false：关闭静音。

setAudioQuality

设置音频质量

```
/// 设置音频质量，支持的值为1 2 3，代表低中高
/// - Parameter quality 音频质量
- (void)setAudioQuality:(NSInteger)quality
NS_SWIFT_NAME(setAudioiQuality(quality:));
```

参数如下表所示：

参数	类型	含义
quality	NSInteger	1：语音；2：标准； 3：音乐。

背景音乐音效相关接口函数

getAudioEffectManager

获取背景音乐音效管理对象 TXAudioEffectManager 。

```
/// 获取音效管理对象
- (TXAudioEffectManager *)getAudioEffectManager;
```

美颜滤镜相关接口函数

getBeautyManager

获取美颜管理对象 TXBeautyManager 。

```
/* 获取美颜管理对象 TXBeautyManager
*
* 通过美颜管理，您可以使用以下功能：
* - 设置"美颜风格"、“美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小鼻”、“亮
眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。
* - 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”
* - 设置人脸挂件（素材）等动态效果
* - 添加美妆
* - 进行手势识别
*/
- (TXBeautyManager *)getBeautyManager;
```

通过美颜管理，您可以使用以下功能：

- 设置“美颜风格”、“美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小

鼻”、“亮眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。

- 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”。
- 设置人脸挂件（素材）等动态效果。
- 添加美妆。
- 进行手势识别。

消息发送相关接口函数

sendRoomTextMsg

在房间中广播文本消息，一般用于弹幕聊天。

```
/// 发送文本消息，房间内所有成员都可见
/// - Parameters:
///   - message: 文本消息。
///   - callback: 发送回调。
- (void)sendRoomTextMsg:(NSString *)message callback:(Callback _Nullable)callback
NS_SWIFT_NAME(sendRoomTextMsg(message:callback:));
```

参数如下表所示：

参数	类型	含义
message	String	文本消息。
callback	(_ code: Int, _ message: String?) -> Void	发送结果回调。

sendRoomCustomMsg

发送自定义文本消息。

```
/// 发送自定义消息
/// - Parameters:
///   - command: 命令字，由开发者自定义，主要用于区分不同消息类型
///   - message: 本文消息。
///   - callback: 发送回调。
- (void)sendRoomCustomMsgWithCommand:(NSString *)command message:(NSString
*)message callback:(Callback _Nullable)callback
NS_SWIFT_NAME(sendRoomCustomMsg(command:message:callback:));
```

参数如下表所示：

参数	类型	含义

command	String	命令字，由开发者自定义，主要用于区分不同消息类型。
message	String	文本消息。
callback	(_ code: Int, _ message: String?) -> Void	发送结果回调。

调试相关接口函数

showVideoDebugLog

是否在界面中展示 Debug 信息。

```
/// 是否在界面中展示debug信息
/// - Parameter isShow: 开启/关闭 Debug 信息显示。
- (void)showVideoDebugLog:(BOOL)isShow
NS_SWIFT_NAME(showVideoDebugLog(_:));
```

参数如下表所示：

参数	类型	含义
isShow	Bool	开启/关闭 Debug 信息显示。

TRTCLiveRoomDelegate事件回调

通用事件回调

onError

错误回调。

> **说明**
> SDK 不可恢复的错误，一定要监听，并分情况给用户适当的界面提示。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
        onError:(NSInteger)code
        message:(NSString  *)message
NS_SWIFT_NAME(trtcLiveRoom(_:onError:message:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
code	Int	错误码。
message	String?	错误信息。

onWarning

警告回调。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
          onWarning:(NSInteger)code
            message:(NSString *)message
NS_SWIFT_NAME(trtcLiveRoom(_:onWarning:message:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
code	Int	错误码 TRTCWarningCode。
message	String?	警告信息。

onDebugLog

Log 回调。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
         onDebugLog:(NSString *)log
NS_SWIFT_NAME(trtcLiveRoom(_:onDebugLog:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
log	String	日志信息。

房间事件回调

onRoomDestroy

房间被销毁的回调。主播退房时，房间内的所有用户都会收到此通知。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
      onRoomDestroy:(NSString *)roomID
NS_SWIFT_NAME(trtcLiveRoom(_:onRoomDestroy:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
roomID	String	房间 ID。

onRoomInfoChange

直播房间信息变更回调。多用于直播连麦、PK下房间状态变化通知场景。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
    onRoomInfoChange:(TRTCLiveRoomInfo *)info
NS_SWIFT_NAME(trtcLiveRoom(_:onRoomInfoChange:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
info	TRTCLiveRoomInfo	房间信息。

主播和观众进出事件回调

onAnchorEnter

收到新主播进房通知。连麦观众和跨房 PK 主播进房后观众会收到新主播的进房事件，您可以调用 `TRTCLiveRoom` 的 `startPlay()` 显示该主播的视频画面。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
      onAnchorEnter:(NSString *)userID
NS_SWIFT_NAME(trtcLiveRoom(_:onAnchorEnter:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
userID	String	新进房用户 ID。

onAnchorExit

收到主播退房通知。房间内的主播（和连麦中的观众）会收到新主播的退房事件，您可以调用 `TRTCLiveRoom` 的 `stopPlay()` 关闭该主播的视频画面。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
       onAnchorExit:(NSString *)userID
NS_SWIFT_NAME(trtcLiveRoom(_:onAnchorExit:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
userID	String	退房用户 ID。

onAudienceEnter

收到观众进房通知。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
    onAudienceEnter:(TRTCLiveUserInfo *)user
NS_SWIFT_NAME(trtcLiveRoom(_:onAudienceEnter:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
user	TRTCLiveUserInfo	进房观众信息。

onAudienceExit

收到观众退房通知。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
```

```
    onAudienceExit:(TRTCLiveUserInfo *)user
NS_SWIFT_NAME(trtcLiveRoom(_:onAudienceExit:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
user	TRTCLiveUserInfo	退房观众信息。

主播和观众连麦事件回调

onRequestJoinAnchor

主播收到观众连麦请求时的回调。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
 onRequestJoinAnchor:(TRTCLiveUserInfo *)user
              reason:(NSString * _Nullable)reason
             timeout:(double)timeout
NS_SWIFT_NAME(trtcLiveRoom(_:onRequestJoinAnchor:reason:timeout:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
user	TRTCLiveUserInfo	请求连麦观众信息。
reason	String?	连麦原因描述。
timeout	Double	处理请求的超时时间。

onKickoutJoinAnchor

连麦观众收到被移出连麦的通知。连麦观众收到被主播移除连麦的消息，您需要调用 `TRTCLiveRoom` 的 `stopPublish()` 退出连麦。

```
- (void)trtcLiveRoomOnKickoutJoinAnchor:(TRTCLiveRoom *)liveRoom
NS_SWIFT_NAME(trtcLiveRoomOnKickoutJoinAnchor(_:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。

主播 PK 事件回调

onRequestRoomPK

收到请求跨房 PK 通知。主播收到其他房间主播的 PK 请求，如果同意 PK ，您需要等待 `TRTCLiveRoomDelegate` 的 `onAnchorEnter()` 通知，然后调用 `startPlay()` 来播放邀约主播的流。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
    onRequestRoomPK:(TRTCLiveUserInfo *)user
            timeout:(double)timeout
NS_SWIFT_NAME(trtcLiveRoom(_:onRequestRoomPK:timeout:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
user	TRTCLiveUserInfo	发起跨房连麦的主播信息。
timeout	Double	处理请求的超时时间。

onQuitRoomPK

收到断开跨房 PK 通知。

```
- (void)trtcLiveRoomOnQuitRoomPK:(TRTCLiveRoom *)liveRoom
NS_SWIFT_NAME(trtcLiveRoomOnQuitRoomPK(_:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。

消息事件回调

onRecvRoomTextMsg

收到文本消息。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
   onRecvRoomTextMsg:(NSString *)message
            fromUser:(TRTCLiveUserInfo *)user
NS_SWIFT_NAME(trtcLiveRoom(_:onRecvRoomTextMsg:fromUser:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
message	String	文本消息。
user	TRTCLiveUserInfo	发送者用户信息。

onRecvRoomCustomMsg

收到自定义消息。

```
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom
onRecvRoomCustomMsgWithCommand:(NSString *)command
             message:(NSString *)message
            fromUser:(TRTCLiveUserInfo *)user
NS_SWIFT_NAME(trtcLiveRoom(_:onRecvRoomCustomMsg:message:fromUser:));
```

参数如下表所示：

参数	类型	含义
trtcLiveRoom	TRTCLiveRoomImpl	当前 TRTCLiveRoom 组件实例。
command	String	命令字，由开发者自定义，主要用于区分不同消息类型。
message	String	文本消息。
user	TRTCLiveUserInfo	发送者用户信息。

TRTCLiveRoom API (Android)

最近更新时间：2023-08-25 15:47:43

TRTCLiveRoom 是基于腾讯云实时音视频（TRTC）和即时通信 IM 服务组合而成的，支持以下功能：

- 主播创建新的直播间开播，观众进入直播间观看。
- 主播和观众进行视频连麦互动。
- 两个不同房间的主播 PK 互动。
- 支持发送各种文本消息和自定义消息，自定义消息可用于实现弹幕、点赞和礼物。

> **说明**
> TUIKit 系列组件同时使用了腾讯云 实时音视频 TRTC 和 即时通信 IM 两个基础 PaaS 服务，开通实时音视频后会同步开通即时通信IM服务。即时通信 IM 服务详细计费规则请参见 即时通信 – 价格说明，TRTC 开通会默认关联开通 IM SDK 的体验版，仅支持100个 DAU。

TRTCLiveRoom 是一个开源的 Class，依赖腾讯云的两个闭源 SDK，具体的实现过程请参见 视频连麦直播（Android）。

- TRTC SDK：使用 TRTC SDK 作为低延时直播组件。
- IM SDK：使用 IM SDK 的 AVChatroom 实现直播聊天室的功能，同时，通过 IM 消息串联主播间的连麦流程。

TRTCLiveRoom API 概览

SDK 基础函数

API	描述
sharedInstance	获取单例对象。
destroySharedInstance	销毁单例对象。
setDelegate	设置事件回调。
setDelegateHandler	设置事件回调所在的线程。
login	登录。
logout	登出。
setSelfProfile	修改个人信息。

房间相关接口函数

API	描述
createRoom	创建房间（主播调用），若房间不存在，系统将自动创建一个新房间。
destroyRoom	销毁房间（主播调用）。
enterRoom	进入房间（观众调用）。
exitRoom	离开房间（观众调用）。
getRoomInfos	获取房间列表的详细信息。
getAnchorList	获取房间内所有的主播列表，enterRoom() 成功后调用才有效。
getAudienceList	获取房间内所有的观众信息，enterRoom() 成功后调用才有效。

推拉流相关接口函数

API	描述
startCameraPreview	开启本地视频的预览画面。
stopCameraPreview	停止本地视频采集及预览。
startPublish	开始直播（推流）。
stopPublish	停止直播（推流）。
startPlay	播放远端视频画面，可以在普通观看和连麦场景中调用。
stopPlay	停止渲染远端视频画面。

主播和观众连麦

API	描述
requestJoinAnchor	观众请求连麦。
responseJoinAnchor	主播处理连麦请求。
kickoutJoinAnchor	主播踢除连麦观众。

主播跨房间 PK

API	描述
requestRoomPK	主播请求跨房 PK。

responseRoomPK	主播响应跨房 PK 请求。
quitRoomPK	退出跨房 PK。

音视频控制相关接口函数

API	描述
switchCamera	切换前后摄像头。
setMirror	设置是否镜像展示。
muteLocalAudio	静音本地音频。
muteRemoteAudio	静音远端音频。
muteAllRemoteAudio	静音所有远端音频。

背景音乐音效相关接口函数

API	描述
getAudioEffectManager	获取背景音乐音效管理对象 TXAudioEffectManager 。

美颜滤镜相关接口函数

API	描述
getBeautyManager	获取美颜管理对象 TXBeautyManager 。

消息发送相关接口函数

API	描述
sendRoomTextMsg	在房间中广播文本消息，一般用于弹幕聊天。
sendRoomCustomMsg	发送自定义文本消息。

调试相关接口函数

API	描述
showVideoDebugLog	是否在界面中展示 debug 信息。

TRTCLiveRoomDelegate API 概览

通用事件回调

API	描述
onError	错误回调。
onWarning	警告回调。
onDebugLog	Log 回调。

房间事件回调

API	描述
onRoomDestroy	房间被销毁的回调。
onRoomInfoChange	直播房间信息变更回调。

主播和观众进出事件回调

API	描述
onAnchorEnter	收到新主播进房通知。
onAnchorExit	收到主播退房通知。
onAudienceEnter	收到观众进房通知。
onAudienceExit	收到观众退房通知。

主播和观众连麦事件回调

API	描述
onRequestJoinAnchor	主播收到观众连麦请求时的回调。
onKickoutJoinAnchor	连麦观众收到被踢出连麦的通知。

主播 PK 事件回调

API	描述
onRequestRoomPK	收到请求跨房 PK 通知。
onQuitRoomPK	收到断开跨房 PK 通知。

消息事件回调

API	描述
onRecvRoomTextMsg	收到文本消息。
onRecvRoomCustomMsg	收到自定义消息。

SDK 基础函数

sharedInstance

获取 TRTCLiveRoom 单例对象。

```
public static synchronized TRTCLiveRoom sharedInstance(Context context);
```

参数如下表所示：

参数	类型	含义
context	Context	Android 上下文，内部会转为 ApplicationContext 用于系统 API 调用

destroySharedInstance

销毁 TRTCLiveRoom 单例对象。

> **说明**
> 销毁实例后，外部缓存的 TRTCLiveRoom 实例无法再使用，需要重新调用 sharedInstance 获取新实例。

```
public static void destroySharedInstance();
```

setDelegate

TRTCLiveRoom 事件回调，您可以通过 TRTCLiveRoomDelegate 获得 TRTCLiveRoom 的各种状态通知。

```
public abstract void setDelegate(TRTCLiveRoomDelegate delegate);
```

> **说明**
> setDelegate 是 TRTCLiveRoom 的代理回调。

setDelegateHandler

设置事件回调所在的线程。

```
public abstract void setDelegateHandler(Handler handler);
```

参数如下表所示：

参数	类型	含义
handler	Handler	TRTCLiveRoom 中的各种状态通知回调会通过该 handler 通知给您，请勿与 setDelegate 混用。

login

登录。

```
public abstract void login(int sdkAppId,
 String userId, String userSig,
 TRTCLiveRoomDef.TRTCLiveRoomConfig config,
 TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
sdkAppID	Int	您可以在实时音视频控制台 >应用管理> 应用信息中查看 SDKAppID。
userID	String	当前用户的 ID，字符串类型，只允许包含英文字母（a-z 和 A-Z）、数字（0-9）、连词符（-）和下划线（_）。
userSig	String	腾讯云设计的一种安全保护签名，获取方式请参见 如何计算及使用 UserSig。
config	TRTCLiveRoomConfig	全局配置信息，请在登录时初始化，登录之后不可变更。u • seCDNFirst 属性：用于设置观众观看方式。true 表示普通观众通过 CDN 观看，计费便宜但延时较高。false 表示普通观众通过低延时观看，计费价格介于 CDN 和连麦之间，但延迟可控制在1s以内。 • CDNPlayDomain 属性：在 useCDNFirst 设置为 true 时才会生效，用于指定 CDN 观看的播放域名，您可以登录直播控制台 >域名管理 页面中进行设置。

callback	(_ code: Int, _ message: String?) -> Void	登录回调，成功时 code 为0。

logout

登出。

```
public abstract void logout(TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
callback	ActionCallback	登出回调，成功时 code 为0。

setSelfProfile

修改个人信息。

```
public abstract void setSelfProfile(String userName, String avatarURL,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
userName	String	昵称。
avatarURL	String	头像地址。
callback	ActionCallback	个人信息设置回调，成功时 code 为0。

房间相关接口函数

createRoom

创建房间（主播调用）。

```
public abstract void createRoom(int roomId, TRTCLiveRoomDef.TRTCCreateRoomParam
roomParam, TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
roomId	int	房间标识，需要由您分配并进行统一管理。多个 roomID 可以汇总成一个直播间列表，腾讯云暂不提供直播间列表的管理服务，请自行管理您的直播间列表。
roomParam	TRTCCreateRoomParam	房间信息，用于房间描述的信息，例如房间名称，封面信息等。如果房间列表和房间信息都由您的服务器自行管理，可忽略该参数。
callback	ActionCallback	创建房间的结果回调，成功时 code 为0。

主播开播的正常调用流程如下：

1. **主播** 调用 `startCameraPreview()` 打开摄像头预览，此时可以调整美颜参数。
2. **主播** 调用 `createRoom()` 创建直播间，房间创建成功与否会通过 ActionCallback 通知给主播。
3. **主播** 调用 `startPublish()` 开始推流。

destroyRoom

销毁房间（主播调用）。主播在创建房间后，可以调用该函数来销毁房间。

```
public abstract void destroyRoom(TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
callback	ActionCallback	销毁房间的结果回调，成功时 code 为0。

enterRoom

进入房间（观众调用）。

```
public abstract void enterRoom(int roomId, TRTCLiveRoomCallback.ActionCallback
callback);
```

参数如下表所示：

参数	类型	含义
roomId	int	房间标识。
callback	ActionCallback	进入房间的结果回调，成功时 code 为0。

观众观看直播的正常调用流程如下：

1. **观众** 向您的服务端获取最新的直播间列表，可能包含多个直播间的 roomID 和房间信息。
2. **观众** 观众选择一个直播间，并调用 `enterRoom()` 进入该房间。
3. **观众** 调用 `startPlay(userId)` 并传入主播的 userId 开始播放。
   - 若直播间列表已包含主播端的 userId 信息，观众端可直接调用 `startPlay(userId)` 即可开始播放。
   - 若在进房前暂未获取主播的 userId，观众端在进房后会收到 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userId)` 的事件回调，该回调中携带主播的 userId 信息，再调用 `startPlay(userId)` 即可播放。

exitRoom

离开房间。

```
public abstract void exitRoom(TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
callback	ActionCallback	退出房间的结果回调，成功时 code 为0。

getRoomInfos

获取房间列表的详细信息，房间信息是主播在创建 `createRoom()` 时通过 roomInfo 设置的。

> **说明**
> 如果房间列表和房间信息都由您自行管理，可忽略该函数。

```
public abstract void getRoomInfos(List<Integer> roomIdList,
TRTCLiveRoomCallback.RoomInfoCallback callback);
```

参数如下表所示：

参数	类型	含义
roomIdList	List<Integer>	房间号列表。
callback	RoomInfoCallback	房间详细信息回调。

getAnchorList

获取房间内所有的主播列表， `enterRoom()` 成功后调用才有效。

```
public abstract void getAnchorList(TRTCLiveRoomCallback.UserListCallback callback);
```

参数如下表所示：

参数	类型	含义
callback	UserListCallback	用户详细信息回调。

getAudienceList

获取房间内所有的观众信息，`enterRoom()` 成功后调用才有效。

```
public abstract void getAudienceList(TRTCLiveRoomCallback.UserListCallback callback);
```

参数如下表所示：

参数	类型	含义
callback	UserListCallback	用户详细信息回调。

推拉流相关接口函数

startCameraPreview

开启本地视频的预览画面。

```
public abstract void startCameraPreview(boolean isFront, TXCloudVideoView view,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
isFront	boolean	true：前置摄像头；false：后置摄像头。
view	TXCloudVideoView	承载视频画面的控件。
callback	ActionCallback	操作回调。

stopCameraPreview

停止本地视频采集及预览。

```
public abstract void stopCameraPreview();
```

startPublish

开始直播（推流），适用于以下场景：

- **主播开播的时候调用**
- **观众开始连麦时调用**

```
public abstract void startPublish(String streamId,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
streamId	String	用于绑定直播 CDN 的 streamId，如果您希望观众通过直播 CDN 进行观看，需要指定当前主播的直播 streamId。
callback	ActionCallback	操作回调。

stopPublish

停止直播（推流），适用于以下场景：

- **主播结束直播时调用**
- **观众结束连麦时调用**

```
public abstract void stopPublish(TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
callback	ActionCallback	操作回调。

startPlay

播放远端视频画面，可以在普通观看和连麦场景中调用。

```
public abstract void startPlay(String userId, TXCloudVideoView view,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
userId	String	需要观看的用户id。
view	TXCloudVideoView	承载视频画面的 view 控件。
callback	ActionCallback	操作回调。

普通观看场景

- 若直播间列表已包含主播端的 userId 信息，观众端可以直接在 `enterRoom()` 成功后调用 `startPlay(userId)` 播放主播的画面。
- 若在进房前暂未获取主播的 userId，观众端在进房后会收到 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userId)` 的事件回调，该回调中携带主播的 userId 信息，再调用 `startPlay(userId)` 即可播放主播的画面。

直播连麦场景

发起连麦后，主播会收到来自 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userId)` 回调，此时使用回调中的 userId 调用 startPlay(userId) 即可播放连麦画面。

stopPlay

停止渲染远端视频画面。需在 `onAnchorExit()` 回调时，调用该接口。

```
public abstract void stopPlay(String userId, TRTCLiveRoomCallback.ActionCallback
callback);
```

参数如下表所示：

参数	类型	含义
userId	String	对方的用户信息。
callback	ActionCallback	操作回调。

主播和观众连麦

requestJoinAnchor

观众请求连麦。

```
public abstract void requestJoinAnchor(String reason, int timeout,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
reason	String	连麦原因。
timeout	int	超时时间。
callback	ActionCallback	主播响应回调。

主播和观众的连麦流程如下：

1. 观众调用 `requestJoinAnchor()` 向主播发起连麦请求。
2. 主播会收到 `TRTCLiveRoomDelegate` 的 `onRequestJoinAnchor()` 回调通知。
3. 主播调用 `responseJoinAnchor()` 决定是否接受来自观众的连麦请求。
4. 观众会收到 responseCallback 回调通知，该通知会携带主播的处理结果。
5. 观众如果请求被同意，则调用 `startCameraPreview()` 开启本地摄像头。
6. 观众然后调用 `startPublish()` 正式进入推流状态。
7. 主播一旦观众进入连麦状态，主播会收到 `TRTCLiveRoomDelegate` 的 `onAnchorEnter()` 通知。
8. 主播调用 `startPlay()` 即可看到连麦观众的视频画面。
9. 观众如果直播间里已有其他观众正在跟主播连麦，新加入的连麦观众会收到 `onAnchorEnter()` 通知，调用 `startPlay()` 播放其他连麦者的视频画面。

responseJoinAnchor

主播处理连麦请求。主播在收到 `TRTCLiveRoomDelegate` 的 `onRequestJoinAnchor()` 回调后需要调用此接口来处理观众的连麦请求。

```
public abstract void responseJoinAnchor(String userId, boolean agree, String reason);
```

参数如下表所示：

参数	类型	含义
userId	String	观众 ID。
agree	boolean	true：同意；false：拒绝。
reason	String	同意/拒绝连麦的原因描述。

kickoutJoinAnchor

主播踢除连麦观众。主播调用此接口踢除连麦观众后，被踢连麦观众会收到 TRTCLiveRoomDelegate 的 onKickoutJoinAnchor() 回调通知。

```
public abstract void kickoutJoinAnchor(String userId,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
userId	String	连麦观众 ID。
callback	ActionCallback	操作回调。

主播跨房间 PK

requestRoomPK

主播请求跨房 PK。

```
public abstract void requestRoomPK(int roomId, String userId,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
roomId	int	被邀约房间 ID。
userId	String	被邀约主播 ID。
callback	ActionCallback	请求跨房 PK 的结果回调。

主播和主播之间可以跨房间 PK，两个正在直播中的主播 A 和 B 之间的跨房 PK 流程如下：

1. 主播 A调用 requestRoomPK() 向主播 B 发起连麦请求。
2. 主播 B会收到 TRTCLiveRoomDelegate 的 onRequestRoomPK() 回调通知。
3. 主播 B调用 responseRoomPK() 决定是否接受主播 A 的 PK 请求。
4. 主播 B如果接受主播 A 的要求，等待 TRTCLiveRoomDelegate 的 onAnchorEnter() 通知，然后调用 startPlay() 来显示主播 A 的视频画面。
5. 主播 A会收到 responseCallback 回调通知，该通知会携带来自主播 B 的处理结果。
6. 主播 A如果请求被同意，等待 TRTCLiveRoomDelegate 的 onAnchorEnter() 通知，然后调用 startPlay() 显示主播 B 的视频画面。

responseRoomPK

主播响应跨房 PK 请求。主播响应后，对方主播会收到 `requestRoomPK` 传入的 `responseCallback` 回调。

```
public abstract void responseRoomPK(String userId, boolean agree, String reason);
```

参数如下表所示：

参数	类型	含义
userId	String	发起 PK 请求的主播 ID。
agree	boolean	true：同意；false：拒绝。
reason	String	同意/拒绝 PK 的原因描述。

quitRoomPK

退出跨房 PK。PK 中的任何一个主播退出跨房 PK 状态后，另一个主播会收到 `TRTCLiveRoomDelegate` 的 `onQuitRoomPk()` 回调通知。

```
public abstract void quitRoomPK(TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
callback	ActionCallback	操作回调。

音视频控制相关接口函数

switchCamera

切换前后摄像头。

```
public abstract void switchCamera();
```

setMirror

设置是否镜像展示。

```
public abstract void setMirror(boolean isMirror);
```

参数如下表所示：

参数	类型	含义
isMirror	boolean	开启/关闭镜像。

muteLocalAudio

静音本地音频。

```
public abstract void muteLocalAudio(boolean mute);
```

参数如下表所示：

参数	类型	含义
mute	boolean	true：开启静音；false：关闭静音。

muteRemoteAudio

静音远端音频。

```
public abstract void muteRemoteAudio(String userId, boolean mute);
```

参数如下表所示：

参数	类型	含义
userId	String	远端的用户 ID。
mute	boolean	true：开启静音；false：关闭静音。

muteAllRemoteAudio

静音所有远端音频。

```
public abstract void muteAllRemoteAudio(boolean mute);
```

参数如下表所示：

参数	类型	含义
mute	boolean	true：开启静音；false：关闭静音。

背景音乐音效相关接口函数

getAudioEffectManager

获取背景音乐音效管理对象 TXAudioEffectManager 。

```
public abstract TXAudioEffectManager getAudioEffectManager();
```

美颜滤镜相关接口函数

getBeautyManager

获取美颜管理对象 TXBeautyManager 。

```
public abstract TXBeautyManager getBeautyManager();
```

通过美颜管理，您可以使用以下功能：

- 设置“美颜风格”、“美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小鼻”、“亮眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。
- 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”。
- 设置人脸挂件（素材）等动态效果。
- 添加美妆。
- 进行手势识别。

消息发送相关接口函数

sendRoomTextMsg

在房间中广播文本消息，一般用于弹幕聊天。

```
public abstract void sendRoomTextMsg(String message,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
message	String	文本消息。
callback	ActionCallback	发送结果回调。

sendRoomCustomMsg

发送自定义文本消息。

```
public abstract void sendRoomCustomMsg(String cmd, String message,
TRTCLiveRoomCallback.ActionCallback callback);
```

参数如下表所示：

参数	类型	含义
cmd	String	命令字，由开发者自定义，主要用于区分不同消息类型。
message	String	文本消息。
callback	ActionCallback	发送结果回调。

调试相关接口函数

showVideoDebugLog

是否在界面中展示debug信息。

```
public abstract void showVideoDebugLog(boolean isShow);
```

参数如下表所示：

参数	类型	含义
isShow	boolean	开启/关闭 Debug 信息显示。

TRTCLiveRoomDelegate事件回调

通用事件回调

onError

错误回调。

> **说明**
> SDK 不可恢复的错误，一定要监听，并分情况给用户适当的界面提示。

```
void onError(int code, String message);
```

参数如下表所示：

参数	类型	含义
code	int	错误码。
message	String	错误信息。

onWarning

警告回调。

```
void onWarning(int code, String message);
```

参数如下表所示：

参数	类型	含义
code	int	错误码。
message	String	警告信息。

onDebugLog

Log 回调。

```
void onDebugLog(String message);
```

参数如下表所示：

参数	类型	含义
message	String	日志信息。

房间事件回调

onRoomDestroy

房间被销毁的回调。主播退房时，房间内的所有用户都会收到此通知。

```
void onRoomDestroy(String roomId);
```

参数如下表所示：

参数	类型	含义

roomId	String	房间 ID。

onRoomInfoChange

直播房间信息变更回调。多用于直播连麦、PK下房间状态变化通知场景。

```
void onRoomInfoChange(TRTCLiveRoomDef.TRTCLiveRoomInfo roomInfo);
```

参数如下表所示：

参数	类型	含义
roomInfo	TRTCLiveRoomInfo	房间信息。

主播和观众进出事件回调

onAnchorEnter

收到新主播进房通知。连麦观众和跨房 PK 主播进房后观众会收到新主播的进房事件，您可以调用 `TRTCLiveRoom` 的 `startPlay()` 显示该主播的视频画面。

```
void onAnchorEnter(String userId);
```

参数如下表所示：

参数	类型	含义
userId	String	新进房主播 ID。

onAnchorExit

收到主播退房通知。房间内的主播（和连麦中的观众）会收到新主播的退房事件，您可以调用 `TRTCLiveRoom` 的 `stopPlay()` 关闭该主播的视频画面。

```
void onAnchorExit(String userId);
```

参数如下表所示：

参数	类型	含义
userId	String	退房用户 ID。

onAudienceEnter

收到观众进房通知。

```
void onAudienceEnter(TRTCLiveRoomDef.TRTCLiveUserInfo userInfo);
```

参数如下表所示：

参数	类型	含义
userInfo	TRTCLiveUserInfo	进房观众信息。

onAudienceExit

收到观众退房通知。

```
void onAudienceExit(TRTCLiveRoomDef.TRTCLiveUserInfo userInfo);
```

参数如下表所示：

参数	类型	含义
userInfo	TRTCLiveUserInfo	退房观众信息。

主播和观众连麦事件回调

onRequestJoinAnchor

主播收到观众连麦请求时的回调。

```
void onRequestJoinAnchor(TRTCLiveRoomDef.TRTCLiveUserInfo userInfo, String reason,
int timeOut);
```

参数如下表所示：

参数	类型	含义
userInfo	TRTCLiveUserInfo	请求连麦观众信息。
reason	String	连麦原因描述。
timeout	int	处理请求的超时时间，如果上层超过该时间没有处理，则会自动将该次请求废弃。

onKickoutJoinAnchor

连麦观众收到被踢出连麦的通知。连麦观众收到被主播踢除连麦的消息，您需要调用 TRTCLiveRoom 的 stopPublish() 退出连麦。

```
void onKickoutJoinAnchor();
```

主播 PK 事件回调

onRequestRoomPK

收到请求跨房 PK 通知。主播收到其他房间主播的 PK 请求，如果同意 PK ，您需要等待 TRTCLiveRoomDelegate 的 onAnchorEnter() 通知，然后调用 startPlay() 来播放邀约主播的流。

```
void onRequestRoomPK(TRTCLiveRoomDef.TRTCLiveUserInfo userInfo, int timeout);
```

参数如下表所示：

参数	类型	含义
userInfo	TRTCLiveUserInfo	发起跨房连麦的主播信息。
timeout	int	处理请求的超时时间。

onQuitRoomPK

收到断开跨房 PK 通知。

```
void onQuitRoomPK();
```

消息事件回调

onRecvRoomTextMsg

收到文本消息。

```
void onRecvRoomTextMsg(String message, TRTCLiveRoomDef.TRTCLiveUserInfo
userInfo);
```

参数如下表所示：

参数	类型	含义

message	String	文本消息。
userInfo	TRTCLiveUserInfo	发送者用户信息。

onRecvRoomCustomMsg

收到自定义消息。

```
void onRecvRoomCustomMsg(String cmd, String message,
TRTCLiveRoomDef.TRTCLiveUserInfo userInfo);
```

参数如下表所示：

参数	类型	含义
command	String	命令字，由开发者自定义，主要用于区分不同消息类型。
message	String	文本消息。
userInfo	TRTCLiveUserInfo	发送者用户信息。

TRTCAudioEffectManager

playBGM

播放背景音乐。

```
void playBGM(String url, int loopTimes, int bgmVol, int micVol, TRTCCloud.BGMNotify
notify);
```

参数如下表所示：

参数	类型	含义
url	String	背景音乐文件路径。
loopTimes	int	循环次数
bgmVol	int	BGM 音量
micVol	int	采集音量
notify	TRTCCloud.BGMNotify	播放通知

stopBGM

停止播放背景音乐。

```
void stopBGM();
```

pauseBGM

暂停播放背景音乐。

```
void pauseBGM();
```

resumeBGM

继续播放背景音乐。

```
void resumeBGM();
```

setBGMVolume

设置背景音乐的音量大小，播放背景音乐混音时使用，用来控制背景音的音量大小。

```
void setBGMVolume(int volume);
```

参数如下表所示：

参数	类型	含义
volume	int	音量大小，100表示正常音量，取值范围为0 – 100，默认值为100。

setBGMPosition

设置背景音乐播放进度。

```
int setBGMPosition(int position);
```

参数如下表所示：

参数	类型	含义
position	int	背景音乐播放进度，单位为毫秒（ms）。

返回

0：成功。

setMicVolume

设置麦克风的音量大小，播放背景音乐混音时使用，用来控制麦克风音量大小。

```
void setMicVolume(int volume);
```

参数如下表所示：

参数	类型	含义
volume	Int	音量大小，取值0 – 100，默认值为100。

setReverbType

设置混响效果。

```
void setReverbType(int reverbType);
```

参数如下表所示：

参数	类型	含义
reverbType	int	混响类型，详情请参见 TRTCCloudDef 中的 TRTC_REVERB_TYPE 定义。

setVoiceChangerType

设置变声类型。

```
void setVoiceChangerType(int type);
```

参数如下表所示：

参数	类型	含义
type	int	混响类型，详情请参见 TRTCCloudDef 中的 TRTC_VOICE_CHANGER_TYPE 定义。

playAudioEffect

播放音效，每个音效都需要您指定具体的 ID，您可以通过该 ID 对音效的开始、停止、音量等进行设置。支持 aac、mp3 以及 m4a 格式。

```
void playAudioEffect(int effectId, String path, int count, boolean publish, int volume);
```

参数如下表所示：

参数	类型	含义
effectId	int	音效 ID。
path	String	音效路径。
count	int	循环次数。
publish	boolean	是否推送 / true 推送给观众, false 本地预览。
volume	int	音量大小，取值范围为0 – 100，默认值为100。

pauseAudioEffect

暂停音效播放。

```
void pauseAudioEffect(int effectId);
```

参数如下表所示：

参数	类型	含义
effectId	int	音效 ID。

resumeAudioEffect

恢复音效播放。

```
void resumeAudioEffect(int effectId);
```

参数如下表所示：

参数	类型	含义
effectId	int	音效 ID。

stopAudioEffect

停止音效播放。

```
void stopAudioEffect(int effectId);
```

参数如下表所示：

参数	类型	含义
effectId	int	音效 ID。

stopAllAudioEffects

停止全部音效播放。

```
void stopAllAudioEffects();
```

setAudioEffectVolume

设置音效音量。

```
void setAudioEffectVolume(int effectId, int volume);
```

参数如下表所示：

参数	类型	含义
effectId	int	音效 ID。
volume	int	音量大小，取值范围为0 – 100，默认值为100。

setAllAudioEffectsVolume

设置所有音效的音量。

```
void setAllAudioEffectsVolume(int volume);
```

参数如下表所示：

参数	类型	含义
volume	int	音量大小，取值范围为0 – 100，默认值为100。

TRTCLiveRoom API (Flutter)

最近更新时间：2023-08-24 16:22:42

TRTCLiveRoom 是基于腾讯云实时音视频（TRTC）和即时通信 IM 服务组合而成的，支持以下功能：

- 主播创建新的直播间开播，观众进入直播间观看。
- 主播和观众进行视频连麦互动。
- 两个不同房间的主播 PK 互动。
- 支持发送各种文本消息和自定义消息，自定义消息可用于实现弹幕、点赞和礼物。

> **说明**
> TUIKit 系列组件同时使用了腾讯云 实时音视频 TRTC 和 即时通信 IM 两个基础 PaaS 服务，开通实时音视频后会同步开通即时通信 IM 服务。即时通信 IM 服务详细计费规则请参见 即时通信 – 价格说明，TRTC 开通会默认关联开通 IM SDK 的体验版，仅支持100个 DAU。

TRTCLiveRoom 是一个开源的 Class，依赖腾讯云的两个闭源 SDK，具体的实现过程请参见 视频连麦直播（Flutter）。

- TRTC SDK：使用 TRTC SDK 作为低延时直播组件。
- IM SDK：使用 IM SDK 的 AVChatroom 实现直播聊天室的功能，同时，通过 IM 消息串联主播间的连麦流程。

TRTCLiveRoom API 概览

SDK 基础函数

API	描述
sharedInstance	获取单例对象。
destroySharedInstance	销毁单例对象。
registerListener	设置事件回调。
unRegisterListener	设置事件回调所在的线程。
login	登录。
logout	登出。
setSelfProfile	修改个人信息。

房间相关接口函数

API	描述
createRoom	创建房间（主播调用），若房间不存在，系统将自动创建一个新房间。
destroyRoom	销毁房间（主播调用）。
enterRoom	进入房间（观众调用）。
exitRoom	离开房间（观众调用）。
getRoomInfos	获取房间列表的详细信息。
getAnchorList	获取房间内所有的主播列表，enterRoom() 成功后调用才有效。
getRoomMemberList	获取房间内所有的成员信息，enterRoom() 成功后调用才有效。

推拉流相关接口函数

API	描述
startCameraPreview	开启本地视频的预览画面。
stopCameraPreview	停止本地视频采集及预览。
startPublish	开始直播（推流）。
stopPublish	停止直播（推流）。
startPlay	播放远端视频画面，可以在普通观看和连麦场景中调用。
stopPlay	停止渲染远端视频画面。

主播和观众连麦

API	描述
requestJoinAnchor	观众请求连麦。
responseJoinAnchor	主播处理连麦请求。
kickoutJoinAnchor	主播移除连麦观众。

主播跨房间 PK

API	描述
requestRoomPK	主播请求跨房 PK。

responseRoomPK	主播响应跨房 PK 请求。
quitRoomPK	退出跨房 PK。

音视频控制相关接口函数

API	描述
switchCamera	切换前后摄像头。
setMirror	设置是否镜像展示。
muteLocalAudio	静音本地音频。
muteRemoteAudio	静音远端音频。
muteAllRemoteAudio	静音所有远端音频。

背景音乐音效相关接口函数

API	描述
getAudioEffectManager	获取背景音乐音效管理对象 TXAudioEffectManager 。

美颜滤镜相关接口函数

API	描述
getBeautyManager	获取美颜管理对象 TXBeautyManager 。

消息发送相关接口函数

API	描述
sendRoomTextMsg	在房间中广播文本消息，一般用于弹幕聊天。
sendRoomCustomMsg	发送自定义文本消息。

TRTCLiveRoomDelegate API 概览

通用事件回调

API	描述
onError	错误回调。
onWarning	警告回调。

onKickedOffline	其他用户登录了同一账号，被移下线。

房间事件回调

API	描述
onEnterRoom	本地进房回调。
onUserVideoAvailable	远端用户是否存在可播放的主路画面（一般用于摄像头）。
onRoomDestroy	房间被销毁的回调。

主播和观众进出事件回调

API	描述
onAnchorEnter	收到新主播进房通知。
onAnchorExit	收到主播退房通知。
onAudienceEnter	收到观众进房通知。
onAudienceExit	收到观众退房通知。

主播和观众连麦事件回调

API	描述
onRequestJoinAnchor	主播收到观众连麦请求时的回调。
onAnchorAccepted	主播同意观众的连麦请求。
onAnchorRejected	主播拒绝观众的连麦请求。
onKickoutJoinAnchor	连麦观众收到被移出连麦的通知。

主播 PK 事件回调

API	描述
onRequestRoomPK	收到请求跨房 PK 通知。
onRoomPKAccepted	主播接受跨房 PK 请求。
onRoomPKRejected	主播拒绝跨房 PK 请求。
onQuitRoomPK	收到断开跨房 PK 通知。

消息事件回调

API	描述
onRecvRoomTextMsg	收到文本消息。
onRecvRoomCustomMsg	收到自定义消息。

SDK 基础函数

sharedInstance

获取 TRTCLiveRoom 单例对象。

```
static Future<TRTCLiveRoom> sharedInstance()
```

destroySharedInstance

销毁 TRTCLiveRoom 单例对象。

> **说明**
> 销毁实例后，外部缓存的 TRTCLiveRoom 实例无法再使用，需要重新调用 sharedInstance 获取新实例。

```
static void destroySharedInstance()
```

registerListener

TRTCLiveRoom 事件回调，您可以通过 TRTCLiveRoomDelegate 获得 TRTCLiveRoom 的各种状态通知。

```
void registerListener(VoiceListenerFunc func);
```

> **说明**
> registerListener 是 TRTCLiveRoom 的代理回调。

unRegisterListener

移除组件事件监听接口。

```
void unRegisterListener(VoiceListenerFunc func);
```

login

登录。

```
Future<ActionCallback> login(
    int sdkAppId, String userId, String userSig, TRTCLiveRoomConfig config);
```

参数如下表所示：

参数	类型	含义
sdkAppID	Int	您可以在实时音视频控制台 >应用管理> 应用信息中查看 SDKAppID。
userID	String	当前用户的 ID，字符串类型，只允许包含英文字母（a-z 和 A-Z）、数字（0-9）、连词符（-）和下划线（_）。
userSig	String	腾讯云设计的一种安全保护签名，获取方式请参见 如何计算及使用 UserSig。
config	TRTCLiveRoomConfig	全局配置信息，请在登录时初始化，登录之后不可变更。 • useCDNFirst 属性：用于设置观众观看方式。true 表示普通观众通过 CDN 观看，计费便宜但延时较高。false 表示普通观众通过低延时观看，计费价格介于 CDN 和连麦之间，但延迟可控制在1s以内。 • CDNPlayDomain 属性：在 useCDNFirst 设置为 true 时才会生效，用于指定 CDN 观看的播放域名，您可以登录直播控制台 > 域名管理 页面中进行设置。

logout

登出。

```
Future<ActionCallback> logout();
```

setSelfProfile

修改个人信息。

```
Future<ActionCallback> setSelfProfile(String userName, String avatarURL);
```

参数如下表所示：

参数	类型	含义
userName	String	昵称。
avatarURL	String	头像地址。

房间相关接口函数

createRoom

创建房间（主播调用）。

```
Future<ActionCallback> createRoom(int roomId, TRTCCreateRoomParam roomParam);
```

参数如下表所示：

参数	类型	含义
roomId	int	房间标识，需要由您分配并进行统一管理。多个 roomID 可以汇总成一个直播间列表，腾讯云暂不提供直播间列表的管理服务，请自行管理您的直播间列表。
roomParam	RoomParam	房间信息，用于房间描述的信息，例如房间名称，封面信息等。如果房间列表和房间信息都由您的服务器自行管理，可忽略该参数。

主播开播的正常调用流程如下：

1. 主播 调用 `startCameraPreview()` 打开摄像头预览，此时可以调整美颜参数。
2. 主播 调用 `createRoom()` 创建直播间，房间创建成功与否会通过 ActionCallback 通知给主播。
3. 主播 调用 `starPublish()` 开始推流。

destroyRoom

销毁房间（主播调用）。主播在创建房间后，可以调用该函数来销毁房间。

```
Future<ActionCallback> destroyRoom();
```

enterRoom

进入房间（观众调用）。

```
Future<ActionCallback> enterRoom(int roomId);
```

参数如下表所示：

参数	类型	含义
roomId	int	房间标识。

观众观看直播的正常调用流程如下：

1. **观众** 向您的服务端获取最新的直播间列表，可能包含多个直播间的 roomID 和房间信息。
2. **观众** 观众选择一个直播间，并调用 `enterRoom()` 进入该房间。
3. **观众** 调用 `startPlay(userId)` 并传入主播的 userId 开始播放。
   - 若直播间列表已包含主播端的 userId 信息，观众端可直接调用 `startPlay(userId)` 即可开始播放。
   - 若在进房前暂未获取主播的 userId，观众端在进房后会收到 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userId)` 的事件回调，该回调中携带主播的 userId 信息，再调用 `startPlay(userId)` 即可播放。

exitRoom

离开房间。

```
Future<ActionCallback> exitRoom();
```

getRoomInfos

获取房间列表的详细信息，房间信息是主播在创建 `createRoom()` 时通过 roomInfo 设置的。

> **说明**
> 如果房间列表和房间信息都由您自行管理，可忽略该函数。

```
Future<RoomInfoCallback> getRoomInfos(List<String> roomIdList);
```

参数如下表所示：

参数	类型	含义
roomIdList	List<String>	房间号列表。

getAnchorList

获取房间内所有的主播列表， `enterRoom()` 成功后调用才有效。

```
Future<UserListCallback> getAnchorList();
```

getRoomMemberList

获取房间内所有的观众信息， enterRoom() 成功后调用才有效。

```
Future<UserListCallback> getRoomMemberList(int nextSeq)
```

参数如下表所示：

参数	类型	含义
nextSeq	int	分页拉取标志，第一次拉取填0，回调成功如果 nextSeq 不为零，需要分页，传入再次拉取，直至为0。

推拉流相关接口函数

startCameraPreview

开启本地视频的预览画面。

```
Future<void> startCameraPreview(bool isFrontCamera, int viewId);
```

参数如下表所示：

参数	类型	含义
isFrontCamera	bool	true：前置摄像头；false：后置摄像头。
viewId	int	视频 view 的回调 ID。

stopCameraPreview

停止本地视频采集及预览。

```
Future<void> stopCameraPreview();
```

startPublish

开始直播（推流），适用于以下场景：

- 主播开播的时候调用。
- 观众开始连麦时调用。

```
Future<void> startPublish(String streamId);
```

参数如下表所示：

参数	类型	含义
streamId	String?	用于绑定直播 CDN 的 streamId，如果您希望观众通过直播 CDN 进行观看，需要指定当前主播的直播 streamId。

stopPublish

停止直播（推流），适用于以下场景：

- 主播结束直播时调用。
- 观众结束连麦时调用。

```
Future<void> stopPublish();
```

startPlay

播放远端视频画面，可以在普通观看和连麦场景中调用。

```
Future<void> startPlay(String userId, int viewId);
```

参数如下表所示：

参数	类型	含义
userId	String	需要观看的用户 ID。
viewId	int	视频view的回调 ID。

- **普通观看场景：**
  - 若直播间列表已包含主播端的 userId 信息，观众端可以直接在 `enterRoom()` 成功后调用 `startPlay(userId)` 播放主播的画面。
  - 若在进房前暂未获取主播的 userId，观众端在进房后会收到 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userId)` 的事件回调，该回调中携带主播的 userId 信息，再调用 `startPlay(userId)` 即可播放主播的画面。
- **直播连麦场景：**
  发起连麦后，主播会收到来自 `TRTCLiveRoomDelegate` 中的 `onAnchorEnter(userId)` 回调，此时使用回调中的 userId 调用 `startPlay(userId)` 即可播放连麦画面。

stopPlay

停止渲染远端视频画面。需在 `onAnchorExit()` 回调时，调用该接口。

```
Future<void> stopPlay(String userId);
```

参数如下表所示：

参数	类型	含义
userId	String	对方的用户信息。

主播和观众连麦

requestJoinAnchor

观众请求连麦。

```
Future<ActionCallback> requestJoinAnchor();
```

主播和观众的连麦流程如下：

1. 观众调用 requestJoinAnchor() 向主播发起连麦请求。
2. 主播会收到 TRTCLiveRoomDelegate 的 onRequestJoinAnchor() 回调通知。
3. 主播调用 responseJoinAnchor() 决定是否接受来自观众的连麦请求。
4. 观众会收到 responseCallback 回调通知，该通知会携带主播的处理结果。
5. 观众如果请求被同意，则调用 startCameraPreview() 开启本地摄像头。
6. 观众然后调用 startPublish() 正式进入推流状态。
7. 主播一旦观众进入连麦状态，主播会收到 TRTCLiveRoomDelegate 的 onAnchorEnter() 通知。
8. 主播调用 startPlay() 即可看到连麦观众的视频画面。
9. 观众如果直播间里已有其他观众正在跟主播连麦，新加入的连麦观众会收到 onAnchorEnter() 通知，调用 startPlay() 播放其他连麦者的视频画面。

responseJoinAnchor

主播处理连麦请求。主播在收到 TRTCLiveRoomDelegate 的 onRequestJoinAnchor() 回调后需要调用此接口来处理观众的连麦请求。

```
Future<ActionCallback> responseJoinAnchor(String userId, boolean agreee);
```

参数如下表所示：

参数	类型	含义
userId	String	观众 ID。

agree	bool	true：同意；false：拒绝。

kickoutJoinAnchor

主播移除连麦观众。主播调用此接口移除连麦观众后，被移连麦观众会收到 TRTCLiveRoomDelegate 的 onKickoutJoinAnchor() 回调通知。

```
Future<ActionCallback> kickoutJoinAnchor(String userId);
```

参数如下表所示：

参数	类型	含义
userId	String	连麦观众 ID。

主播跨房间 PK

requestRoomPK

主播请求跨房 PK。

```
Future<ActionCallback> requestRoomPK(int roomId, String userId);
```

参数如下表所示：

参数	类型	含义
roomId	int	被邀约房间 ID。
userId	String	被邀约主播 ID。

主播和主播之间可以跨房间 PK，两个正在直播中的主播 A 和 B 之间的跨房 PK 流程如下：

1. 主播 A调用 requestRoomPK() 向主播 B 发起连麦请求。
2. 主播 B会收到 TRTCLiveRoomDelegate 的 onRequestRoomPK() 回调通知。
3. 主播 B调用 responseRoomPK() 决定是否接受主播 A 的 PK 请求。
4. 主播 B如果接受主播 A 的要求，等待 TRTCLiveRoomDelegate 的 onAnchorEnter() 通知，然后调用 startPlay() 来显示主播 A 的视频画面。
5. 主播 A会收到 onRoomPKAccepted 或 onRoomPKRejected 回调通知。
6. 主播 A如果请求被同意，等待 TRTCLiveRoomDelegate 的 onAnchorEnter() 通知，然后调用 startPlay() 显示主播 B 的视频画面。

responseRoomPK

主播响应跨房 PK 请求。主播响应后，对方主播会收到 `requestRoomPK` 传入的 `responseCallback` 回调。

```
Future<ActionCallback> responseRoomPK(String userId, boolean agree);
```

参数如下表所示：

参数	类型	含义
userId	String	发起 PK 请求的主播 ID。
agree	bool	true：同意；false：拒绝。

quitRoomPK

退出跨房 PK。PK 中的任何一个主播退出跨房 PK 状态后，另一个主播会收到 `TRTCLiveRoomDelegate` 的 `onQuitRoomPk()` 回调通知。

```
Future<ActionCallback> quitRoomPK();
```

音视频控制相关接口函数

switchCamera

切换前后摄像头。

```
Future<void> switchCamera(boolean isFrontCamera);
```

setMirror

设置是否镜像展示。

```
Future<void> setMirror(boolean isMirror);
```

参数如下表所示：

参数	类型	含义
isMirror	bool	开启/关闭镜像。

muteLocalAudio

静音本地音频。

```
Future<void> muteLocalAudio(boolean mute);
```

参数如下表所示：

参数	类型	含义
mute	boolean	true：开启静音；false：关闭静音。

muteRemoteAudio

静音远端音频。

```
Future<void> muteRemoteAudio(String userId, boolean mute);
```

参数如下表所示：

参数	类型	含义
userId	String	远端的用户 ID。
mute	boolean	true：开启静音；false：关闭静音。

muteAllRemoteAudio

静音所有远端音频。

```
Future<void> muteAllRemoteAudio(boolean mute);
```

参数如下表所示：

参数	类型	含义
mute	boolean	true：开启静音；false：关闭静音。

背景音乐音效相关接口函数

getAudioEffectManager

获取背景音乐音效管理对象 TXAudioEffectManager 。

```
getAudioEffectManager();
```

美颜滤镜相关接口函数

getBeautyManager

获取美颜管理对象 TXBeautyManager 。

```
getBeautyManager();
```

通过美颜管理，您可以使用以下功能：

- 设置“美颜风格”、“美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小鼻”、“亮眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。
- 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”。
- 设置人脸挂件（素材）等动态效果。
- 添加美妆。
- 进行手势识别。

消息发送相关接口函数

sendRoomTextMsg

在房间中广播文本消息，一般用于弹幕聊天。

```
Future<ActionCallback> sendRoomTextMsg(String message);
```

参数如下表所示：

参数	类型	含义
message	String	文本消息。

sendRoomCustomMsg

发送自定义文本消息。

```
Future<ActionCallback> sendRoomCustomMsg(String cmd, String message);
```

参数如下表所示：

参数	类型	含义
cmd	String	命令字，由开发者自定义，主要用于区分不同消息类型。
message	String	文本消息。

TRTCLiveRoomDelegate事件回调

通用事件回调

onError

错误回调。

> **说明**
> SDK 不可恢复的错误，一定要监听，并分情况给用户适当的界面提示。

参数如下表所示：

参数	类型	含义
errCode	int	错误码。
errMsg	String	错误信息。

onWarning

警告回调。
参数如下表所示：

参数	类型	含义
warningCode	int	错误码。
warningMsg	String	警告信息。

onKickedOffline

其他用户登录了同一账号，被移下线。

房间事件回调

onRoomDestroy

房间被销毁的回调。主播退房时，房间内的所有用户都会收到此通知。

onEnterRoom

本地进房。
参数如下表所示：

参数	类型	含义
result	int	result > 0 时为进房耗时（ms），result < 0 时为进房错误码。

onUserVideoAvailable

远端用户是否存在可播放的主路画面（一般用于摄像头）。
参数如下表所示：

参数	类型	含义
userId	String	用户标识。
available	boolean	画面是否开启。

主播和观众进出事件回调

onAnchorEnter

收到新主播进房通知。连麦观众和跨房 PK 主播进房后观众会收到新主播的进房事件，您可以调用 `TRTCLiveRoom` 的 `startPlay()` 显示该主播的视频画面。
参数如下表所示：

参数	类型	含义
userId	String	新进房主播 ID。
userName	String	用户昵称。
userAvatar	String	用户头像地址。

onAnchorExit

收到主播退房通知。房间内的主播（和连麦中的观众）会收到新主播的退房事件，您可以调用 `TRTCLiveRoom` 的 `stopPlay()` 关闭该主播的视频画面。
参数如下表所示：

参数	类型	含义
userId	String	退出主播 ID。
userName	String	用户昵称。

userAvatar	String	用户头像地址。

onAudienceEnter

收到观众进房通知。

```
void onAudienceEnter(TRTCLiveRoomDef.TRTCLiveUserInfo userInfo);
```

参数如下表所示：

参数	类型	含义
userInfo	TRTCLiveRoomDef.TRTCLiveUserInfo	进房观众用户 ID、昵称、头像等信息。

onAudienceExit

收到观众退房通知。
参数如下表所示：

参数	类型	含义
userId	String	退房观众信息。
userName	String	用户昵称。
userAvatar	String	用户头像地址。

onRequestJoinAnchor

主播收到观众连麦请求时的回调。
参数如下表所示：

参数	类型	含义
userId	String	请求连麦用户 ID。
userName	String	用户昵称。
userAvatar	String	用户头像地址。

onAnchorAccepted

主播同意观众的连麦请求。
参数如下表所示：

参数	类型	含义
userId	String	主播的用户 ID。

onAnchorRejected

主播拒绝观众的连麦请求。
参数如下表所示：

参数	类型	含义
userId	String	主播的用户 ID。

onKickoutJoinAnchor

连麦观众收到被移出连麦的通知。连麦观众收到被主播移除连麦的消息，您需要调用 `TRTCLiveRoom` 的 `stopPublish()` 退出连麦。

主播 PK 事件回调

onRequestRoomPK

收到请求跨房 PK 通知。主播收到其他房间主播的 PK 请求，如果同意 PK ，您需要等待 `TRTCLiveRoomDelegate` 的 `onAnchorEnter()` 通知，然后调用 `startPlay()` 来播放邀约主播的流。
参数如下表所示：

参数	类型	含义
userId	String	请求跨房用户 ID。
userName	String	用户昵称。
userAvatar	String	用户头像地址。

onRoomPKAccepted

主播接受跨房 PK 请求。
参数如下表所示：

参数	类型	含义
userId	String	接收跨房 PK 的用户 ID。

onRoomPKRejected

主播接受跨房 PK 请求。
参数如下表所示：

参数	类型	含义
userId	String	拒绝跨房 PK 的用户 ID。

onQuitRoomPK

收到断开跨房 PK 通知。

消息事件回调

onRecvRoomTextMsg

收到文本消息。
参数如下表所示：

参数	类型	含义
message	String	文本消息。

onRecvRoomCustomMsg

收到自定义消息。
参数如下表所示：

参数	类型	含义
command	String	命令字，由开发者自定义，主要用于区分不同消息类型。
message	String	文本消息。