Tencent Real-Time Communication Demoの構築

製品ドキュメント



©2013-2019 Tencent Cloud. All rights reserved.

Copyright Notice

©2013-2019 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice

Tencent Cloud

All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

カタログ：

Demoの構築
Web インタラクティブライブストリーミング
TWebLive ILVBコンポーネント
ビデオ通話
ビデオ通話(iOS)
ビデオ通話(Android)
ビデオ通話(Web)
ビデオ通話(Flutter)
TUICalling API(iOS)
TUICalling API(Android)
TRTCCalling API(Web)
TRTCCalling API(Flutter)
音声通話
音声通話(iOS)
音声通話(Android)
音声通話(Web)
音声通話(Flutter)
TUICalling API(iOS)
TUICalling API(Android)
TRTCCalling API(Web)
TRTCCalling API(Flutter)
ビデオインタラクティブライブストリーミング
ビデオ・インタラクティブストリーミング(iOS)
ビデオ・インタラクティブストリーミング(Android)
ビデオ・インタラクティブストリーミング(Flutter)
TRTCLiveRoom API (iOS)
TRTCLiveRoom API (Android)
TRTCLiveRoom API (Flutter)
音声チャットルーム
ボイスチャットルーム(iOS)
ボイスチャットルーム(Android)
TRTCVoiceRoom (iOS)
TRTCVoiceRoom (Android)
多人数ビデオミーティング
多人数ビデオミーティング(iOS)

©2013-2019 Tencent Cloud. All rights reserved.

- 多人数ビデオミーティング(Android)
- 多人数ビデオミーティング(Flutter)
- 多人数オーディオビデオルーム(Windows&Mac)
- TUIRoom(iOS)
- TUIRoom(Android)
- TRTCMeeting(Flutter)
- TUIRoom(Windows&Mac)

ボイスサロン

- ボイスサロン(Android)
- ボイスサロン(Flutter)
- TRTCChatSalon(iOS)
- TRTCChatSalon(Flutter)

リアルタイム双方向対話型授業

- リアルタイムインタラクティブ授業(Electron)
- trtc-electron-education API

オンラインカラオケ_KTV

- オンラインカラオケ_KTV（iOS）
- オンラインカラオケ_KTV（Android）
- TRTCKaraoke(iOS)
- TRTCKaraoke(Android)

©2013-2019 Tencent Cloud. All rights reserved.

Demoの構築
Web インタラクティブライブストリーミング
TWebLive ILVBコンポーネント

最終更新日：：2022-04-02 17:02:04

ここでは、UIを有するWeb端末ライブブロードキャストインタラクションコンポーネント、TUIPusher & TUIPlayerをご紹介します。TUIPusher & TUIPlayerはTRTC、IMなどの基本SDKを統合し、企業によるライブブロードキャスト、eコマースマーケティング、業界研修、リモート教育などの様々なライブブロードキャストシーン向けに、すぐにリリースできるWeb端末ライブブロードキャストプッシュプルストリーミングツールによるソリューションをご提供します。

TUIPusher & TUIPlayerのメリット：

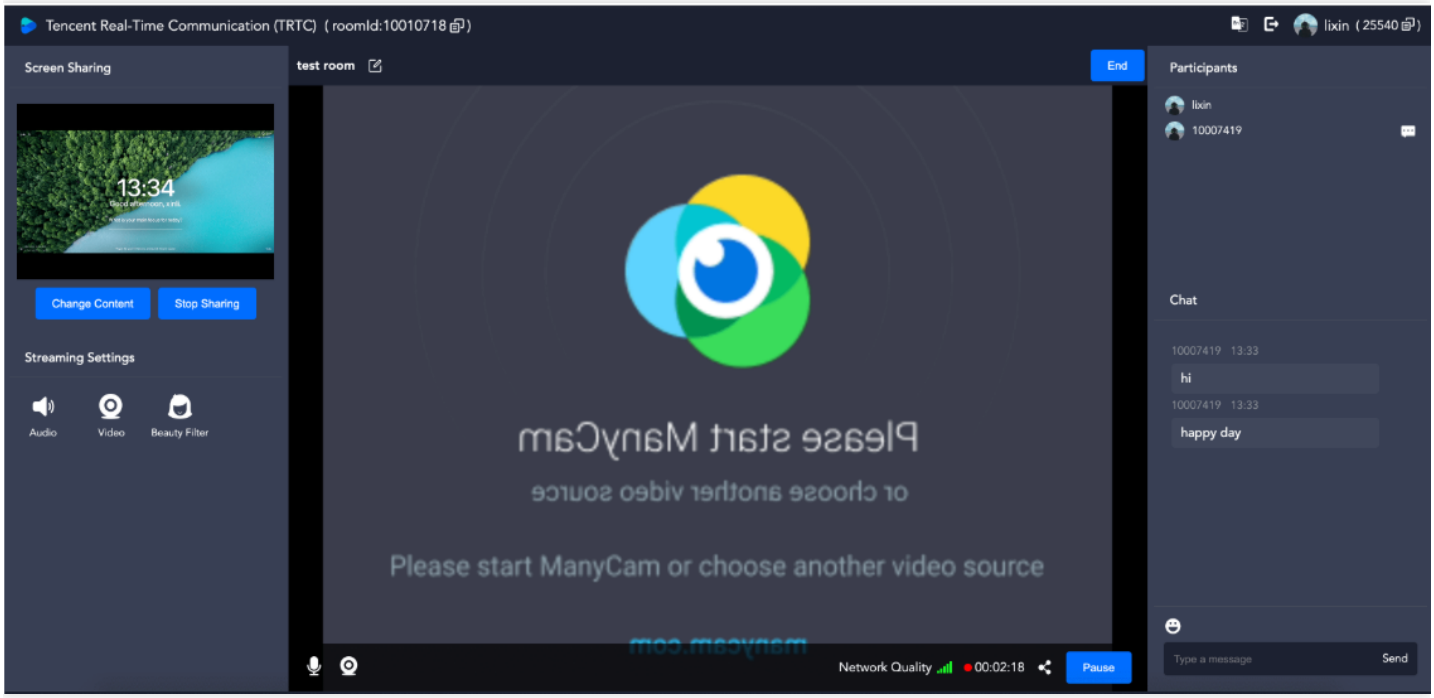
- ライブブロードキャストシーンのニーズに合わせ、UI付きのライブブロードキャストシーン向け汎用ソリューションをご提供します。ライブブロードキャストシーンの一般的な機能（デバイス選択、美顔、CSSプッシュ、視聴者によるプル、チャットなど）をすべて網羅し、ビジネスの速やかなリリースを支援します。
- Tencent Cloud Real-Time Communication（TRTC）、Tencent Cloud Instant Messaging（IM）、およびTencent Cloud Super Player（TCPlayer） などの基本SDKに直接接続し、お客様のビジネス機能のフレキシブルな拡張に役立ちます。
- Web端末はユーザーにとって使いやすく、機能のイテレーションも行いやすいというメリットがあります。

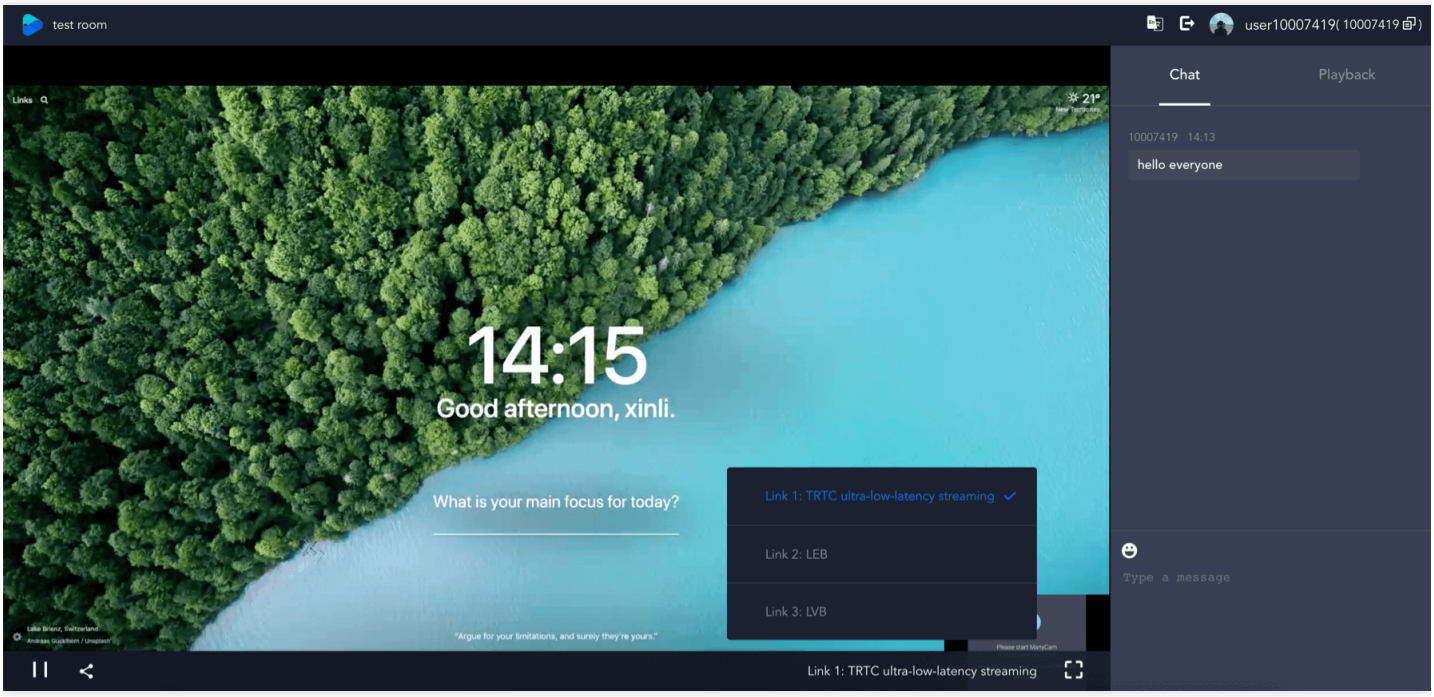
クイック体験

TUIPusherとTUIPlayerの機能デモンストレーションについては下図をご参照ください。また、TUIPusher & TUIPlayerの機能をすぐに体験していただけるよう、ユーザー管理システムとルーム管理システムを組み合わせたTUIPusher体験リンクおよびTUIPlayer体験リンクをご提供しています。

> 注意：
> TUIPusherとTUIPlayerを同時に体験するには、異なるアカウントを使用してログインする必要があります。

©2013-2019 Tencent Cloud. All rights reserved.





コードのダウンロード

Web端末ライブブロードキャストインタラクションコンポーネント（TUIPusher & TUIPlayer）のダウンロード方法は次のとおりです。

- TUIPusher

©2013-2019 Tencent Cloud. All rights reserved.

- TUIPlayer

機能の説明

TUIPusherプッシュコンポーネント

- カメラとマイクからのストリームの収集とプッシュをサポート
  - 必要に応じてビデオパラメータ（フレームレート、解像度、ビットレート）を設定
  - 美顔の起動とビデオ美顔パラメータの設定をサポート
- 画面共有ストリームの収集とプッシュをサポート
- Tencent Real-Time Communicationバックエンドへのプッシュ、Tencent Cloud CDNへのプッシュをサポート
- オンラインチャットルームをサポートし、オンライン視聴者とのインタラクティブなチャットを実現
- 視聴者リストの取得、オンライン視聴者のミュート操作の実行をサポート

TUIPlayerプルコンポーネント

- オーディオビデオストリーミングと画面共有ストリーミングの同時再生をサポート
- オンラインチャットルームをサポートし、キャスターおよびその他視聴者のインタラクティブなチャットを実現
- 超低遅延ライブブロードキャスト（300msの遅延）、ライブイベントストリーミング（1000ms以下の遅延）、標準ライブブロードキャスト（超高速同時視聴をサポート）の3種類のプル回線をサポート
- デスクトップブラウザおよびモバイルブラウザの互換性があり、モバイルブラウザの横画面表示が可能

> 注意：
>
> 一部のブラウザはWebRTCをサポートせず、標準のライブブロードキャスト回線でしか視聴できません。他の回線を体験する場合はブラウザを変更してみてください。

アクセス方法

注意事項

- TUIPusher&TUIPlayerは、Tencent Cloud TRTCとInstant Messagingサービスをベースに開発されています。アカウントと認証を再利用するためには、TRTCアプリケーションとIMアプリケーションのSDKAppIDが一致している必要があります。

©2013-2019 Tencent Cloud. All rights reserved.

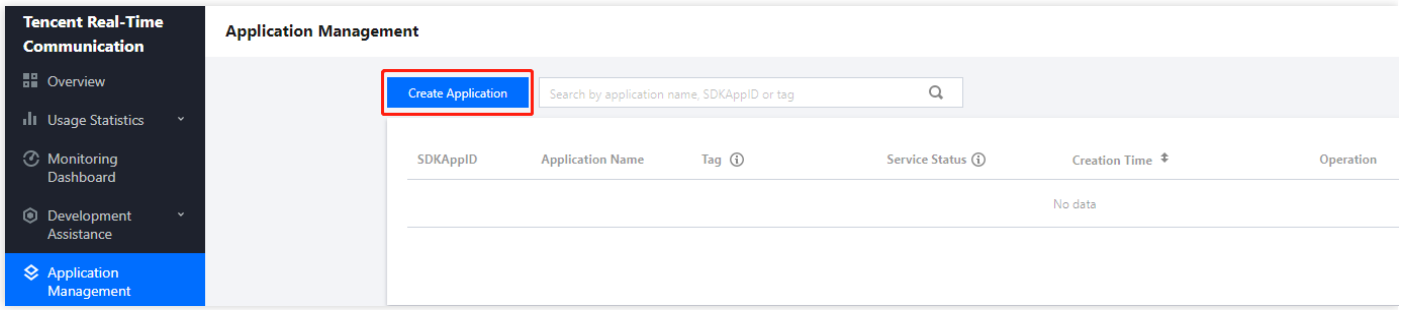
- IMアプリケーションは、テキストメッセージに対し、ベーシック版のセキュリティ対策機能を提供します。不適切な単語のカスタム機能を使用したい場合は、**アップグレード**をクリックします。
- UserSigをローカルで計算する方式は、ローカルでの開発とデバッグのみに使用されます。SECRETKEYが漏洩すると、攻撃者がTencent Cloudトラフィックを盗用する可能性があるので、オンラインで直接発行しないでください。UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。 UserSigが必要なときは、Appからサービスサーバーにリクエストを発出しダイナミックUserSigを取得します。詳細はサーバーでのUserSig新規作成をご参照ください。

ステップ1：Tencent Cloudサービスのアクティブ化

- 方法1：TRTCの場合
- 方法2：インスタントメッセージ IMの場合

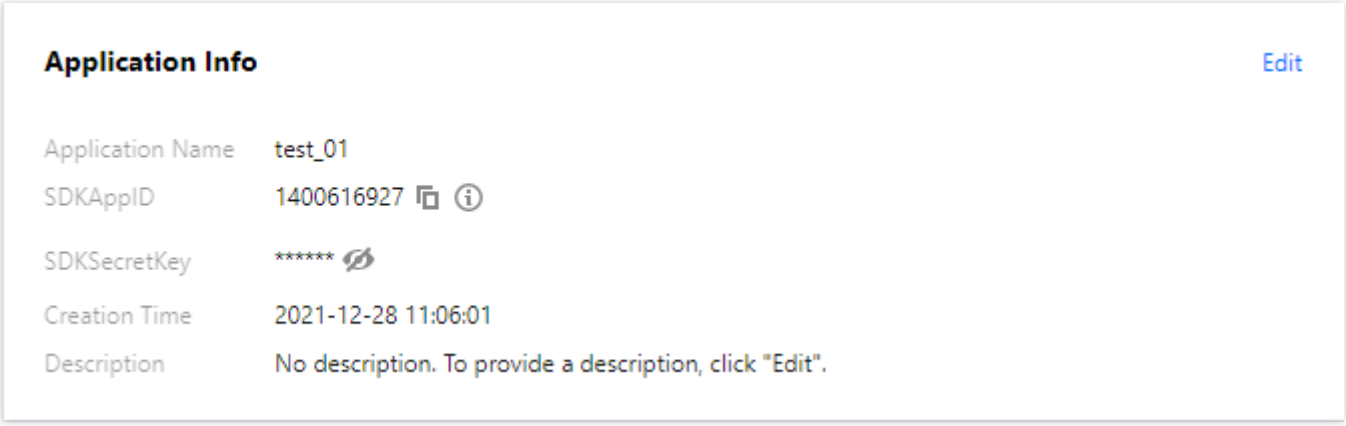
手順1：TRTCアプリケーションの作成

1. Tencent Cloudアカウントの登録を行い、Tencent Real-Time Communication（TRTC）とInstant Messagingサービスをアクティブ化します。
2. TRTCコンソール で、**アプリケーション管理>アプリケーションの作成** をクリックし、新たなアプリケーションを作成します。

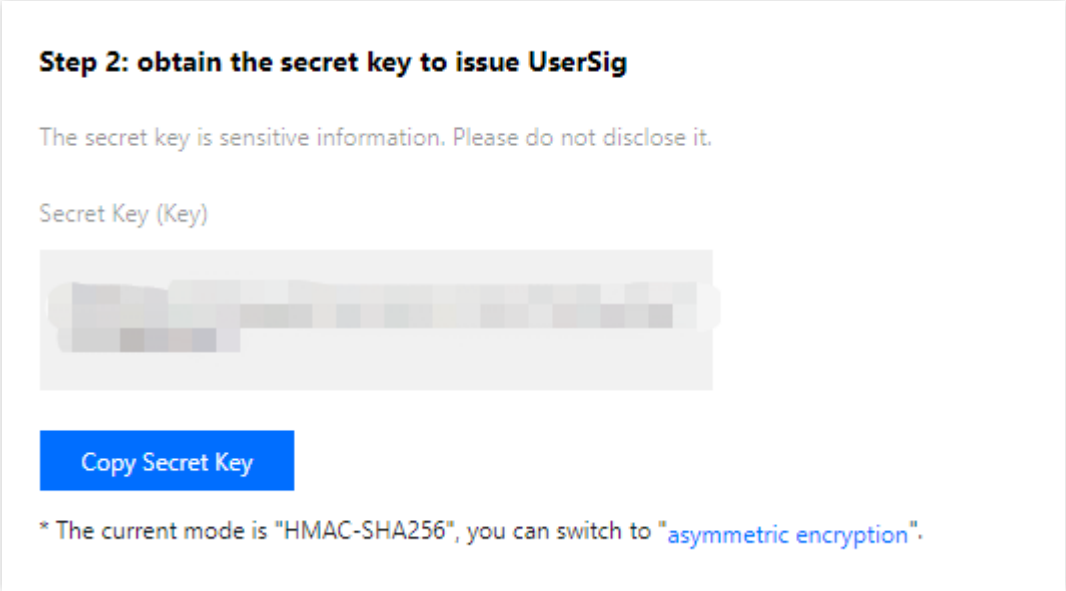


ステップ2： TRTCキー情報の取得

1. **アプリケーション管理>アプリケーション情報**でSDKAppID情報を取得します。



2. **アプリケーション管理>クイックマスター**でアプリケーションのsecretKey情報を取得します。



説明

- TRTCアプリケーションを初めて作成するTencent Cloudアカウントは、10000分間のオーディオビデオリソース無料トライアルパッケージを受け取ることができます。
- TRTCプリケーションを作成すると、同じSDKAppIDのIMアプリケーションが自動的に作成され、IMコンソールでこのアプリケーションのパッケージ情報を設定することができます。

ステップ2：プロジェクトの準備

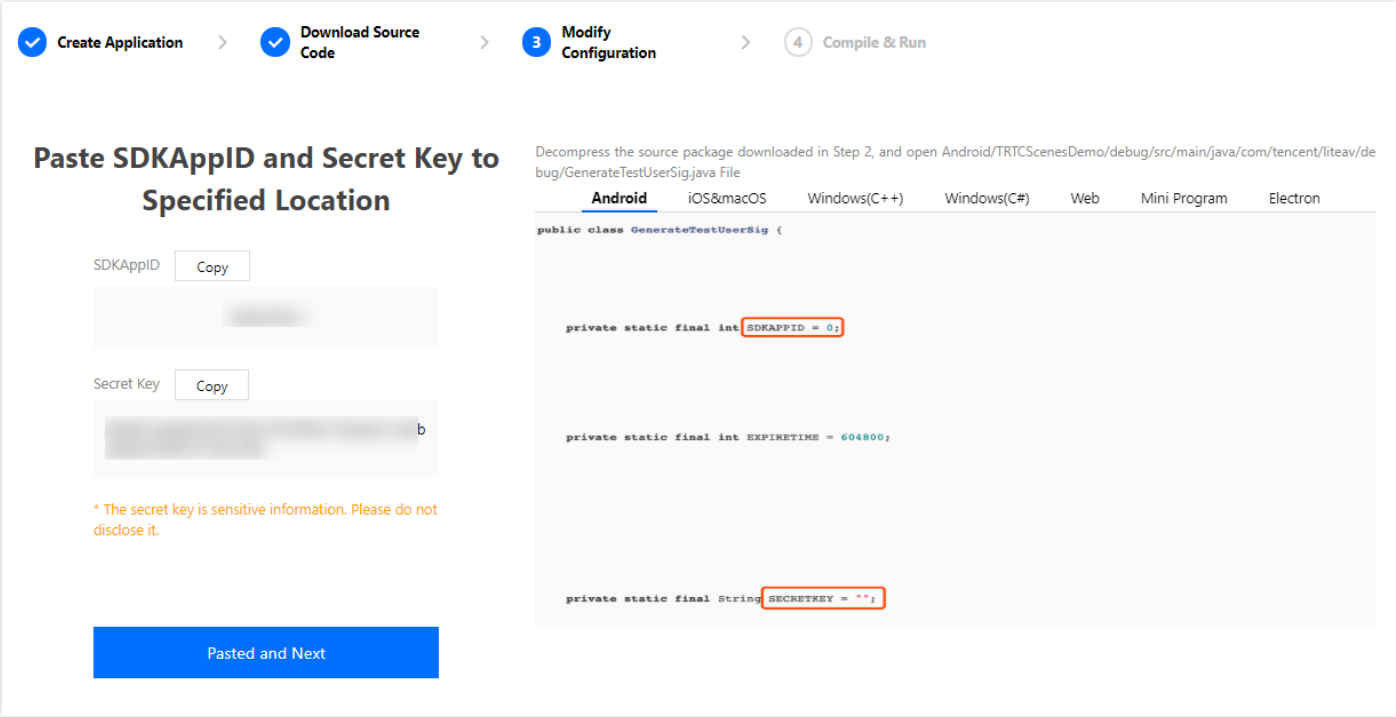
1. GitHubでTUIPusher & TUIPlayerコードをダウンロードします。

©2013-2019 Tencent Cloud. All rights reserved.

2. TUIPusher & TUIPlayerのためにインストールして依存します。

```
cd Web/TUIPusher
npm install
cd Web/TUIPlayer
npm install
```

3. sdkAppIdとsecretKeyを `TUIPusher/src/config/basic-info-config.js` 、 `TUIPlayer/src/config/basic-info-config.js` 設定ファイルに入力します。



4. ローカル開発環境でTUIPusher&TUIPlayerを実行します。

```
cd Web/TUIPusher
npm run serve
cd Web/TUIPlayer
npm run serve
```

5. `http://localhost:8080` と `http://localhost:8081` を開き、TUIPusherとTUIPlayer機能を体験することができます。

6. `TUIPusher/src/config/basic-info-config.js` 、 `TUIPlayer/src/config/basic-info-config.js` 設定ファイル中のルーム，キャスターおよび視聴者などの情報を変更することができます。**TUIPusherとTUIPlayerの**

©2013-2019 Tencent Cloud. All rights reserved.

ルーム情報、キャスター情報を一致させてください。

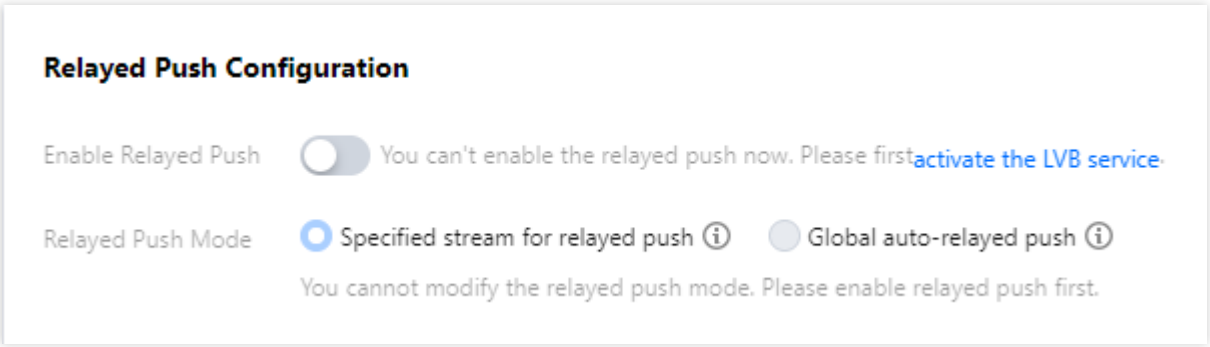
注意：

- 上記設定が完了すると、TUIPusher & TUIPlayerを使用して超低遅延ライブブロードキャストを実行することができます。ライブイベントストリーミングと標準ライブブロードキャストをサポートする必要がある場合は、引き続きステップ3：Relayed Live Streamingをご参照ください。
- UserSigをローカルで計算する方法は、ローカルの開発とデバッグにのみ使用されます。`SECRETKEY`が漏えいすると、攻撃者がTencent Cloudのトラフィックを盗用する可能性があるので、ネット上にそのままリリースしないでください。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appからサービスサーバーにリクエストを送信し動的にUserSigを取得します。詳細はサーバーでのUserSig新規作成をご参照ください。

ステップ3：Relayed live streaming

TUIPusher&TUIPlayerによって実装されるライブイベントストリーミングと標準ライブブロードキャストはTencent Cloud[CSSサービス]（https://intl.cloud.tencent.com/document/product/267）に依存しているため、ライブイベントストリーミングと標準ライブブロードキャスト回線をサポートするには、Relayed Push機能を有効にする必要があります。

1. **TRTCコンソール** で現在使用中のアプリケーションのRelayed Push設定を有効にします。必要に応じてRelayed Push用指定ストリームまたはGlobal Auto-relayを起動することができます。



2. **ドメイン名管理**ページで独自の再生ドメイン名を追加します。具体的な内容は独自のドメイン名の追加をご参照ください。
3. `TUIPlayer/src/config/basic-info-config.js` 設定ファイルで再生ドメイン名を設定します。

上記設定が完了すると、TUIPusher & TUIPlayerがサポートする超低遅延ライブブロードキャスト、ライブイベントストリーミング、標準ライブブロードキャストの全機能を体験できます。

©2013-2019 Tencent Cloud. All rights reserved.

ステップ4：本番環境アプリケーション

TUIPusher & TUIPlayerを本番環境アプリケーションに使用する場合は、TUIPusher & TUIPlayerにアクセスすることに加え、次の手順を実行する必要があります。

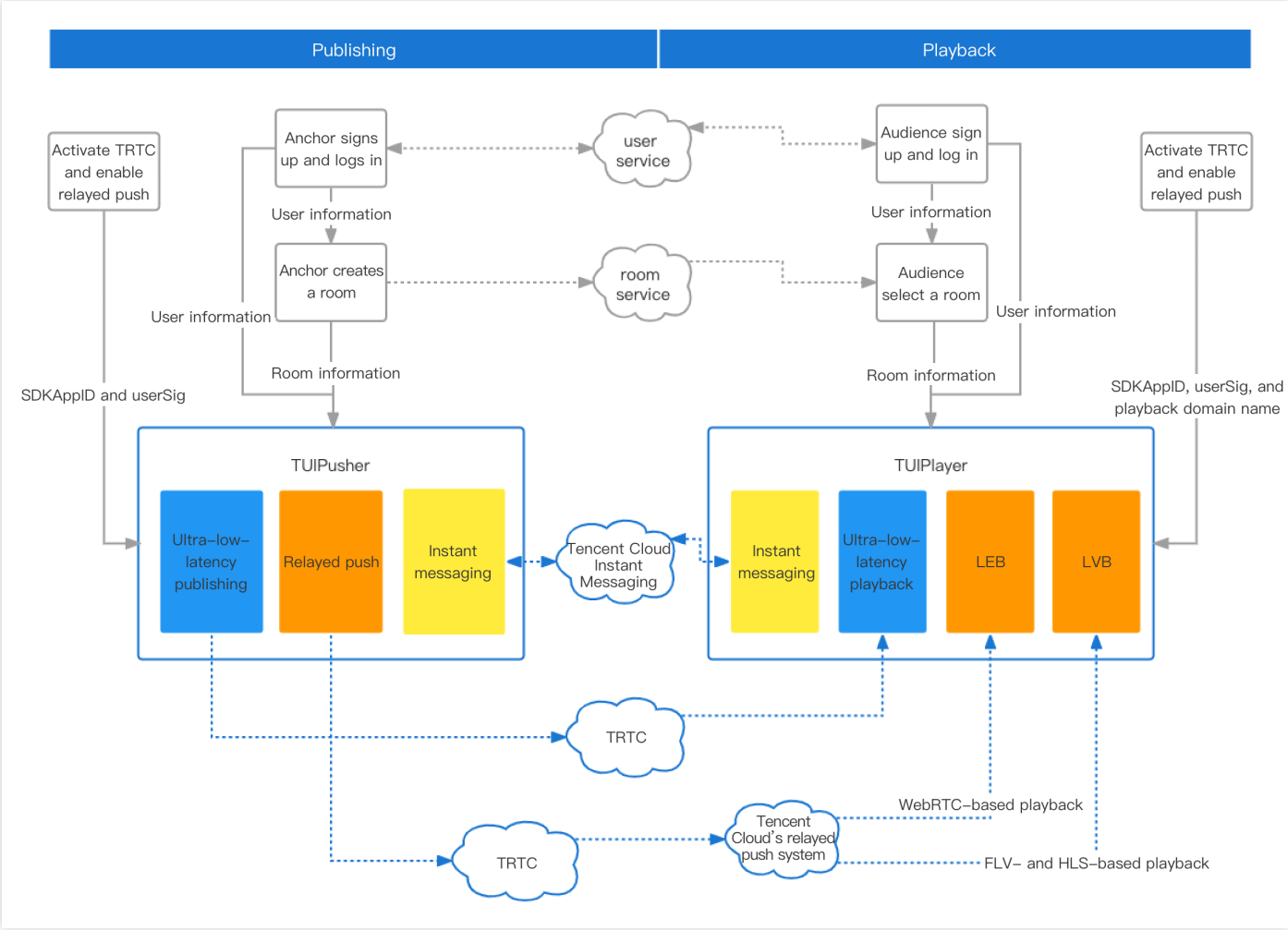
- ユーザーID、ユーザー名、ユーザープロフィール画像などを含む製品ユーザー情報を管理するためのユーザー管理システムを作成します。
- ライブブロードキャストルームID、ライブブロードキャストルーム名、ライブブロードキャストルームのホスト情報などを含む製品のライブブロードキャストルーム情報を管理するためのルーム管理システムを作成します。
- サーバーがUserSigを発行します。

> 注意：
>
> - ここでのUserSigの発行方法は、クライアントが入力したsdkAppIdとsecretKeyに基づきuserSigを発行する方法です。この方法で発行されるsecretKeyは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この方法はローカルのTUIPusher & TUIPlayerクイックスタート機能デバッグの実行のみに適しています**。
> - UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、アプリケーションからサービスサーバーにリクエストを送信し動的にUserSigを取得します。詳細はサーバーでのUserSig新規作成をご参照ください。

- `TUIPusher/src/pusher.vue` 、 `TUIPlayer/src/player.vue` ファイルを参照し、ユーザー情報、ライブブロードキャストルーム情報、SDKAppId、UserSigなどのアカウント情報をvuexのstoreに送信し、グローバルストレージを実行すると、プッシュプルストリーミングする2つのクライアントの全機能をクイックスタートするこ

©2013-2019 Tencent Cloud. All rights reserved.

とができます。詳細なサービスフローについては下図をご参照ください。



関連する質問

Web端末での美顔機能の実現方法。

美顔を有効にするをご参照ください。

Web端末での画面共有の実現方法。

画面共有をご参照ください。

Web端末でのクラウドレコーディングの実現方法。

1. **クラウドレコーディング**機能を有効にするための具体的な操作については、クラウドレコーディングと再生の実現をご参照ください 。
2. **クラウドレコーディング> 指定ユーザーレコーディング**を有効にすると、Web端末がTRTC.createClientインターフェース呼び出し時にuserDefineRecordIdパラメータを渡すことによってレコーディングを開始できま

©2013-2019 Tencent Cloud. All rights reserved.

す。

Web端末でのCDNへのプッシュの実現方法。

Web端末でのCDNへのプッシュについては、CDNへのプッシュの実現 をご参照ください。

#Web端末でのライブイベントストリーミングプルの実現方法。

ライブイベントストリーミングプルの実現方法は、Web SDKを介してCDNにプッシュした後、WebRTCプロトコルを使用してプルします。

注意事項

プラットフォームのサポート

オペレーションシステム（OS）	ブラウザタイプ	ブラウザ最低バージョン要件	TUIPlayer	TUIPusher
Mac OS	デスクトップ版Safariブラウザ	11+	サポートあり	サポートあり
Mac OS	デスクトップ版Chromeブラウザ	56+	サポートあり	サポートあり
Mac OS	デスクトップ版Firefoxブラウザ	56+	サポートあり	サポートあり
Mac OS	デスクトップ版Edgeブラウザ	80+	サポートあり	サポートあり
Mac OS	デスクトップ版WeChat Embedded Web	-	サポートあり	サポートなし
Mac OS	デスクトップ版WeCom Embedded Web	-	サポートあり	サポートなし

©2013-2019 Tencent Cloud. All rights reserved.

オペレーションシステム（OS）	ブラウザタイプ	ブラウザ最低バージョン要件	TUIPlayer	TUIPusher
Windows	デスクトップ版Chromeブラウザ	56+	サポートあり	サポートあり
Windows	デスクトップ版QQブラウザ（クイックコア）	10.4+	サポートあり	サポートあり
Windows	デスクトップ版Firefoxブラウザ	56+	サポートあり	サポートあり
Windows	デスクトップ版Edgeブラウザ	80+	サポートあり	サポートあり
Windows	デスクトップ版WeChat Embedded Web	-	サポートあり	サポートなし
Windows	デスクトップ版WeCom Embedded Web	-	サポートあり	サポートなし
iOS	WeChat内蔵ブラウザ	-	サポートあり	サポートなし
iOS	WeChat Work内蔵ブラウザ	-	サポートあり	サポートなし
iOS	モバイル版Safariブラウザ	-	サポートあり	サポートなし
iOS	モバイル版Chromeブラウザ	-	サポートあり	サポートなし
Android	WeChat内蔵ブラウザ	-	サポートあり	サポートなし

©2013-2019 Tencent Cloud. All rights reserved.

オペレーションシステム（OS）	ブラウザタイプ	ブラウザ最低バージョン要件	TUIPlayer	TUIPusher
Android	WeComブラウザ	-	サポートあり	サポートなし
Android	モバイル版Chromeブラウザ	-	サポートあり	サポートなし
Android	モバイル版QQブラウザ	-	サポートあり	サポートなし
Android	モバイル版Firefoxブラウザ	-	サポートあり	サポートなし
Android	モバイル端末UCブラウザ	-	サポートあり（標準ライブブロードキャスト視聴のみ）	サポートなし

ドメイン名要件

ユーザーのセキュリティ、プライバシーなどの問題を考慮し、ブラウザはHTTPSプロトコルでしかTUIPusher & TUIPlayerの全機能を正常に使用できないようにウェブページを制限しています。本番環境のユーザーがTUIPusher & TUIPlayerの全機能にスムーズにアクセスし体験できるようにするには、HTTPSプロトコルを使用してオーディオビデオアプリケーションページにアクセスしてください。

> 注意：
> ローカル開発には、`http://localhost` プロトコルを使用してアクセスできます。

URLドメイン名およびプロトコルのサポート状況については、以下の表をご参照ください。

ユースケース	プロトコル	TUIPlayer	TUIPusher	TUIPusher画面共有	備考
本番環境	HTTPSプロトコル	サポートあり	サポートあり	サポートあり	推奨

©2013-2019 Tencent Cloud. All rights reserved.

ユースケース	プロトコル	TUIPlayer	TUIPusher	TUIPusher画面共有	備考
本番環境	HTTPプロトコル	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	`http://localhost`	サポートあり	サポートあり	サポートあり	推奨
ローカル開発環境	`http://127.0.0.1`	サポート	サポート	サポート	-
ローカル開発環境	`http://[ローカルマシンIP]`	サポートあり	サポートなし	サポートなし	-

ファイアウォールの制限

TUIPusher & TUIPlayerは次のポートに依存してデータ送信を実行しています。これらのポートをファイアウォールのホワイトリストに追加してください。

- TCPポート：8687
- UDPポート：8000、8080、8800、843、443、16285
- ドメイン名：qcloud.rtc.qq.com

結び

今後のイテレーションにおいて、TRTC Web端末のプッシュプルコンポーネントは徐々にiOS、Andriodなどの各端末との接続が可能になるほか、Web端末上での視聴者のマイク接続、高度な美顔、カスタムレイアウト、複数のプラットフォームへのリツイート、画像・テキスト・音楽などのアップロード機能が実装されます。ぜひご利用いただき、貴重なご意見をお寄せください。

ご要望やフィードバックなどがございましたら、colleenyu@tencent.comまでご連絡ください。

©2013-2019 Tencent Cloud. All rights reserved.

ビデオ通話
ビデオ通話(iOS)

最終更新日：：2022-03-10 11:09:53

デモンストレーション

Appをダウンロードしてインストールすると、TRTC通話の効果を体験できます。

発呼側	着呼側

©2013-2019 Tencent Cloud. All rights reserved.





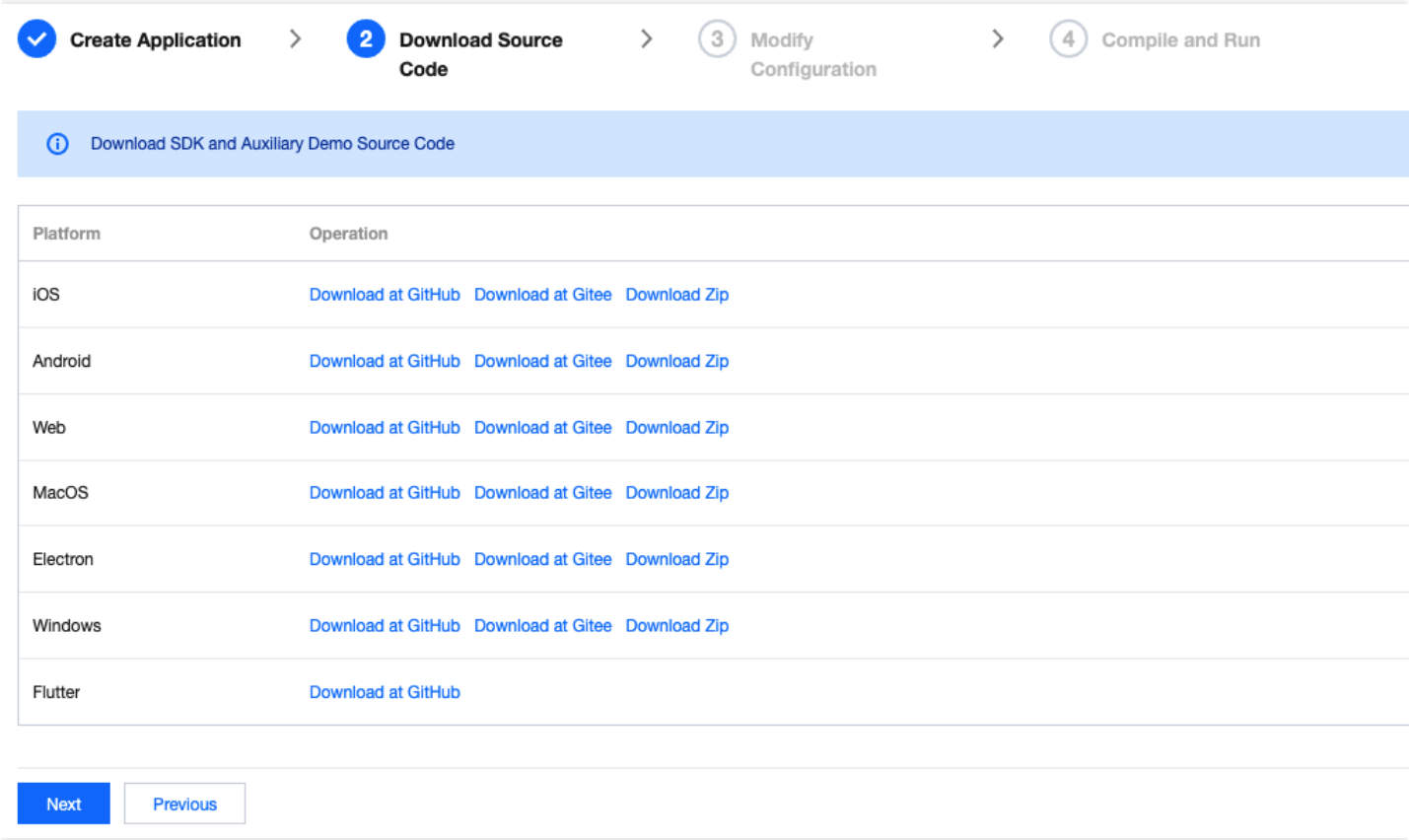
注意：
オーディオビデオ通話機能を素早く実装できるように、当社はTUICallingコンポーネントをリモデルし、通話UIをTUICallingコンポーネントの内部に実装しました。お客様がUIに気をつかう必要はありません。

©2013-2019 Tencent Cloud. All rights reserved.

Appの実行と体験

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援>Demoクイックスタート**を選択します。
   2. **アプリケーション名（例：** `TestVideoCall` **）を入力して、**作成**をクリックします。
2. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUICallingに進み、ソースコードをCloneまたはダウンロードします。

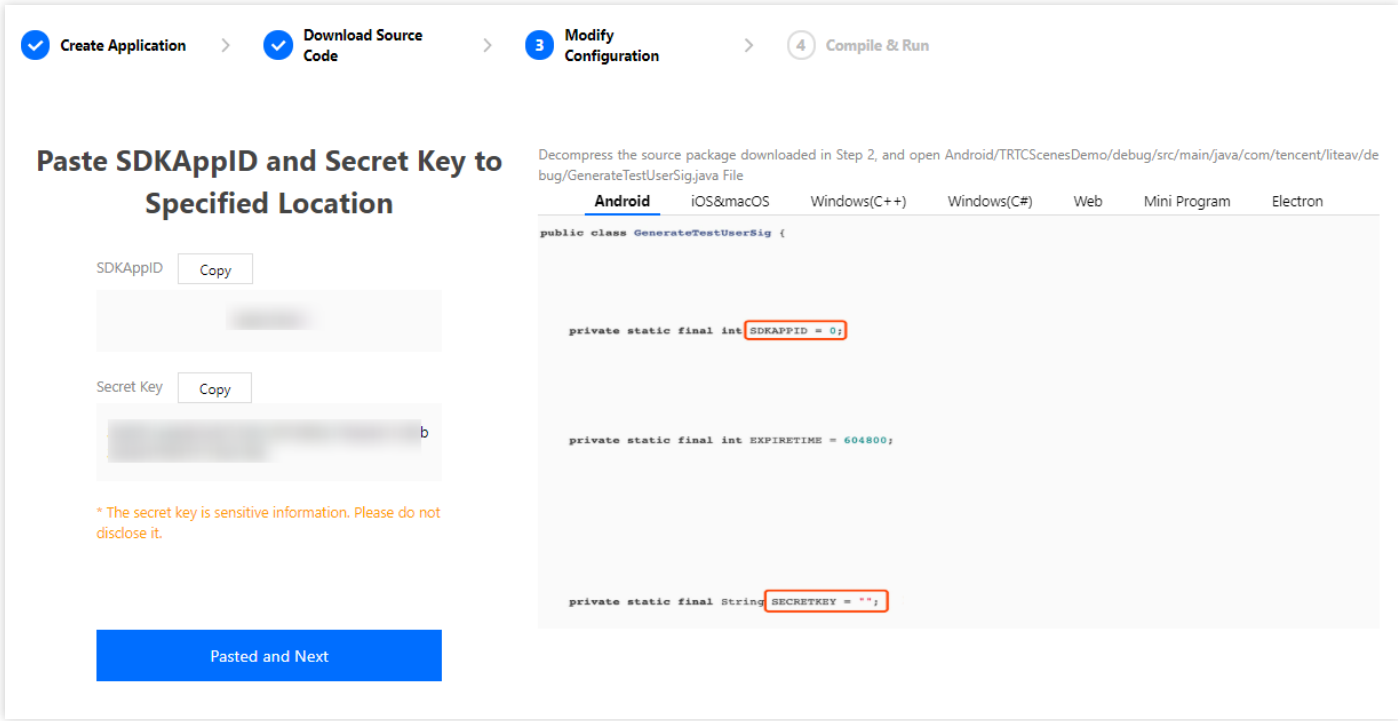
手順3：Appプロジェクトファイルの設定

©2013-2019 Tencent Cloud. All rights reserved.

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `iOS/Example/Debug/GenerateTestUserSig.swift` のファイルを見つけて開きます。

3. `GenerateTestUserSig.swift` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトは0。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトは空文字列。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

©2013-2019 Tencent Cloud. All rights reserved.

手順4：Appの実行

Xcode（バージョン11.0以上）を使用してソースコードプロジェクト `TUICalling/Example/TUICallingApp.xcworkspace` を開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. 電話をかけたいuserIdを入力し、検索をクリックします。
3. **呼出**をクリックし、**ビデオ通話**を選択して電話をかけます（**着呼者がアプリケーション内にいることを保証してください。保証できない場合、電話をかけても失敗する可能性があります**）。

ユーザーB

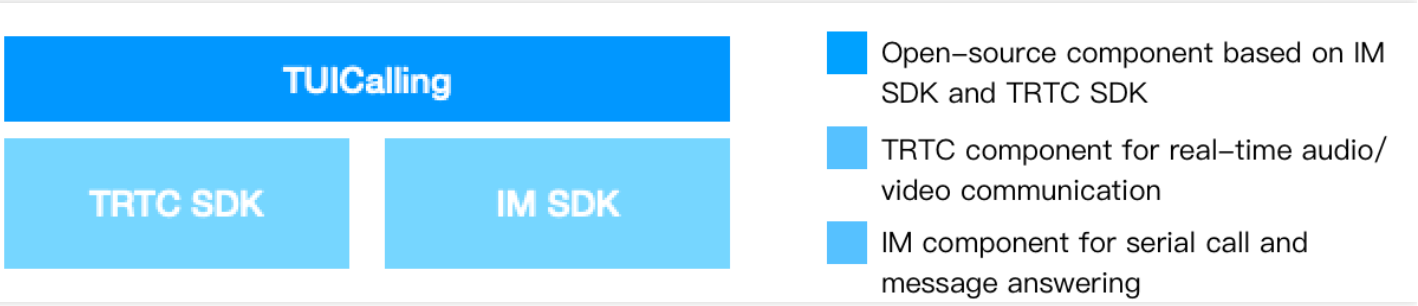
1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. メインページに進み、通話応答を待ちます。

具体的な接続フロー

ソースコードの `Source` フォルダの中にはui、model、Serviceの3つのサブフォルダが含まれています。そのうち、Serviceフォルダには外部に公開しているオープンソースコンポーネントTUICallingManagerが含まれています。 `TUICallingManager.h` ファイルの中でこのコンポーネントが提供するインターフェース関数を見つけるこ

©2013-2019 Tencent Cloud. All rights reserved.

とができます。



オープンソースコンポーネントTUICallingのTUICallingManagerをそのまま使用し、オーディオビデオ通話機能を手軽に実装することができます。もう複雑な通話UIやロジックの実装をご自身で行う必要はありません。

手順1：SDKの統合

通話コンポーネントTRTCCallingは、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

- **方法1：cocoapodsリポジトリを介する依存**

```
pod 'TXIMSDK_iOS'
pod 'TXLiteAVSDK_TRTC'
```

- **方法2：ローカルを介する依存**
  開発環境でのcocoapodsリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定

info.plistファイルに `Privacy - Camera Usage Description` 、 `Privacy - Microphone Usage Description` を追加し、カメラとマイクの権限を申請する必要があります。

手順3：TUICallingコンポーネントをインポート

cocoapodsによってコンポーネントをインポートします。具体的な手順については、以下のとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

1. プロジェクトディレクトリ下の `Source` 、 `Resources` 、 `TXAppBasic` フォルダ、 `TUICalling.podspec` ファイルをプロジェクトディレクトリ下へコピーします。
2. `Podfile` ファイル内に以下の依存関係を追加します。その後、 `pod install` コマンドを実行すると、インポートが完了します。

```
pod 'TXAppBasic', :path =&gt; "../TXAppBasic/"
pod 'TXLiteAVSDK_TRTC'
pod 'TUICalling', :path =&gt; "../", :subspecs =&gt; ["TRTC"]
```

手順4：コンポーネントの初期化およびログイン

1. `TUICallingManager.sharedInstance()` を呼び出し、コンポーネントを初期化します。
2. `TUILogin.initWithSdkAppID(SDKAPPID)` を呼び出し、ログインを初期化します。
3. `TUILogin.login(userId, userSig)` を呼び出し、コンポーネントのログインを完了します。このうちの重要なパラメータの入力については、下表をご参照ください。

パラメータ名	作用
sdkAppID	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。計算方法については UserSigの計算方法をご参照ください。

```
// コンポーネントの初期化
TUICallingManager.sharedInstance();
// ログイン
TUILogin.initWithSdkAppID(SDKAPPID)
TUILogin.login(userId, userSig) {
print("login success")
} fail: { code, errorDes in
print("login failed, code:¥(code), error: ¥(errorDes ?? "nil")")
}
```

手順5：オーディオビデオ通話の実装

1. 発信側：TUICallingManagerの `call();` メソッドを呼び出して通話リクエストを送信し、ユーザーIDの配列（userIDs）と通話タイプ（type）を渡します。通話タイプパラメータは `.audio` （オーディオ通話）または `.video` （ビデオ通話）を渡します。ユーザーID配列（userIDs）のuserIDが1つのみの時は、シングル通話と見なされ、ユーザーID配列（userIDs）のuserIDが複数（>=2）の時は多人数通話と見なされます。

©2013-2019 Tencent Cloud. All rights reserved.

2. 受信側：受信側がログイン状態の場合は、対応するインターフェースが自動的に起動します。受信側がログインしていない状態でも通話リクエストを受け取りたい場合は、オフライン応答をご参照ください。

```
//1. リスナーの登録
TUICallingManager.shareInstance().setCallingListener(listener: TUICallingListerner())

// 2. カスタムページ設定の有無（デフォルトはオフ）
TUICallingManager.shareInstance().enableCustomViewRoute(enable: true)

// 3. リスナーコールバックメソッドの実装
public func shouldShowOnCallView() -&gt; Bool {
return true;
}

public func callStart(userIDs: [String], type: TUICallingType, role: TUICallingRole, viewControll
er: UIViewController?) { if let vc = viewController {
callingVC = vc;
vc.modalPresentationStyle = .fullScreen

if var topController = UIApplication.shared.keyWindow?.rootViewController {
while let presentedViewController = topController.presentedViewController {
topController = presentedViewController
}

if let navigationVC = topController as? UINavigationController {
if navigationVC.viewControllers.contains(self) {
present(vc, animated: false, completion: nil)
}else{
navigationVC.popToRootViewController(animated: false)
navigationVC.pushViewController(self, animated: false)
navigationVC.present(vc, animated: false, completion: nil)
}
}else{
topController.present(vc, animated: false, completion: nil)
}
}
}
}

public func callEnd(userIDs: [String], type: TUICallingType, role: TUICallingRole, totalTime: Flo
at) {
callingVC.dismiss(animated: true, completion: nil)
}

public func onCallEvent(event: TUICallingEvent, type: TUICallingType, role: TUICallingRole, messa
ge: String) {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
}
// 4.電話をかけます
TUICallingManager.shareInstance().call(userIDs, .video)
```

手順6：オフライン応答の実装

> 説明：
> ビジネスの位置付けがオンラインカスタマーサービスなどのオフライン応答機能を必要としないシーンである場合は、上記[手順1]（＃model.step1）-[手順5]（＃model.step5）の対応で問題ありません。 しかし、ビジネスの位置付けがソーシャルシーンである場合は、オフライン応答を実装することをお勧めします。

IM SDKはオフラインプッシュをサポートしていますが、使用可能な基準を満たすためには相応の設定を行う必要があります。

1. Appleプッシュ証明書を申請する場合の具体的な操作については、Appleプッシュ証明書の申請をご参照ください。
2. バックエンドおよびクライアントがオフラインプッシュを設定する場合の具体的な操作については、オフラインプッシュ（iOS）をご参照ください。
3. 現在、TRTCCallingImplのsendModelシグナリング送信関数にオフライン送信関数が統合されており、Appのオフラインプッシュを設定した後、メッセージのオフラインプッシュを実装することができます。

コンポーネントAPIリスト

TUICallingコンポーネントのAPIインターフェースリストは次のとおりです。

インターフェース関数	インターフェースの機能
call	C2C通話に招待します
receiveAPNSCalled	被招待者として着信に応答します
setCallingListener	リスナーを設定します
setCallingBell	着信音の設定(30s以内を推奨)
enableMuteMode	ミュートモードをオンにします

©2013-2019 Tencent Cloud. All rights reserved.

インターフェース関数	インターフェースの機能
enableCustomViewRoute	カスタムビューをオンにします。オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewのインスタンスを受信します。開発者自身が表示方式を決定します。注意：フルスクリーンまたはスクリーンと同じ比率で表示する必要があります。そうしない場合、表示に異常が生じます

©2013-2019 Tencent Cloud. All rights reserved.

ビデオ通話(Android)

最終更新日：：2022-03-09 17:44:04

デモンストレーション

Appをダウンロードしてインストールすると、TRTC通話の効果を体験できます。

発呼側	着呼側

©2013-2019 Tencent Cloud. All rights reserved.





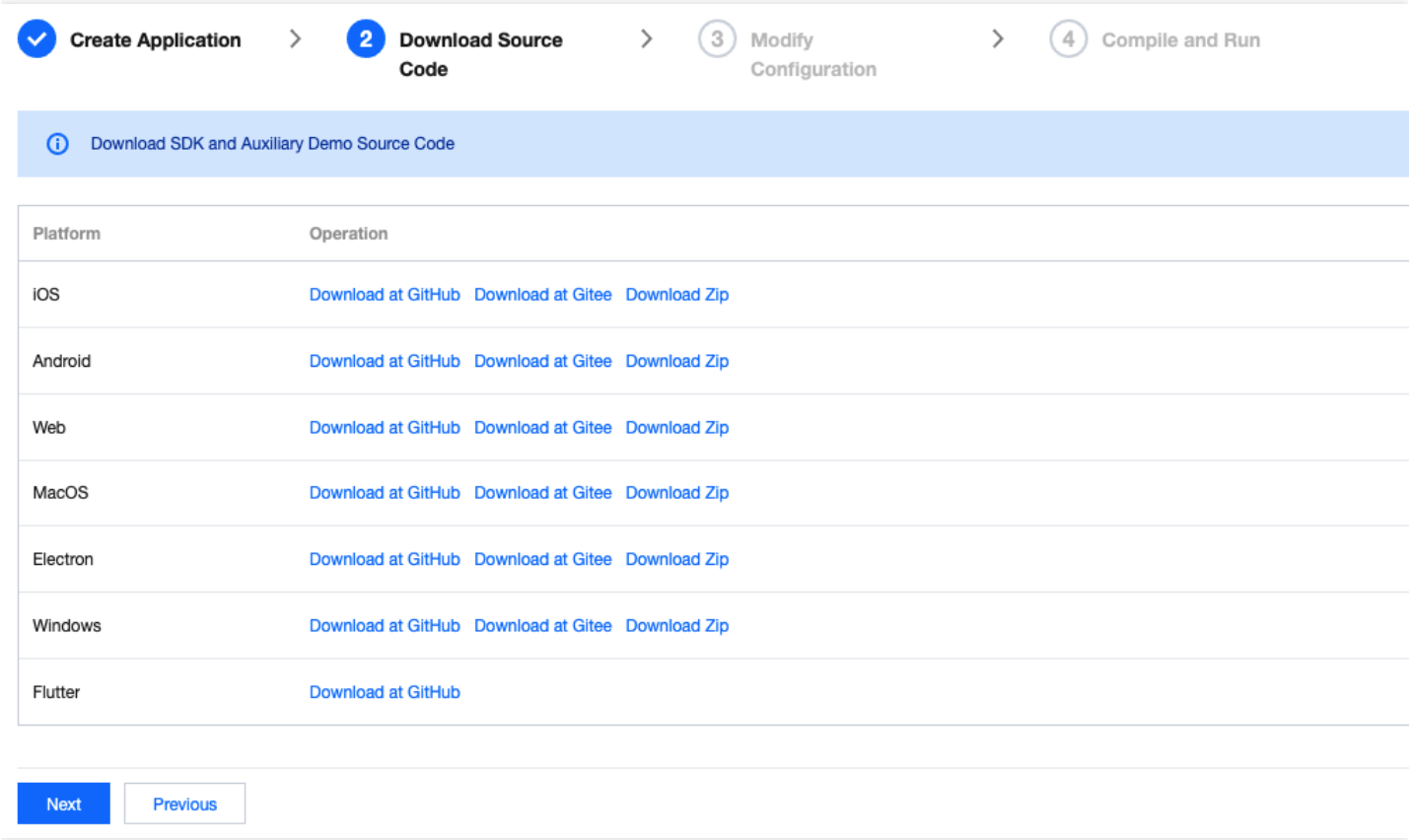
注意：
オーディオビデオ通話機能を素早く実装できるように、当社はTUICallingコンポーネントをリモデルし、通話UIをTUICallingコンポーネントの内部に実装しました。お客様がUIに気をつかう必要はありません。

©2013-2019 Tencent Cloud. All rights reserved.

Appの実行と体験

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援>Demoクイックスタート**を選択します。
   2. **アプリケーション名（例：** `TestVideoCall` **）を入力して、**作成**をクリックします。
2. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUICallingに進み、ソースコードをCloneまたはダウンロードします。

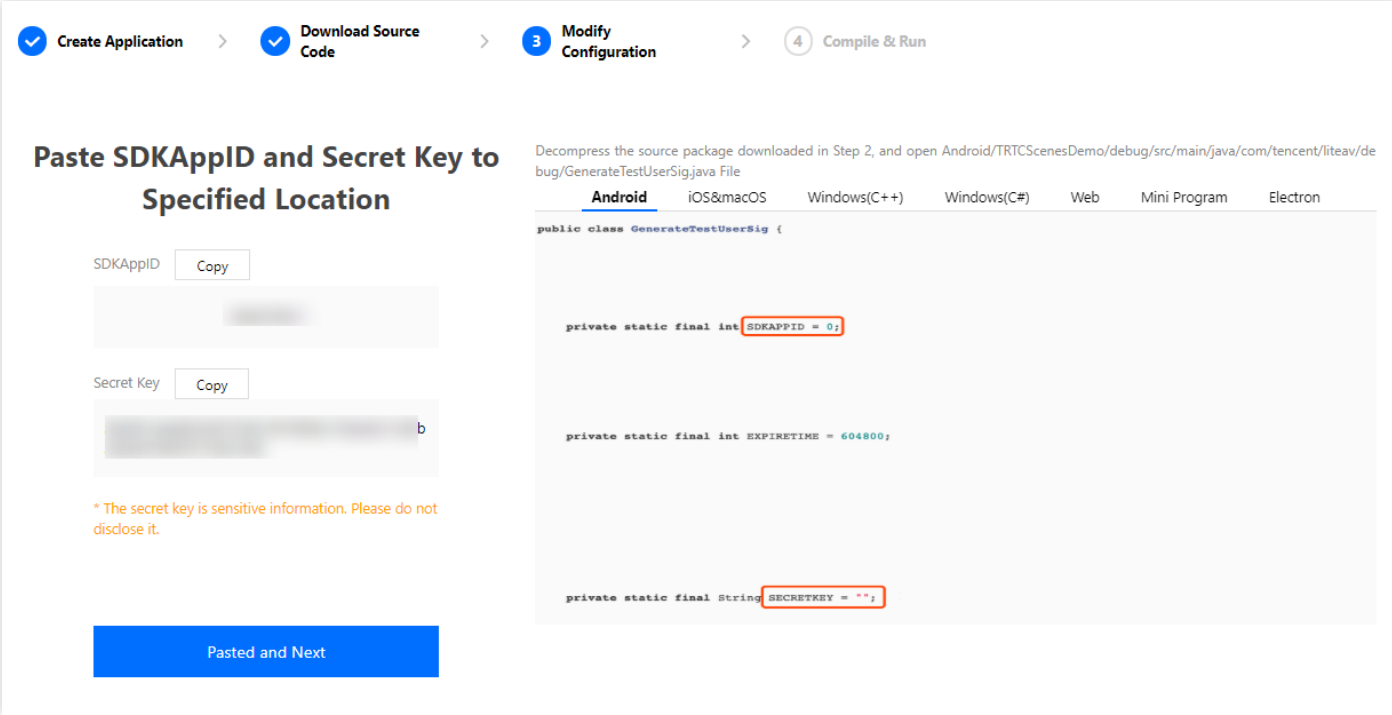
手順3：Appプロジェクトファイルの設定

©2013-2019 Tencent Cloud. All rights reserved.

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `Android/Debug/src/main/java/com/tencent/liteav/debug/GenerateTestUserSig.java` のファイルを見つけて開きます。

3. `GenerateTestUserSig.java` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはプレースホルダー(PLACEHOLDER)。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトはプレースホルダー(PLACEHOLDER)。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを

©2013-2019 Tencent Cloud. All rights reserved.

取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Android Studio（バージョン3.5以降）を使用してソースコードプロジェクト `TUICalling` を開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

体験アプリケーション

注意：
体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. 電話をかけたいuserIdを入力し、検索をクリックします。
3. **呼出**をクリックし、**ビデオ通話**を選択して電話をかけます（**着呼者がアプリケーション内にいることを保証してください。保証できない場合、電話をかけても失敗する可能性があります**）。

ユーザーB

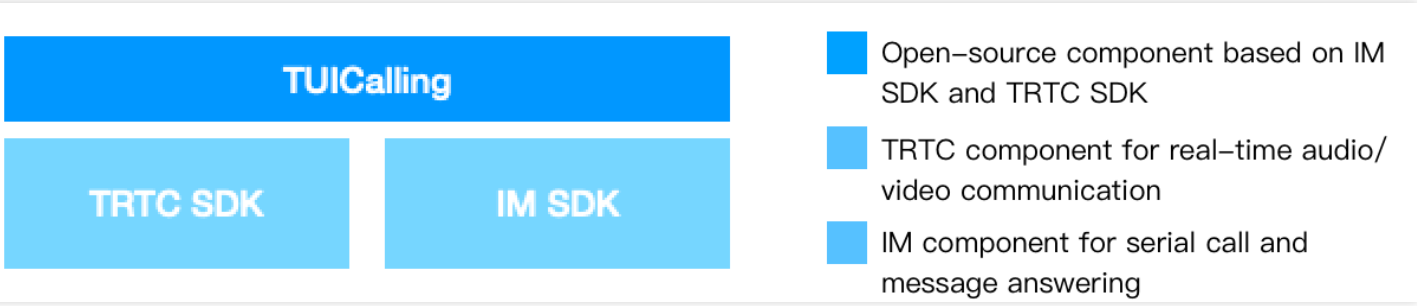
1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. メインページに進み、通話応答を待ちます。

具体的な接続フロー

ソースコードの `Source` フォルダの中にはuiとmodelの2つのサブフォルダが含まれています。そのうち、modelフォルダには外部に公開しているオープンソースコンポーネントTUICallingManagerが含まれています。 `TUICalling.java` ファイルの中でこのコンポーネントが提供するインターフェース関数を見つけることがで

©2013-2019 Tencent Cloud. All rights reserved.

きます。



オープンソースコンポーネントTUICallingのTUICallingManagerをそのまま使用し、オーディオビデオ通話機能を手軽に実装することができます。もう複雑な通話UIやロジックの実装をご自身で行う必要はありません。

手順1：SDKの統合

オーディオビデオ通話コンポーネントTUICallingは、TRTC SDKとIM SDKに依存しています。次の手順で2つのSDKをプロジェクトに統合することができます。

方法1：mavenCentralリポジトリを介する依存

1. dependenciesにTRTCSDKとIMSDKの依存を追加します。

```
dependencies {
compile "com.tencent.liteav:LiteAVSDK_TRTC:latest.release"
compile 'com.tencent.imsdk:imsdk:latest.release'

// 当社はgson解析を使用していますので、さらにgoogle のGsonに依存する必要はありません。
compile 'com.google.code.gson:gson:latest.release'
}
```

2. defaultConfigでAppが使用するCPUアーキテクチャを指定します。

```
defaultConfig {
ndk {
abiFilters "armeabi-v7a"
}
}
```

3. **Sync Now**をクリックし、SDKを同期します。

説明：

©2013-2019 Tencent Cloud. All rights reserved.

> mavenCentral へのネットワーク接続に問題がない場合、SDKは自動的にダウンロードされ、プロジェクトに統合されます。

方法2：ローカルAARを介する依存

開発環境でのmavenCentralリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定および難読化ルール

AndroidManifest.xmlにAppの権限を設定します。SDKには次の権限が必要です（6.0以上のAndroidシステムではカメラ、読み取りストレージの権限を動的に申請する必要があります）。

```
<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" />
```

proguard-rules.proファイルでは、SDK関連を非難読化リストに追加します。

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

手順3：TUICallingコンポーネントをインポート

Sourceディレクトリをプロジェクトにコピーし、 `setting.gradle` へのインポートを完了します。以下をご参照ください。

```
include ':Source'
```

©2013-2019 Tencent Cloud. All rights reserved.

手順4：コンポーネントの初期化およびログイン

1. `TUICallingManager.sharedInstance()` を呼び出し、コンポーネントを初期化します。
2. `TUILogin.init(context, SDKAppID, config, listener)` を呼び出し、ログインを初期化します。
3. `TUILogin.login(userId, userSig, callback)` を呼び出し、コンポーネントのログインを完了します。このうちの重要なパラメータの入力については、下表をご参照ください。

パラメータ名	作用
SDKAppID	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。計算方法については UserSigの計算方法をご参照ください。
config	SDKの設定。ログレベルおよびログコールバック（nullを渡すことも可能）の設定に使用します。詳細については、下記サンプルコードをご参照ください。
listener	IMリスナー。いくつかの必要なシステムコールバック通知の受信に使用します。例：キックアウトによるオフライン、userSig期限切れなど。詳細については、下記サンプルコードをご参照ください。
callback	ログイン結果リスナー。ログイン成功の有無を通知します。詳細については、下記サンプルコードをご参照ください。

```
// コンポーネントの初期化
TUICallingManager manager = TUICallingManager.sharedInstance();
// ログイン
V2TIMSDKConfig config = new V2TIMSDKConfig();
config.setLogLevel(V2TIMSDKConfig.V2TIM_LOG_DEBUG);
config.setLogListener(new V2TIMLogListener() {
@Override
public void onLog(int logLevel, String logContent) {

}
});
TUILogin.init(this, ${あなたのSDKAPPID}, config, new V2TIMSDKListener() {
@Override
public void onKickedOffline() { // ログインがキックアウトされたオフライン通知
mIsKickedOffline = true;
checkUserStatus();
}
@Override
```

©2013-2019 Tencent Cloud. All rights reserved.

```
public void onUserSigExpired() { // suerSig期限切れ通知
mIsUserSigExpired = true;
checkUserStatus();
}
});
TUILogin.login("${あなたのuserId}", "${あなたのuserSig}", new V2TIMCallback() {
@Override
public void onError(int code, String msg) {
Log.d(TAG, "code: " + code + " msg:" + msg);
}
@Override
public void onSuccess() {
Log.d(TAG, "onSuccess");
}
});
```

手順5：オーディオビデオ通話の実装

1. 発信側：TUICallingManagerの `call();` メソッドを呼び出して通話リクエストを送信し、ユーザーIDの配列（userids）と通話タイプ（type）を渡します。通話タイプパラメータは `TUICalling.Type.AUDIO` （オーディオ通話）または `TUICalling.Type.VIDEO` （ビデオ通話）を渡します。ユーザーID配列（userids）のuserIdが1つのみの時は、シングル通話と見なされ、ユーザーID配列（userids）のuserIdが複数（>=2）の時は多人数通話と見なされます。
2. 受信側：受信側がログイン状態の場合は、対応するインターフェースが自動的に起動します。受信側がログインしていない状態でも通話リクエストを受け取りたい場合は、オフライン応答をご参照ください。

```
// 1. コンポーネントの初期化
TUICallingManager manager = TUICallingManager.sharedInstance();
// 2. リスナーの登録
manager.setCallingListener(new TUICalling.TUICallingListener() {
@Override
public boolean shouldShowOnCallView() {
return true;
}

@Override
public void onCallStart(String[] userIDs, TUICalling.Type type, TUICalling.Role role, final View
tuiCallingView) {
if (!shouldShowOnCallView() || null == tuiCallingView) {
return;
}
runOnUiThread(new Runnable() {
@Override
public void run() {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
FrameLayout.LayoutParams params = new FrameLayout.LayoutParams(FrameLayout.LayoutParams.MATCH_PAR
ENT, FrameLayout.LayoutParams.MATCH_PARENT);
mCallingView = tuiCallingView;
addContentView(tuiCallingView, params);
}
});
}

@Override
public void onCallEnd(String[] userIDs, TUICalling.Type type, TUICalling.Role role, long totalTim
e) {
removeView();
}

@Override
public void onCallEvent(TUICalling.Event event, TUICalling.Type type, TUICalling.Role role, Strin
g message) {
if (TUICalling.Event.CALL_FAILED == event) {
removeView();
}
}
});
// 3.電話をかけます
manager.call(userIDs, TUICalling.Type.VIDEO);
```

手順6：オフライン応答の実装

説明：
ビジネスの位置付けがオンラインカスタマーサービスなどのオフライン応答機能を必要としないシーンである場合は、上記[手順1]（＃model.step1）-[手順5]（＃model.step5）の対応で問題ありません。 しかし、ビジネスの位置付けがソーシャルシーンである場合は、オフライン応答を実装することをお勧めします。

IM SDKはオフラインプッシュをサポートしていますが、Androidの各スマートフォンメーカーは各自のオフラインプッシュサービスを備えていることから、iOSプラットフォームよりアクセスが複雑となり、利用基準を満たすためには、相応の設定を行う必要があります。

1. 対応するメーカーのプッシュチャネルに必要な証明書などを申請の上、IMコンソール内に設定し、プッシュ要件に従って証明書とIDなどを追加します。詳細な操作手順については、IM > オフラインプッシュ（Android）をご参照ください。
2. 現在、TRTCCallingImplのsendModelシグナリング送信関数にオフライン送信関数が統合されており、Appのオフラインプッシュを設定した後、メッセージのオフラインプッシュを実装することができます。

©2013-2019 Tencent Cloud. All rights reserved.

コンポーネントAPIリスト

TUICallingコンポーネントのAPIインターフェースリストは次のとおりです。

インターフェース関数	インターフェースの機能
call	C2C通話に招待します
receiveAPNSCalled	被招待者として着信に応答します
setCallingListener	リスナーを設定します
setCallingBell	着信音の設定(30s以内を推奨)
enableMuteMode	ミュートモードをオンにします
enableCustomViewRoute	カスタムビューをオンにします。オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewのインスタンスを受信します。開発者自身が表示方式を決定します。注意：フルスクリーンまたはスクリーンと同じ比率で表示する必要があります。そうしない場合、表示に異常が生じます

©2013-2019 Tencent Cloud. All rights reserved.

ビデオ通話(Web)

最終更新日：：2022-04-02 16:27:03

このドキュメントは2つのパートから成り、ブラウザで実行できる一連のビデオ通話ソリューションを実装する方法について説明します。

- パート1：サービスをアクティブ化する方法および提供するDemoのクイックスタートについて説明します。
- パート2：TRTCCallingコンポーネントを使用して、独自のビデオ通話機能をすばやく構築する方法について説明します。

環境要件

最新バージョンのChromeブラウザをご使用ください。現在、デスクトップのChromeブラウザはTRTC Web SDKをサポートしており、関連機能は比較的整っていますので、Chromeブラウザを使用して体験することをお勧めします。

- TRTCCallingは以下のポートに依存してデータ送信を行っていますので、これらをファイアウォール・ホワイトリストに追加してください。設定が完了したら、公式サイトDemoにアクセスして体験していただけば、設定が有効かどうかをチェックすることができます。
  - TCPポート：8687
  - UDPポート：8000、8080、8800、843、443、16285
  - ドメイン名：qcloud.rtc.qq.com
    具体的には、ファイアウォール制限の対処関連をご参照ください。

サポートプラットフォーム

現在、このソリューションは、次のプラットフォームをサポートしています。

OS	ブラウザタイプ	ブラウザの最小バージョン要件
Mac OS	デスクトップ版Safariブラウザ	11+
Mac OS	デスクトップ版Chromeブラウザ	56+
Mac OS	デスクトップ版Firefoxブラウザ	56+
Mac OS	デスクトップ版Edgeブラウザ	80+
Windows	デスクトップ版Chromeブラウザ	56+

©2013-2019 Tencent Cloud. All rights reserved.

OS	ブラウザタイプ	ブラウザの最小バージョン要件
Windows	デスクトップ版QQブラウザ（超高速コア）	10.4+
Windows	デスクトップ版Firefoxブラウザ	56+
Windows	デスクトップ版Edgeブラウザ	80+

説明：
互換性照会の詳細については、ブラウザサポート状況をご参照ください。また、TRTC検査ページでオンライン検査することも可能です。

URLドメイン名プロトコルの制限

ユースケース	プロトコル	受信（再生）	送信（マイク・オン）	画面共有	備考
本番環境	HTTPSプロトコル	サポートあり	サポートあり	サポートあり	推奨
本番環境	HTTPプロトコル	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	http://localhost	サポートあり	サポートあり	サポートあり	推奨
ローカル開発環境	http://127.0.0.1	サポートあり	サポートあり	サポートあり	-
ローカル開発環境	http://[ローカルマシンIP]	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	file:///	サポートあり	サポートあり	サポートあり	-

Demoのクイックスタートテスト

手順1：アプリケーションの新規作成

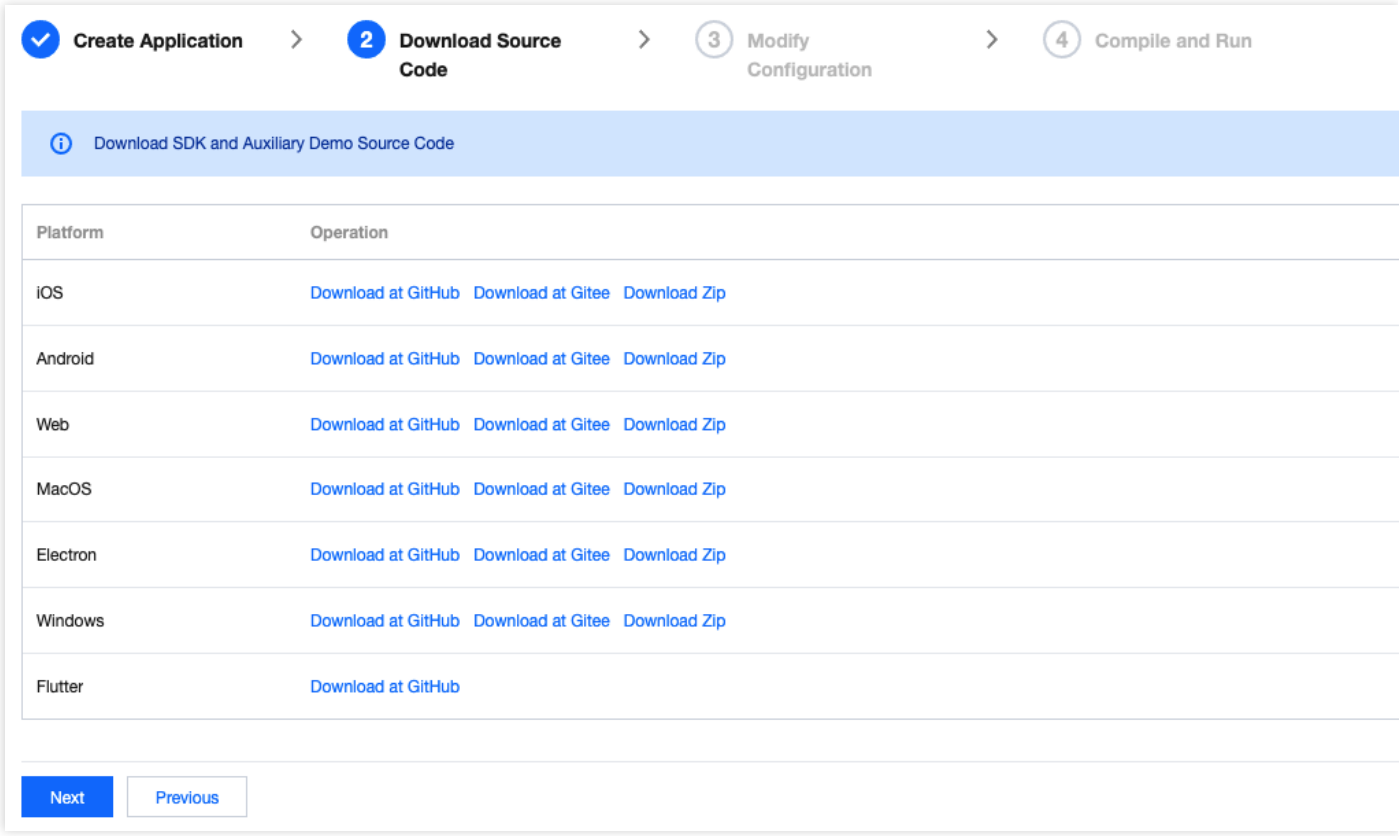
1. Tencent Cloudアカウントの登録を行い、実名認証を完了させます。

©2013-2019 Tencent Cloud. All rights reserved.

2. TRTCコンソールにログインし、**開発支援** > **Demoクイックスタート**を選択します。
3. アプリケーション名（例：TestTRTC）を入力し、**作成**をクリックします。

手順2：Demoのダウンロード

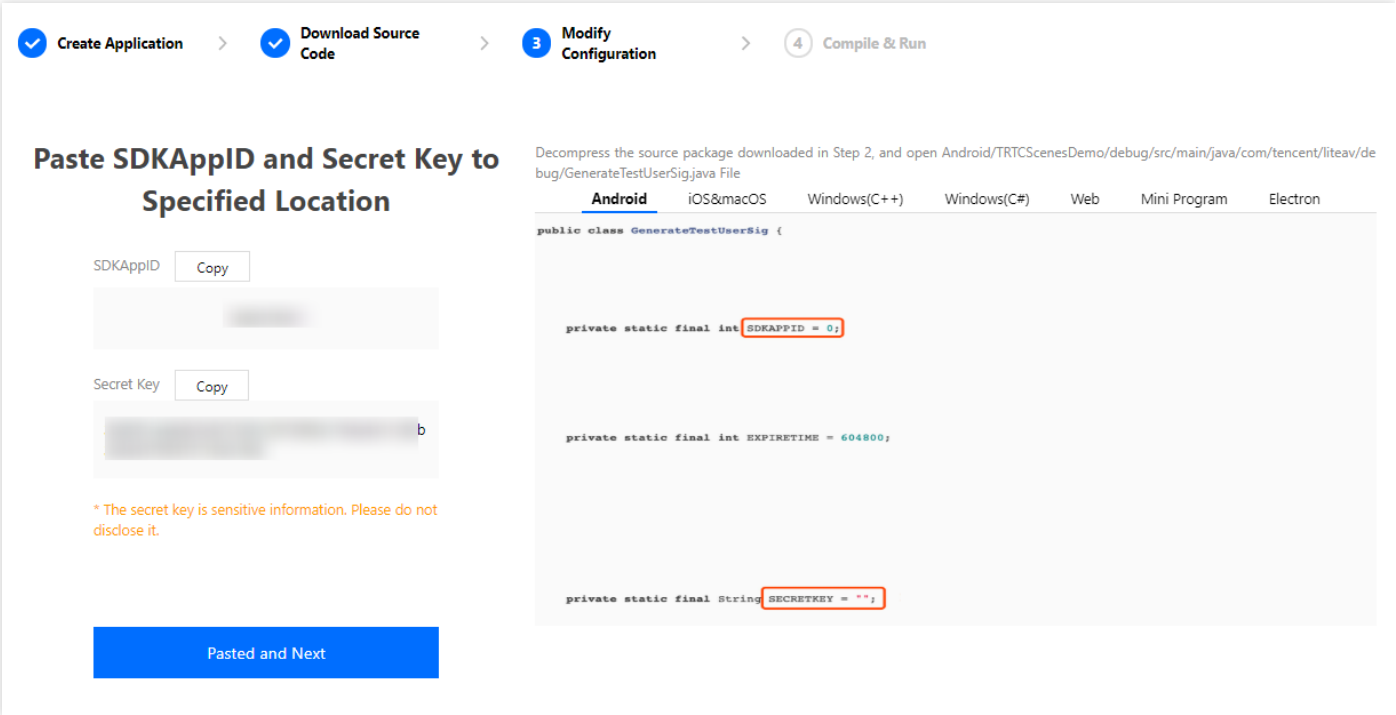
1. 実際のサービスニーズに応じてDemoソースコードをダウンロードするか、または、クリックしてTUICallingに進み、Demoソースコードをダウンロードします。
2. ダウンロード完了後、**ダウンロード完了、次のステップへ**をクリックします。



ステップ3：Demo プログラムファイルの設定

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。
2. `Web/public/debug/GenerateTestUserSig.js` ファイルを見つけて開きます。
3. `GenerateTestUserSig.js` のファイルの関連するパラメータを設定します。
   - SDKAPPID：デフォルトでは0です。実際のSDKAppIDに設定してください。
   - SECRETKEY：デフォルトでは空文字列です。実際のキー情報に設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、**貼り付け完了、次のステップへ**をクリックすると、作成が完了します。
5. コンパイル完了後、**コンソール概要に戻る**をクリックすると、終了します。

注意：

- ここで言及したUserSigの作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appからサービスサーバーにリクエストを発出し動的にUserSigを取得します。詳細はサーバーでのUserSig作成をご参照ください。

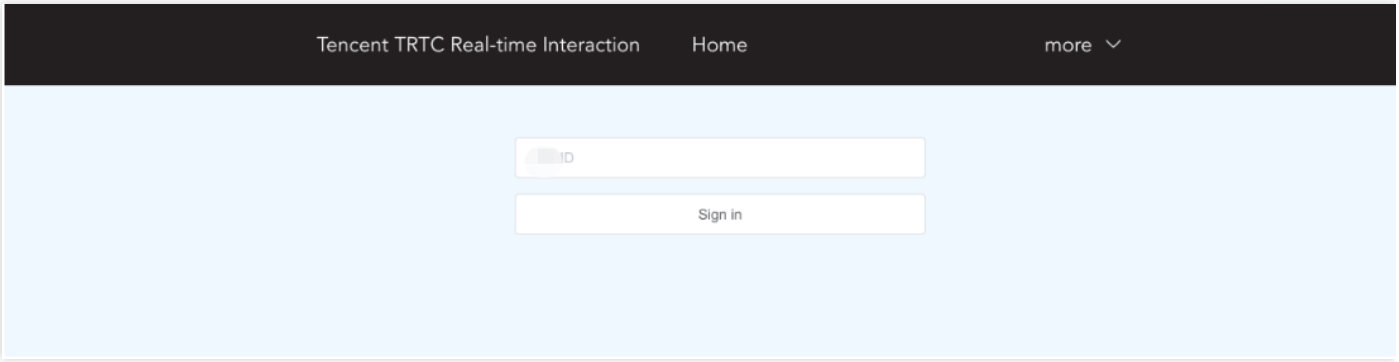
手順4：Demoの実行

1. Demoディレクトリに次のコマンドを順番に入力します。

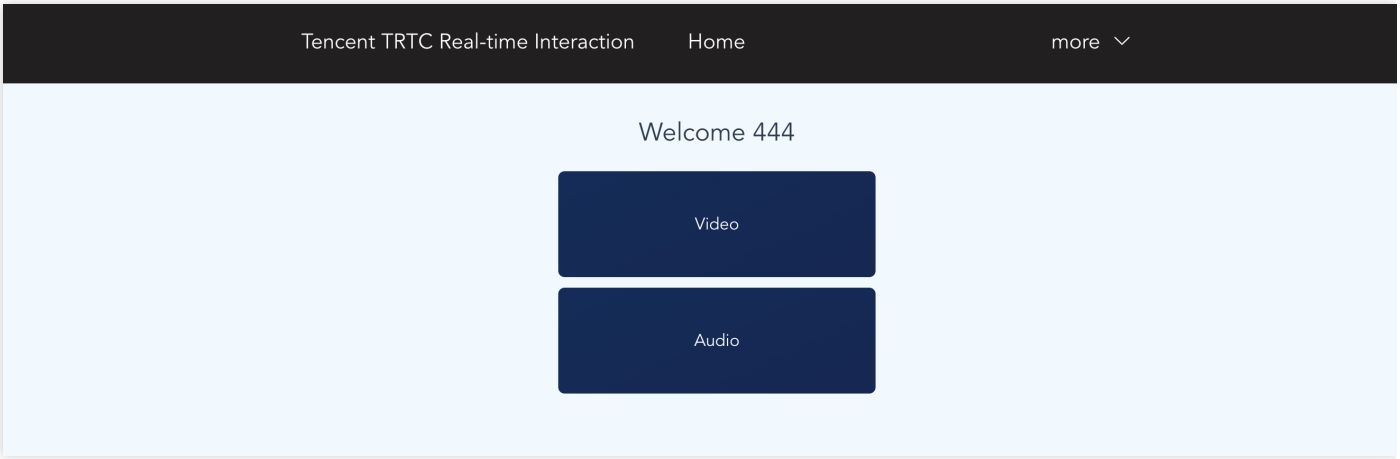
```
npm install
npm run serve
```

©2013-2019 Tencent Cloud. All rights reserved.

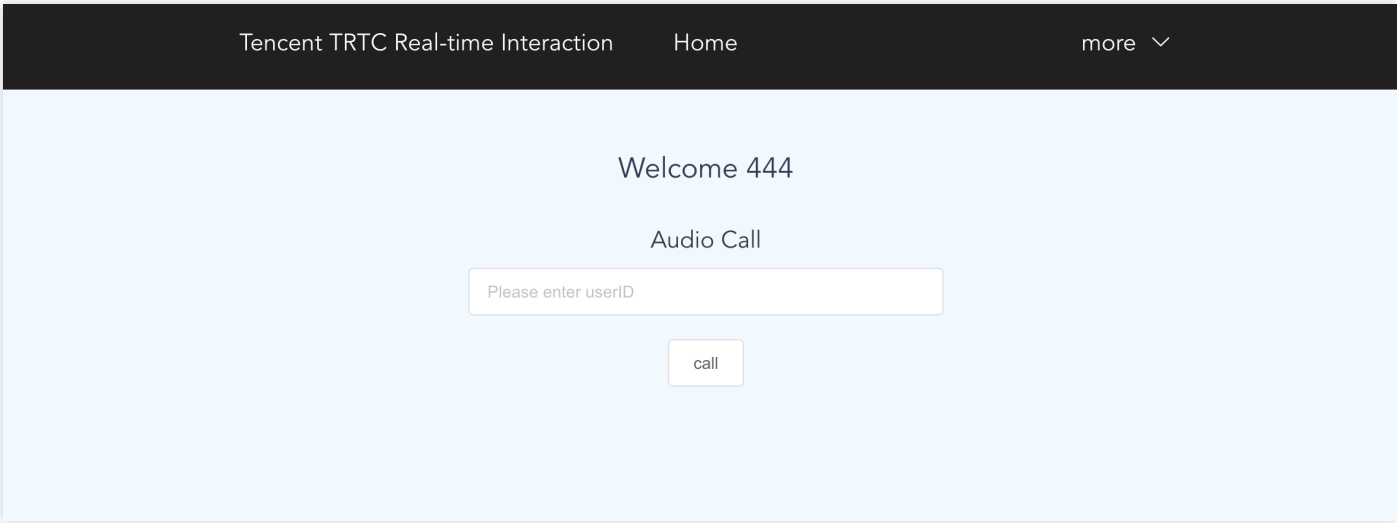
2. Chromeブラウザを起動し、リンク `http://localhost:8080/` を開きます。すべてが正常であれば、Demo実行インターフェースは次の図のようになります。



3. useridを入力し、**ログイン**をクリックして、**ビデオ通話**を選択します。

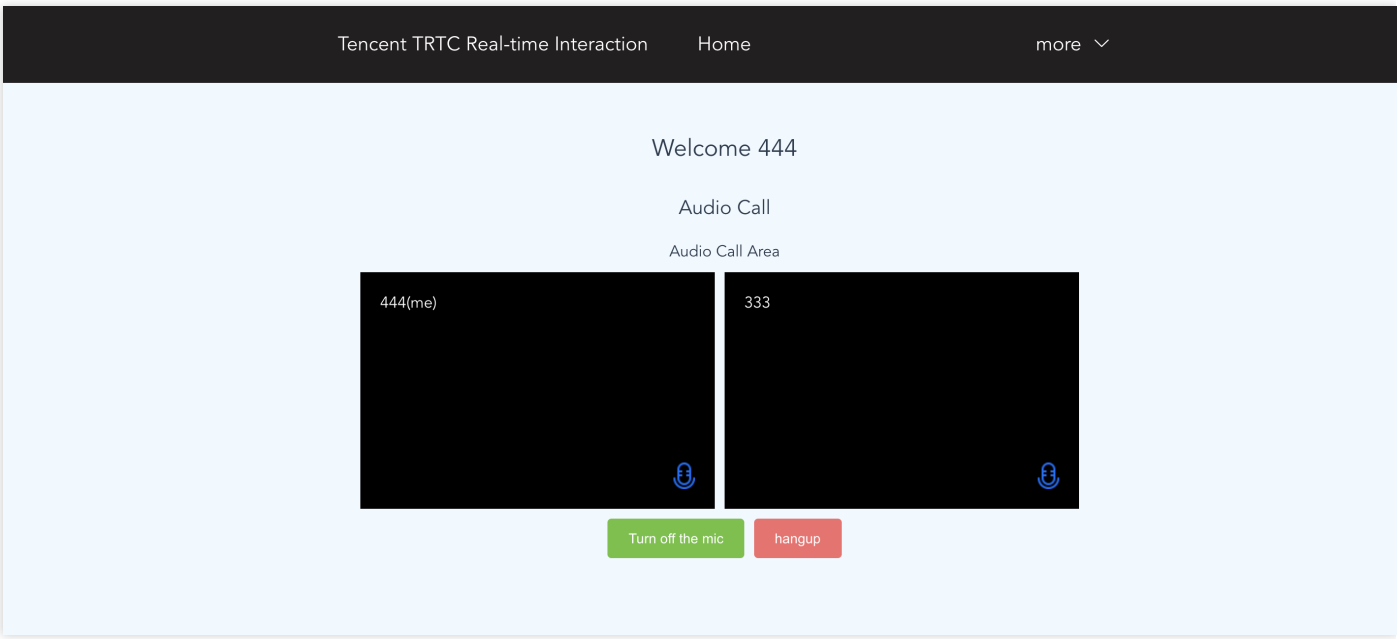


4. 呼び出し先のuseridを入力し、**呼出**をクリックします。



©2013-2019 Tencent Cloud. All rights reserved.

5. ビデオ通話を開始できます。



独自のビデオ通話の作成

手順1：TRTCCallingコンポーネントの統合

説明：

- 0.6.0以降は、trtc-js-sdk、tim-js-sdkおよびtsignalingの依存関係を手動でインストールする必要があります。
- trtc-calling-js.jsのボリュームを縮小し、アクセス側で使用されているtrtc-js-sdk、tim-js-sdkおよびtsignalingのバージョン競合を回避するために、trtc-calling-js.jsは、外部依存関係としてtrtc-js-sdk、tim-js-sdk、tsignalingをパッケージ化しているので、使用する前に依存関係を手動でインストールする必要があります。

```
// npm方式によるインストール
npm install trtc-js-sdk --save

npm install tim-js-sdk --save

npm install tsignaling --save

npm install trtc-calling-js --save
```

©2013-2019 Tencent Cloud. All rights reserved.

```
//スクリプト方式によってtrtc-calling-jsを使用する必要がある場合は、順序どおりに次のリソースをインポートする必要があります。

<script src="./trtc.js"></script>
<script src="./tim-js.js"></script>
<script src="./tsignaling.js"></script>
<script src="./trtc-calling-js.js"></script>
```

手順2：TRTCCallingオブジェクトの作成

TRTCCallingオブジェクトを作成し、SDKAppIDパラメータをお客様のSDKAppIDに設定します。

```
import TRTCCalling from 'trtc-calling-js';

let options = {
SDKAppID: 0, // アクセスするときは、0をお客様のSDKAppIDに置き換える必要があります。
// v0.10.2から、timパラメータを追加します
// timパラメータはサービス内にすでに存在するTIMインスタンスに使用され、TIMインスタンスの一意性を保証します。
tim: tim
};
const trtcCalling = new TRTCCalling(options);
```

手順3：ログインの完了

login関数を呼び出してログイン操作を完了します。パラメータのuserIDはユーザー名、userSigはユーザー署名です。userSigの計算方法については、userSigの計算使用方法をご参照ください。

```
trtcCalling.login({
userID,
userSig,
});
```

手順4：1v1通話の実現

- **発呼者：特定ユーザーを呼び出す**

```
trtcCalling.call({
userID, //ユーザーID
type: 2, //通話タイプ。0-不明、1-音声通話、2-ビデオ通話
timeout //招待タイムアウト時間、単位s（秒）
});
```

©2013-2019 Tencent Cloud. All rights reserved.

- **着呼者：新しい呼び出しに応答**

```
// 応答
trtcCalling.accept();
//拒否します
trtcCalling.reject()
```

- **ローカルカメラのオン**

```
trtcCalling.openCamera()
```

- **リモートビデオ画面の表示**

```
trtcCalling.startRemoteView({
userID, //リモートユーザーID
videoViewDomID //このユーザーデータはDOM IDノードにレンダリングされます
})
```

- **ローカルプレビュー画面の表示**

```
trtcCalling.startLocalView({
userID, //ローカルユーザーID
videoViewDomID //このユーザーデータはDOM IDノードにレンダリングされます
})
```

- **通話終了**

```
trtcCalling.hangup()
```

よくあるご質問

通話がつながらなかったり、強制オフラインになったりするのはなぜですか。

コンポーネントは現在、マルチインスタンスのログインや**オフラインプッシュのシグナリング**機能をサポートしていません。現在のログインアカウントの一意性をご確認ください。

> 説明：

©2013-2019 Tencent Cloud. All rights reserved.

- **マルチインスタンス**：1つのUserIDで繰り返しログインしたり、異なるターミナルからログインしたりすると、シグナリングの混乱が生じます。
- **オフラインプッシュ**：インスタンスはオンラインの場合にのみメッセージを受信できます。インスタンスがオフラインのときに受信したシグナリングは、オンラインになった後は再度プッシュされません。その他よくあるご質問については、TRTCCalling Webに関するご質問をご参照ください。

技術的なお問い合わせ

詳細については、お問い合わせにご連絡をいただくか、colleenyu@tencent.comにメールでご連絡ください。

参考ドキュメント

- TRTCCalling web公式サイトで体験
- TRTCCalling npm
- TRTCCalling web demoソースコード
- TRTCCalling web API
- TRTCCalling webに関するご質問

©2013-2019 Tencent Cloud. All rights reserved.

ビデオ通話(Flutter)

最終更新日：：2022-02-18 12:25:48

ビデオ通話機能をすばやく実装する必要がある場合、当社が提供するDemoをもとにアダプターを修正するか、または当社が提供するTRTCCalling コンポーネントでUIのカスタマイズを実装することができます。

DemoのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、【開発支援】>【快速跑通Demo】を選択します。
2. `TestVideoCall` などアプリケーション名を入力して、【作成】をクリックします。

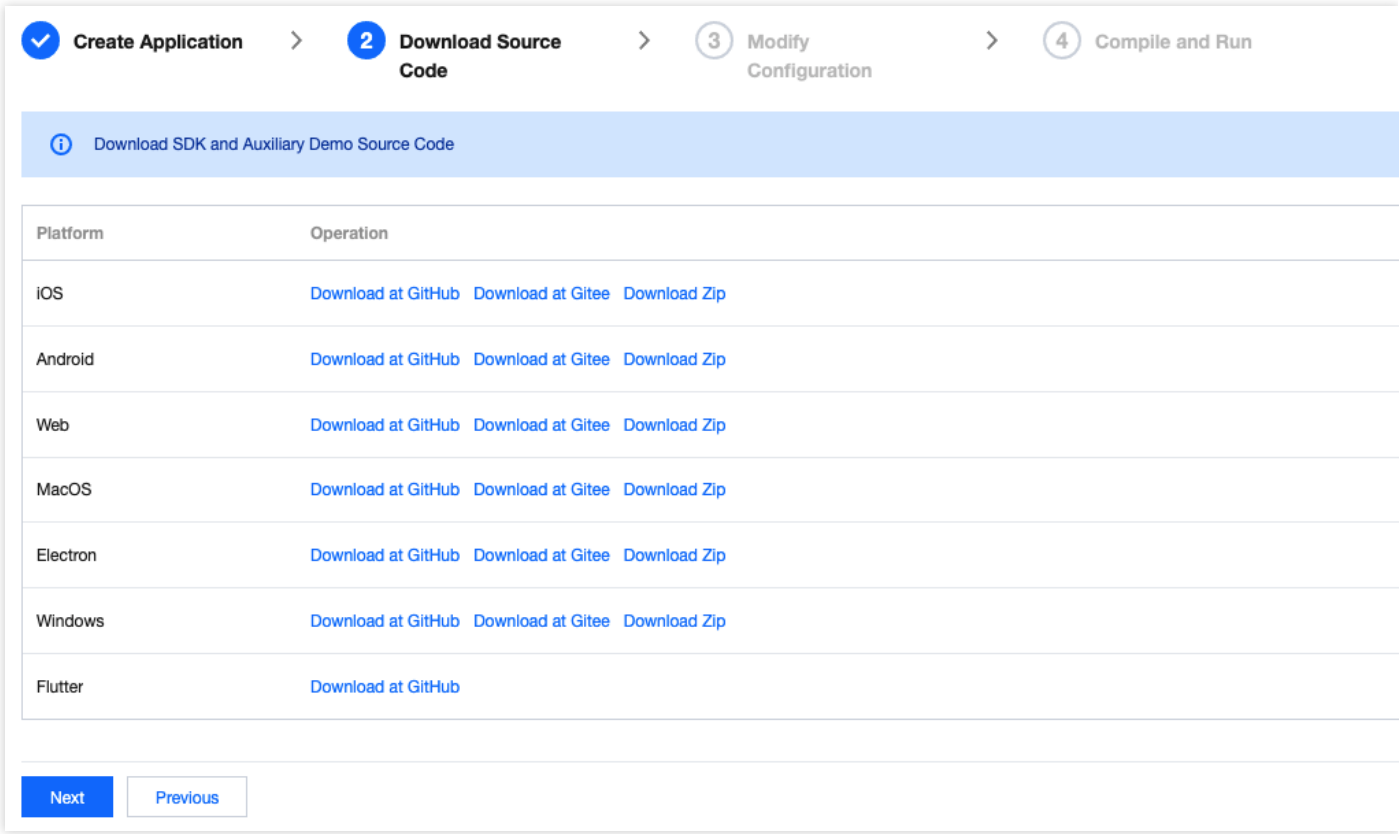
> 注意：
> 本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

ステップ2：SDKおよびDemoソースコードをダウンロード

1. 実際のビジネスニーズに応じて、SDKと付属のDemoソースコードをダウンロードします。

©2013-2019 Tencent Cloud. All rights reserved.

2. ダウンロード完了後、【ダウンロード済み。次へ】をクリックします。



ステップ3：Demo プログラムファイルの設定

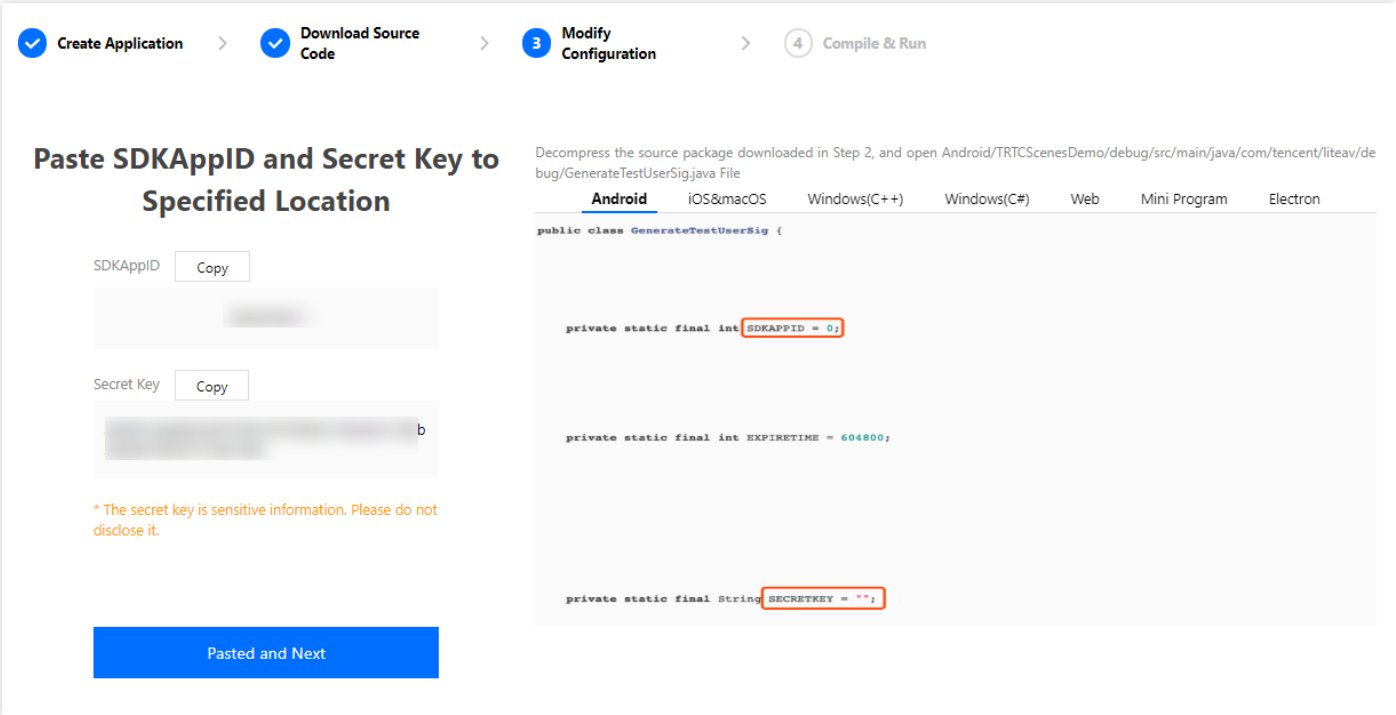
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `/example/lib/debug/GenerateTestUserSig.dart` ファイルを見つけて開きます。

3. `GenerateTestUserSig.dart` ファイル内の関連パラメータを設定します。

   - SDKAPPID：デフォルトはPLACEHOLDER、実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトはPLACEHOLDER、実際のキー情報を設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、【貼り付けました。次へ】をクリックすれば、作成が完了します。

5. コンパイル完了後、【コンソールの概要ページに戻る】をクリックすればOKです。

注意：

- ここで言及したUserSigの新規作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Demoの動作

注意：
Android端末は実際のマシンで実行する必要があり、エミュレーターのデバッグはサポートされていません。

©2013-2019 Tencent Cloud. All rights reserved.

1. `flutter pub get` を実行します。
2. コンパイルを実行し、デバッグを行います。
   - Android 端末
   - iOS 端末
   i. `flutter run` を実行します。
   ii. Android Studio（3.5およびそれ以降のバージョン）を使用して、ソースプロジェクトを開き、【実行】をクリックすれば完了です。

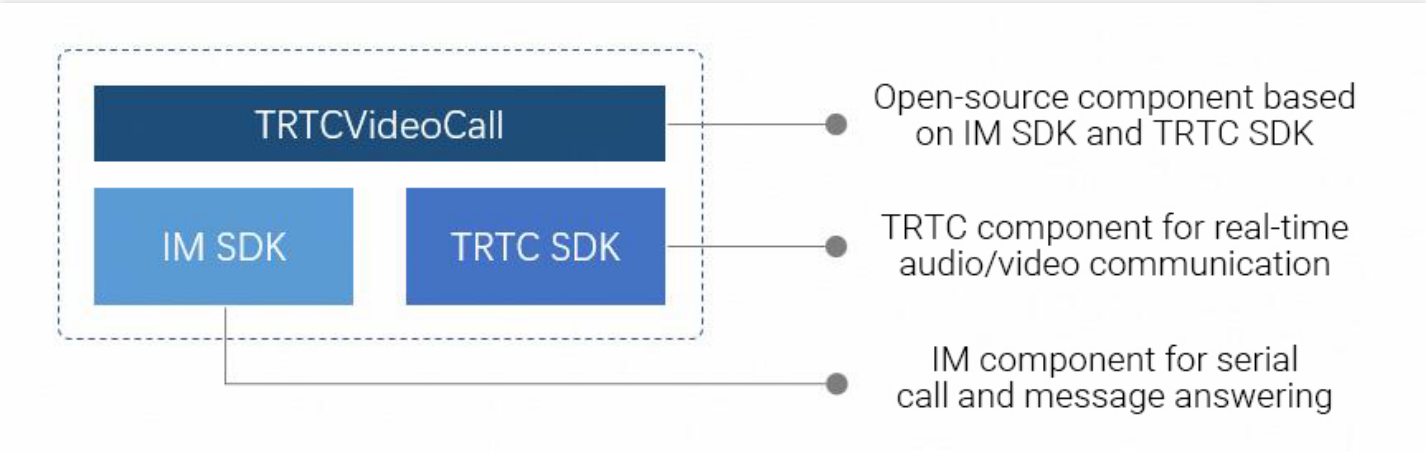
手順5：Demoソースコードの修正

ソースコードフォルダ `TRTCCallingDemo` には2つのサブフォルダuiとmodelが含まれ、そのうちuiフォルダにはインターフェースコードが含まれています。

ファイルまたはフォルダ	機能の説明
TRTCCallingVideo.dart	オーディオビデオ通話のメインインターフェースが表示され、このインターフェースで通話の応答と拒否が完了します。
TRTCCallingContact.dart	連絡先を選択するためのインターフェースを表示するために使用され、このインターフェースを介して登録済みユーザーを検索し、通話を開始できます

UIカスタマイズの実装

ソースコードフォルダ `TRTCCallingDemo` には2つのサブフォルダuiとmodelが含まれ、そのうちmodelフォルダには当社が実装した再利用可能なオープンソースコンポーネントTRTCCallingが含まれています。このコンポーネントが提供するインターフェース関数は `TRTCCalling.dart` ファイルで確認できます。



オープンソースコンポーネントTRTCCallingを使用すれば、自分のUIを実装することができます。すなわちmodel パーツを再利用するだけで、自分でUIパーツを実装できます。

©2013-2019 Tencent Cloud. All rights reserved.

手順1：SDKの統合

オーディオビデオ通話コンポーネントTRTCCallingは、TRTC SDKとIM SDKに依存しています。 `pubspec.yaml` を設定することで、更新を自動的にダウンロードできます。

プロジェクトの `pubspec.yaml` に次の依存関係を記述します。

```
dependencies:
tencent_trtc_cloud: 最新バージョン番号
tencent_im_sdk_plugin: 最新バージョン番号
```

手順2：権限の設定および難読化ルール

- iOS 端末
- Android 端末

`Info.plist` にカメラとマイクの権限申請を追加する必要があります。

```
<key>NSMicrophoneUsageDescription</key>
<string>通常の音声通話が行えるようにマイクの権限を承認します</string>
```

手順3：TRTCCalling コンポーネントをインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
/lib/TRTCCallingDemo/model
```

手順4：コンポーネントの初期化およびログイン

1. `TRTCCalling.sharedInstance()` を呼び出して、コンポーネントインスタンスを取得します。
2. `login(SDKAppID, userId, userSig, callback)` を呼び出し、コンポーネントのログインを完了します。このうちいくつかの重要パラメータの入力については、下表をご参照ください。

パラメータ名	作用
SDKAppID	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。計算方法については UserSigの計算方法をご参照ください。

©2013-2019 Tencent Cloud. All rights reserved.

```
// 初期化
sCall = await TRTCCalling.sharedInstance();
sCall.login(1400000123, "userA", "xxxx");
```

手順5：1v1ビデオ通話を実現

1. 発信者： `TRTCCalling` の `call()` メソッドを呼び出して通話リクエストを開始し、ユーザーID（userid）と通話タイプ（type）を渡して、通話タイプパラメータを `TRTCCalling.typeVideoCall` に渡します。
2. 受信者：受信者がログイン状態の場合は、 `onInvited()` という名称のイベント通知を受け取ります。コールバックにおける `callType` のパラメータは発信者が入力した通話タイプであり、このパラメータを介して対応するインターフェースを起動することができます。
3. 受信者：電話に対応したい場合、受信者は `accept()` 関数をコールすると同時に `openCamera()` 関数を呼び出して、自身のローカルCCDカメラを起動します。受信者は `reject()` を呼び出し、この通話を拒否することもできます。
4. 双方のオーディオ・ビデオチャネルが確立した後、通話する双方は `onUserVideoAvailable()` という名称のイベント通知を受け取ります。これは、相手方のビデオ画面を取得済みであることを示しています。このとき、双方のユーザーは `startRemoteView()` を呼び出し、リモートのビデオ画面を表示することができます。リモートの音声はデフォルトで自動再生に設定されています。

コンポーネントAPIリスト

TRTCCallingコンポーネントのAPIインターフェースリストは次のとおりです。

インターフェース関数	インターフェースの機能
registerListener	TRTCCalling監視装置を追加します。ユーザーはこの監視装置を介してステータス通知を取得することができます
unRegisterListener	監視装置を削除します
destroy	インスタンスを破棄します
login	IMにログインします。すべての機能を使用するためには、まずログインする必要があります
logout	IMからログアウトします。ログアウト後はダイヤル操作ができません
call	C2Cの通話に招待します。被招待者はonInvitedのイベント通知を受け取ります
accept	被招待者として通話に応答します
reject	被招待者として通話を拒否します

©2013-2019 Tencent Cloud. All rights reserved.

インターフェース関数	インターフェースの機能
hangup	通話を終了します
startRemoteView	リモートユーザーのカメラデータを指定のTXCloudVideoViewにレンダリングします
stopRemoteView	任意のリモートユーザーのカメラデータのレンダリングを停止します
openCamera	カメラを起動し、指定のTXCloudVideoViewにレンダリングします
closeCamera	カメラを終了します
switchCamera	フロント/リアカメラを切り替えます
setMicMute	micをミュート/非ミュートに設定します
setHandsFree	ハンズフリーを起動するかどうかを設定します

©2013-2019 Tencent Cloud. All rights reserved.

TUICalling API(iOS)

最終更新日：：2021-12-07 10:29:43

TUICallingは、Tencent CloudのTRTCとIMを組み合わせたサービスであり、1v1および多人数のビデオ通話をサポートしています。TUICallingはTencent Cloudの2つのクローズドソースSDKに依存するオープンソースのClassです。具体的な実装プロセスについては、リアルタイムビデオ通話（iOS）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

TUICalling API概要

SDK 基本関数

API	説明
sharedInstance	コンポーネントシングルトン。
call	C2C通話に招待します。
receiveAPNSCalled	被招待者として着信に応答します。
setCallingListener	リスナーを設定します。
setCallingBell	着信音の設定（30s以内を推奨）
enableMuteMode	ミュートモードをオンにします
enableCustomViewRoute	カスタムビューの有効化

TUICallingListener API概要

イベントコールバック

API	説明
shouldShowOnCallView	呼び出された時の応答ページのプルリクエスト
onCallStart	呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます
onCallEnd	通話コールバック。発呼側、着呼側ともにトリガーされます

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onCallEvent	通話イベントコールバック

Type API概要

通話タイプ

enum	説明
TUICallingTypeAudio	オーディオ通話
TUICallingTypeVideo	ビデオ通話

Role API概要

ユーザーロールタイプ

enum	説明
TUICallingRoleCall	通話発信者（発呼側）
TTUICallingRoleCalled	通話受信者（着呼側）

Event API概要

イベントタイプ

enum	説明
TUICallingEventCallStart	通話開始
TUICallingEventCallSucceed	通話接続成功
TUICallingEventCallEnd	通話終了
TUICallingEventCallFailed	通話失敗

SDK基本関数

©2013-2019 Tencent Cloud. All rights reserved.

sharedInstance

shareInstanceはTUICallingManagerのコンポーネントシングルトンです。

```
+ (instancetype)shareInstance;
```

call

C2C通話に招待します。

```
- (void)call:(NSArray<NSString *> *)userIDs type:(TUICallingType)type;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ

receiveAPNSCalled

被招待者として着信に応答します。

```
- (void)receiveAPNSCalled:(NSArray<NSString *> *)userIDs type:(TUICallingType)type;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ

setCallingListener

リスナーを設定します。

```
- (void)setCallingListener:(id<TUICallingListerner>)listener;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
listener	TUICallingListener	TUIcallingコンポーネントリスナー

©2013-2019 Tencent Cloud. All rights reserved.

setCallingBell

着信音を設定します（30s以内を推奨）。

```
- (void)setCallingBell:(NSString *)filePath;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
filePath	NSString	着信音リソースパス

enableMuteMode

ミュートモードをオンにします。

```
- (void)enableMuteMode:(BOOL)enable;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	BOOL	ミュートモード有効化の有無

enableCustomViewRoute

カスタムビューをオンにします。
オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewControllerのインスタンスを受信します。開発者自身が表示方式を決定します。

> 注意：
> フルスクリーン表示にする必要があります。そうしない場合、表示に異常が生じます。

```
- (void)enableCustomViewRoute:(BOOL)enable;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	BOOL	カスタムビュー有効化の有無

©2013-2019 Tencent Cloud. All rights reserved.

TUICallingListenerコールバック関数

shouldShowOnCallView

呼び出された時の応答ページのプルリクエストに対する同意の有無。

```
- (BOOL)shouldShowOnCallView;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
戻り値	BOOL	同意の有無

onCallStart

呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます。

```
- (void)callStart:(NSArray<NSString *> *)userIDs type:(TUICallingType)type role:(TUICallingRole)r
ole viewController:(UIViewController * _Nullable)viewController;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ
role	TUICallingRole	ユーザーロールタイプ：発呼側/着呼側
viewController	UIViewController	通話ビューViewController

onCallEnd

通話終了コールバック。発呼側、着呼側ともにトリガーされます。enableCustomViewRouteの設定がNOの時、このコールバックメソッドはトリガーされません。

```
- (void)callEnd:(NSArray<NSString *> *)userIDs type:(TUICallingType)type role:(TUICallingRole)rol
e totalTime:(CGFloat)totalTime;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ
role	TUICallingRole	ユーザーロールタイプ：発呼側/着呼側
totalTime	CGFloat	通話時間。単位：秒

onCallEvent

通話イベントコールバック。enableCustomViewRouteの設定がNOの時、このコールバックメソッドはトリガーされません。

```
- (void)onCallEvent:(TUICallingEvent)event type:(TUICallingType)type role:(TUICallingRole)role message:(NSString *)message;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
event	TUICallingEvent	通話イベントタイプ。
type	TUICallingType	通話タイプ：オーディオ/ビデオ
role	TUICallingRole	ユーザーロールタイプ：発呼側/着呼側
message	NSString	イベントの説明情報

©2013-2019 Tencent Cloud. All rights reserved.

TUICalling API(Android)

最終更新日：：2021-12-07 10:29:44

TUICallingは、Tencent CloudのTRTCとIMを組み合わせたサービスであり、1v1および多人数のビデオ通話をサポートしています。TUICallingはTencent Cloudの2つのクローズドソースSDKに依存するオープンソースのClassです。具体的な実装プロセスについては、リアルタイムビデオ通話（Android）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

TUICalling API概要

SDK 基本関数

API	説明
sharedInstance	コンポーネントシングルトン
call	C2C通話に招待します
receiveAPNSCalled	被招待者として着信に応答します
setCallingListener	リスナーを設定します。
setCallingBell	着信音の設定（30s以内を推奨）
enableMuteMode	ミュートモードをオンにします
enableCustomViewRoute	カスタムビューの有効化

TUICallingListener API概要

イベントコールバック

API	説明
shouldShowOnCallView	呼び出された時の応答ページのプルリクエスト
onCallStart	呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます
onCallEnd	通話コールバック。発呼側、着呼側ともにトリガーされます

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onCallEvent	通話イベントコールバック

Type API概要

通話タイプ

enum	説明
AUDIO	オーディオ通話
VIDEO	ビデオ通話

Role API概要

ユーザーロールタイプ

enum	説明
CALL	通話発信者（発呼側）
CALLED	通話受信者（着呼側）

Event API概要

イベントタイプ

enum	説明
CALL_START	通話開始
CALL_SUCCEED	通話接続成功
CALL_END	通話終了
CALL_FAILED	通話失敗

SDK基本関数

©2013-2019 Tencent Cloud. All rights reserved.

sharedInstance

shareInstanceはTUICallingのコンポーネントシングルトンです。

```
public static TUICallingManager sharedInstance();
```

call

C2C通話に招待します。

```
void call(String[] userIDs, Type type);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ

receiveAPNSCalled

被招待者として着信に応答します。

```
void receiveAPNSCalled(String[] userIDs, Type type);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ

setCallingListener

リスナーを設定します。

```
void setCallingListener(TUICallingListener listener);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
listener	TUICallingListener	TUIcallingコンポーネントリスナー

©2013-2019 Tencent Cloud. All rights reserved.

setCallingBell

着信音を設定します(30s以内を推奨)。

```
void setCallingBell(String filePath);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
filePath	String	着信音リソースパス

enableMuteMode

ミュートモードをオンにします。

```
void enableMuteMode(boolean enable);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	boolean	ミュートモード有効化の有無

enableCustomViewRoute

カスタムビューをオンにします。
オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewのインスタンスを受信します。開発者自身が表示方式を決定します。

> 注意：
> フルスクリーンまたはスクリーンと同じ比率で表示する必要があります。そうしない場合、表示に異常が生じます。

```
void enableCustomViewRoute(boolean enable);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	boolean	カスタムビュー有効化の有無

©2013-2019 Tencent Cloud. All rights reserved.

TUICallingListenerコールバック関数

shouldShowOnCallView

呼び出された時の応答ページのプルリクエストに対する同意の有無。

```
boolean shouldShowOnCallView();
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
戻り値	boolean	同意の有無

onCallStart

呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます。

```
void onCallStart(String[] userIDs, TUICalling.Type type, TUICalling.Role role, View tuiCallingVie
w);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト。
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ
role	TUICalling.Role	ユーザーロールタイプ：発呼側/着呼側
tuiCallingView	View	通話ビューView。enableCustomViewRouteの設定がfalseの時、viewはnullになります。

onCallEnd

通話終了コールバック。発呼側、着呼側ともにトリガーされます。

```
void onCallEnd(String[] userIDs, TUICalling.Type type, TUICalling.Role role, long totalTime);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ
role	TUICalling.Role	ユーザーロールタイプ：発呼側/着呼側
totalTime	long	通話時間。単位：秒

onCallEvent

通話イベントコールバック。

```
void onCallEvent(TUICalling.Event event, TUICalling.Type type, TUICalling.Role role, String message);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
event	TUICalling.Event	通話イベントタイプ
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ
role	TUICalling.Role	ユーザーロールタイプ：発呼側/着呼側
message	String	イベントの説明情報

©2013-2019 Tencent Cloud. All rights reserved.

TRTCCalling API(Web)

最終更新日：：2022-04-02 16:27:03

TRTCCallingの概要

TRTCCallingコンポーネントは、Tencent CloudのTencent Real-Time Communication（TRTC）とInstant Messaging（IM）サービスの組み合わせにより構成されており、1v1および複数人でのビデオ/音声通話をサポートしています。具体的な実装プロセスについては、リアルタイムビデオ通話（Web）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

Demoのダウンロード

クリックしてTUICallingに進み、実際のサービスニーズに応じてDemoソースコードをダウンロードします。

環境要件

最新バージョンのChromeブラウザをご使用ください。現在、デスクトップのChromeブラウザはTRTC Web SDKをサポートしており、関連機能は比較的整っていますので、Chromeブラウザを使用して体験することをお勧めします。具体的な内容については、環境要件をご参照ください。

URLドメイン名プロトコルの制限

ユースケース	プロトコル	受信（再生）	送信（マイク・オン）	画面共有	備考
本番環境	HTTPSプロトコル	サポートあり	サポートあり	サポートあり	推奨
本番環境	HTTPプロトコル	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	http://localhost	サポートあり	サポートあり	サポートあり	推奨

©2013-2019 Tencent Cloud. All rights reserved.

ユースケース	プロトコル	受信（再生）	送信（マイク・オン）	画面共有	備考
ローカル開発環境	http://127.0.0.1	サポートあり	サポートあり	サポートあり	-
ローカル開発環境	http://[ローカルマシンIP]	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	file:///	サポートあり	サポートあり	サポートあり	-

TRTCCalling API

イベントサブスクリプション/サブスクリプションのキャンセルに関するインターフェース関数

このコンポーネントはイベントのディスパッチに基づいて管理します。アプリケーション層は、コンポーネントからディスパッチされるイベントに応じてインタラクティブに変更することができます。

API	説明
on(eventName, callback, context)	イベントのサブスクリプション
off(eventName, callback, context)	イベントサブスクリプションのキャンセル

SDK 基本関数

API	説明
login({userID, userSig})	IMインターフェースにログインします。すべての機能を使用するためには、まずログインする必要があります
logout()	インターフェースからログアウトします。ログアウトした後は、ダイヤル操作ができなくなります

通話操作に関連するインターフェース関数

API	説明
call({userID, type, offlinePushInfo}))	シングル通話の招待
groupCall({userIDList, type, groupID, offlinePushInfo})	グループチャット通話の招待
accept()	通話招待に応答

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
reject()	通話招待を拒否
hangup()	現在の通話を終了

ビデオ制御に関連するインターフェース関数

API	説明
startRemoteView({userID, videoViewDomID})	リモート画面レンダリングの起動
stopRemoteView({userID})	リモート画面レンダリングの停止
startLocalView({userID, videoViewDomID})	ローカル画面レンダリングの起動
stopLocalView({userID})	ローカル画面レンダリングの停止
openCamera()	カメラの起動
closeCamera()	カメラの停止
setMicMute(isMute)	デバイスマイクのミュートの有無
setVideoQuality(profile)	ビデオ画質の設定
switchToAudioCall()	ビデオ通話を音声通話へ切り替え
switchToVideoCall()	音声通話をビデオ通話へ切り替え
getCameras()	カメラデバイスリストの取得
getMicrophones()	マイクデバイスリストの取得
switchDevice({deviceType, deviceId})	カメラまたはマイクデバイスの切り替え

TRTCCallingの詳細

TRTCCallingコンポーネントのインスタンスの作成

まず、Tencent Real-Time Communicationコンソールでアプリケーションを作成し、SDKAppIDを取得する必要があります。
その後、 `new TRTCCalling()` メソッドを使ってTRTCCalling コンポーネントのインスタンスを取得することができます。

©2013-2019 Tencent Cloud. All rights reserved.

```
npm i trtc-js-sdk --save
npm i tim-js-sdk --save
npm i tsignaling --save
npm i trtc-calling-js --save
// nodeを通じてダウンロードする依存関係の場合は、importによって次をインポートします
import TRTCCalling from 'trtc-calling-js';
// script方式によってtrtc-calling-jsを使用する場合は、順序どおりに
// trtc.jsを手動でインポートします
<script src="./trtc.js"></script>
// 続いて、tim-js.jsを手動でインポートします。
<script src="./tim-js.js"></script>
// 次に、tsignaling.jsを手動でインポートします。
<script src="./tsignaling.js"></script>
// 最後に、trtc-calling-js.jsを手動でインポートします。
<script src="./trtc-calling-js.js"></script>
let options = {
SDKAppID: 0, // アクセスするときは、0をお客様のIMアプリケーションのSDKAppIDに置き換える必要があります
// v0.10.2から、timパラメータを追加します
// timパラメータはサービス内にすでに存在するTIMインスタンスに使用され、TIMインスタンスの一意性を保証します。
tim: tim
};
let trtcCalling = new TRTCCalling(options);
```

イベントのサブスクリプション/サブスクリプションキャンセルに関するインターフェース関数

on(eventName, callback, context)

コンポーネントからディスパッチされたイベントを監視するために使用します。詳細については、イベントリストをご参照ください。

```
let handleInvite = function ({inviteID, sponsor, inviteData}) {
console.log(`inviteID: ${inviteID}, sponsor: ${sponsor}, inviteData: ${JSON.stringify(inviteDat
a)}`);
};
trtcCalling.on('onInvited', handleInvite, this);
```

off(eventName, callback, context)

イベント監視をキャンセルするために使用します。

```
let handleInvite = function ({inviteID, sponsor, inviteData}) {
console.log(`inviteID: ${inviteID}, sponsor: ${sponsor}, inviteData: ${JSON.stringify(inviteDat
a)}`);
```

©2013-2019 Tencent Cloud. All rights reserved.

```
};
trtcCalling.off('onInvited', handleInvite, this);
```

SDK基本関数

login({userID, userSig})

インターフェースにログインします。

```
trtcCalling.login({userID, userSig})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	現在のユーザーのID。文字列タイプ。アルファベット（a-zおよびA-Z）、数字（0-9）、ハイフン（-）、アンダーバー（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算、使用方法をご参照ください。

logout()

インターフェースからログアウトします。

```
trtcCalling.logout()
```

通話操作に関連するインターフェース関数

call({userID, type, offlinePushInfo})

1v1通話招待、そのうちtypeは通話タイプ、1は音声通話、2はビデオ通話です。

説明：

- v1.0.0以降のバージョンで、timeoutパラメータを取り消しました。
- v1.0.0以降のバージョンに、offlinePushInfoパラメータを新規追加しました（**オフラインプッシュはAndroidまたはiOS端末にのみ適用されます。WebおよびWeChat Mini Programはサポートしていません**）。

```
// v1.0.0以前
trtcCalling.call({userID, type, timeout});
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// v1.0.0以降のバージョン
const offlinePushInfo = {
title: '',
description: '通話リクエストが1件あります',
}
trtcCalling.call({userID, type, offlinePushInfo})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	被招待者のuserID
type	Number	1：音声通話、2：ビデオ通話
timeout	Number	0はタイムアウトしていないことを意味し、単位はs（秒）。**v1.0.0より前のバージョンのみ**
offlinePushInfo	Object	メッセージのオフラインプッシュをカスタマイズします（オプション）。**v1.0.0以降のバージョンのみ**

offlinePushInfoパラメータ（v1.0.0以降のバージョンのみ）

パラメータ	タイプ	意味
title	String	オフラインプッシュタイトル（オプション）
description	String	オフラインプッシュコンテンツ（オプション）
androidOPPOChannelID	String	オフラインプッシュのためのOPPO携帯（システム8.0以上）のチャンネルIDを設定します（オプション）。
extension	String	オフラインプッシュパススルーコンテンツ（オプション）。**TRTCCallingバージョン>=1.0.2、tsignalingバージョン>=0.9.0のみ**

groupCall({userIDList, type, groupID, offlinePushInfo})

groupID パラメータはIM SDKにおけるグループIDです。このパラメータを設定すると、通話リクエストメッセージがグループメッセージシステムを介してブロードキャストされます。このメッセージブロードキャスト方式は簡単で信頼性の高い方法です。設定しない場合は、 `TRTCCalling` コンポーネントが単発メッセージで1人1人に通知を行います。

説明：

©2013-2019 Tencent Cloud. All rights reserved.

v1.0.0以降のバージョンに、offlinePushInfoパラメータを新規追加しました（**オフラインプッシュはAndroidまたはiOS端末にのみ適用されます。WebおよびWeChat Mini Programはサポートしていません**）。

```
// v1.0.0以前
trtcCalling.groupCall({userIDList, type, groupID});

// v1.0.0以降
const offlinePushInfo = {
title: '',
description: '通話リクエストが1件あります',
}
trtcCalling.groupCall({userIDList, type, groupID, offlinePushInfo})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDList	Array	招待リスト
type	Number	1：音声通話、2：ビデオ通話
groupID	String	IMグループID（オプション）
offlinePushInfo	Object	メッセージのオフラインプッシュをカスタマイズします（オプション）。**v1.0.0以降のバージョンのみ**

offlinePushInfoパラメータ（v1.0.0以降のバージョンのみ）

パラメータ	タイプ	意味
title	String	オフラインプッシュタイトル（オプション）
description	String	オフラインプッシュコンテンツ（オプション）
androidOPPOChannelID	String	オフラインプッシュのためのOPPO携帯（システム8.0以上）のチャンネルIDを設定します（オプション）。
extension	String	オフラインプッシュパススルーコンテンツ（オプション）。**TRTCCallingバージョン>=1.0.2、tsignalingバージョン>=0.9.0のみ**

accept()

©2013-2019 Tencent Cloud. All rights reserved.

招待を受け取った後、このインターフェースを呼び出すと、現在の招待を受け入れます。

説明：

- 前回のinvitationの処理が未完了の場合、コンポーネントはデフォルトでビジー状態になり、その後のすべての招待はビジーを返します。
- v1.0.0以降のバージョンで、paramsパラメータを取り消しました。

```
import TRTCCalling from 'trtc-calling-js';
trtcCalling.on(TRTCCalling.EVENT.INVITED, ({inviteID, sponsor, inviteData}) => {
// ...
// v1.0.0以前
const { roomID, callType } = inviteData;
trtcCalling.accept({inviteID, roomID, callType})
// v1.0.0以降
trtcCalling.accept();
})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
inviteID	String	招待ID。1回の招待を標識します（イベントINVITEDコールバックデータのinviteIDを監視）。**v1.0.0より前のバージョンのみ**
roomID	Number	通話ルームナンバーID（イベントINVITEDコールバックデータのinviteData.roomIDを監視）。**v1.0.0より前のバージョンのみ**
callType	Number	1：音声通話、2：ビデオ通話（イベントINVITEDコールバックデータのinviteData.callTypeを監視）。**v1.0.0より前のバージョンのみ**

reject()

招待を受け取った後、このインターフェースを呼び出すと、現在の招待が拒否されます。

説明：
v1.0.0以降のバージョンで、paramsパラメータを取り消しました。

```
import TRTCCalling from 'trtc-calling-js';
trtcCalling.on(TRTCCalling.EVENT.INVITED, ({inviteID, sponsor, inviteData}) => {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// ...
// v1.0.0以前
const { callType } = inviteData;
trtcCalling.reject({inviteID, isBusy, callType})
// v1.0.0以降
trtcCalling.reject();
})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
inviteID	String	招待ID。1回の招待を識別します（イベントINVITEDコールバックデータのinviteIDを監視）。**v1.0.0より前のバージョンのみ**
isBusy	Boolean	回線ビジー状態の有無。**v1.0.0より前のバージョンのみ**
callType	Number	1：音声通話、2：ビデオ通話（イベントINVITEDコールバックデータのinviteData.callTypeを監視）。**v1.0.0より前のバージョンのみ**

hangup()

1. 通話中にこの関数を呼び出せば、通話を終了することができます。
2. 応答がない状態では、通話をキャンセルするために用いることができます。

```
trtcCalling.hangup()
```

ビデオ制御に関連するインターフェース関数

startRemoteView({userID, videoViewDomID})

リモートユーザーのカメラデータを指定されたDOM IDノードにレンダリングします。

```
trtcCalling.startRemoteView({userID, videoViewDomID})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	このユーザーデータは、このDOM IDノードにレンダリングされたvideoタグを介して再生されます

stopRemoteView({userID})

©2013-2019 Tencent Cloud. All rights reserved.

リモートユーザーのカメラデータによってレンダリングされたDOMノードを削除します。

説明：
v1.0.0以降のバージョンで、videoViewDomIDパラメータを削除しました。

```
// v1.0.0以前
trtcCalling.stopRemoteView({userID, videoViewDomID});
// v1.0.0以降
trtcCalling.stopRemoteView({userID});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	このDOM IDノードのvideoタグを削除し、ビデオ再生を停止します。**v1.0.0より前のバージョンのみ**

startLocalView({userID, videoViewDomID})

ローカルユーザーのカメラデータを指定されたDOM IDノードにレンダリングします。

```
trtcCalling.startLocalView({userID, videoViewDomID})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	ローカルユーザーデータは、このDOM IDノードにレンダリングされたvideoタグを介して再生されます

stopLocalView({userID})

ローカルユーザーのカメラデータによってレンダリングされたDOMノードを削除します。

説明：
v1.0.0以降のバージョンで、videoViewDomIDパラメータを削除しました。

©2013-2019 Tencent Cloud. All rights reserved.

```
// v1.0.0以前
trtcCalling.stopLocalView({userID, videoViewDomID});
// v1.0.0以降
trtcCalling.stopLocalView({userID});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	このDOM IDノードのvideoタグを削除し、ビデオ再生を停止します。 **v1.0.0より前のバージョンのみ**

openCamera()

ローカルカメラをオンにします。

```
trtcCalling.openCamera()
```

closeCamera()

カメラをオフにします。

```
trtcCalling.closeCamera()
```

setMicMute(isMute)

マイクをオン/オフします。

```
trtcCalling.setMicMute(true) // マイクオフ
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMute	Boolean	• true：マイクオフ • false：マイクオン

setVideoQuality(profile)

ビデオの画質を設定します。

©2013-2019 Tencent Cloud. All rights reserved.

説明：

- v0.8.0以降のバージョンは、このメソッドが新規追加されました。
- このメソッドは、call、groupCall、acceptの前に設定する必要があり、その後の設定は有効になりません。

```
trtcCalling.setVideoQuality('720p') // ビデオの画質を720pに設定します
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
profile	String	• 480p：640 × 480 • 720p：1280 × 720 • 1080p：1920 × 1080

switchToAudioCall()

ビデオ通話を音声通話へ切り替えます。

説明：

- v0.10.0以降のバージョンは、このメソッドが新規追加されました。
- 1v1通話中のみ使用をサポートします。
- ERRORイベントの監視に失敗しました。code：60001。

```
trtcCalling.switchToVideoCall() // ビデオ通話を音声通話へ切り替え
```

switchToVideoCall()

音声通話をビデオ通話へ切り替えます。

説明：

- v0.10.0以降のバージョンは、このメソッドが新規追加されました。
- 1v1通話中のみ使用をサポートします。
- ERRORイベントの監視に失敗しました。code：60002。

©2013-2019 Tencent Cloud. All rights reserved.

```
trtcCalling.switchToVideoCall() // 音声通話をビデオ通話へ切り替え
```

getCameras()

このインターフェースを呼び出して、カメラデバイスリストを取得することができます。

> 説明：
> v1.0.0以降のバージョンに、このメソッドが新規追加されました。

```
trtcCalling.getCameras() // カメラリストの取得
```

getMicrophones()

このインターフェースを呼び出して、マイクデバイスリストを取得することができます。

> 説明：
> v1.0.0以降のバージョンに、このメソッドが新規追加されました。

```
trtcCalling.getMicrophones() // マイクリストの取得
```

switchDevice({deviceType, deviceId})

このインターフェースを呼び出して、カメラまたはマイクのデバイスを切り替えることができます。

> 説明：
> v1.0.0以降のバージョンに、このメソッドが新規追加されました。

```
trtcCalling.switchDevice({deviceType: 'video', deviceId: deviceId}) // デバイスの切り替え
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
deviceType	String	video：カメラ, audio：マイク

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
deviceId	String	• カメラデバイスIDはgetCameras()で取得します • マイクデバイスIDはgetMicrophones()で取得します

TRTCCallingイベントリスト

次のコードを参照して、TRTCCalling コンポーネントイベントを監視できます。

```
import TRTCCalling from 'trtc-calling-js';
// etc
function handleInviteeReject({userID}) {

}
trtcCalling.on(TRTCCalling.EVENT.REJECT, handleInviteeReject)
```

招待側イベント

CODE	イベント受領側	説明
REJECT	招待側	被招待ユーザーは通話拒否
NO_RESP	招待側	被招待ユーザーはタイムアウト後に応答なし
LINE_BUSY	招待側	被招待ユーザーは通話中、ビジー状態
INVITED	被招待側	招待通知を受領済み
CALLING_CANCEL	被招待側	今回の通話をキャンセル済み
CALLING_TIMEOUT	被招待側	今回の通話はタイムアウト後に応答なし
USER_ENTER	招待側と被招待側	ユーザー入室
USER_LEAVE	招待側と被招待側	通話から離脱したユーザーあり
CALL_END	招待側と被招待側	今回の通話終了
KICKED_OUT	招待側と被招待側	ログインを繰り返したことにより、ルームから強制退室

©2013-2019 Tencent Cloud. All rights reserved.

CODE	イベント受領側	説明
USER_VIDEO_AVAILABLE	招待側と被招待側	リモートユーザーによるカメラのオン/オフ
USER_AUDIO_AVAILABLE	招待側と被招待側	リモートユーザーによるマイクのオン/オフ

一般的なイベントコールバック

SDK_READY

SDKがready状態に入るとこのコールバックを受信します

> 説明：
> v1.0.0以降のバージョンに、このイベントが新規追加されました。

```
let onSDKReady = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.SDK_READY, onSDKReady);
```

USER_ENTER

ユーザーが入室しました。
トリッガー：ユーザーが通話に参加する。

```
let handleUserEnter = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.USER_ENTER, handleUserEnter);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID

USER_LEAVE

ユーザーが退室しました。
トリッガー：ユーザーが通話を退出する。

©2013-2019 Tencent Cloud. All rights reserved.

```
let handleUserLeave = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.USER_LEAVE, handleUserLeave);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID

GROUP_CALL_INVITEE_LIST_UPDATE

グループチャットで招待リストを更新するとこのコールバックを受信します

> 説明：
> v1.0.0以降のバージョンに、このイベントが新規追加されました。

```
let handleGroupInviteeListUpdate = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.GROUP_CALL_INVITEE_LIST_UPDATE, handleGroupInviteeListUpdate);
```

CALL_END

今回の通話は終了しました。
トリッガー：今回の通話を終了する。

```
let handleCallingEnd = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.CALL_END, handleCallingEnd);
```

KICKED_OUT

ログインを繰り返したことにより、強制退室されました。
トリッガー：他のページに重複ログインする。

```
let handleKickedOut = function(event) {
console.log(event)
```

©2013-2019 Tencent Cloud. All rights reserved.

```
};
trtcCalling.on(TRTCCalling.EVENT.KICKED_OUT, handleKickedOut);
```

USER_VIDEO_AVAILABLE

リモートユーザーがカメラをオン/オフにします。

トリッガー：リモートユーザーがカメラをオン/オフにする。

```
let handleUserVideoChange = function({userID, isVideoAvailable}) {
console.log(userID, isVideoAvailable)
};
trtcCalling.on(TRTCCalling.EVENT.USER_VIDEO_AVAILABLE, handleUserVideoChange);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
isVideoAvailable	Boolean	• true：リモートユーザーによるカメラオン • false：リモートユーザーによるカメラオフ

USER_AUDIO_AVAILABLE

リモートユーザーがマイクをオン/オフにしました。

トリッガー：リモートユーザーがマイクをオン/オフする。

```
let handleUserAudioChange = function({userID, isAudioAvailable}) {
console.log(userID, isAudioAvailable)
};
trtcCalling.on(TRTCCalling.EVENT.USER_AUDIO_AVAILABLE, handleUserAudioChange);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
isAudioAvailable	Boolean	• true：リモートユーザーによるマイクオン • false：リモートユーザーによるマイクオフ

招待側イベントコールバック

REJECT

©2013-2019 Tencent Cloud. All rights reserved.

ユーザーが通話を拒否しました。
トリッガー：被招待者が通話を拒否し、発信者がREJECTイベントコールバックを受け取る。

```
let handleInviteeReject = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.REJECT, handleInviteeReject);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID

NO_RESP

招待されたユーザーは応答しませんでした。
トリッガー：call/groupCallにtimeoutを設定し、被招待者がtimeout内に応答しなかった場合、発信者がNO_RESPイベントコールバックを受け取る。

```
let handleNoResponse = function({userID, userIDList}) {
console.log(userID, userIDList)
};
trtcCalling.on(TRTCCalling.EVENT.NO_RESP, handleNoResponse);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
userIDList	Array	タイムアウトユーザーリスト

LINE_BUSY

被招待側は通話中で、ビジー状態です。
トリッガー：被招待者が通話中である場合、発信者がLINE_BUSYイベントコールバックを受け取る。

```
let handleLineBusy = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.LINE_BUSY, handleLineBusy);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userID	String	ユーザーID

被招待側イベントコールバック

INVITED

招待通知を受領しました。
トリッガー：招待通知があった場合、被招待者がINVITEDイベントコールバックを受け取る。

```
let handleNewInvitationReceived = function({
sponsor, userIDList, isFromGroup, inviteData, inviteID
}) {
console.log(sponsor, userIDList, isFromGroup, inviteData, inviteID)
};
trtcCalling.on(TRTCCalling.EVENT.INVITED, handleNewInvitationReceived);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sponsor	String	招待者
userIDList	Array	同時に招待された人
isFromGroup	Boolean	IMグループの招待の有無
inviteData	Object	新規ユーザーへの招待：{version, callType, roomID}
inviteID	String	招待ID、1回招待を識別

CALLING_CANCEL

今回の通話はキャンセルされました。
トリッガー：発信者がコール中に通話をキャンセルし、被招待者がCALLING_CANCELイベントコールバックを受け取る。

```
let handleCallingCancel = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.CALLING_CANCEL, handleCallingCancel);
```

CALLING_TIMEOUT

©2013-2019 Tencent Cloud. All rights reserved.

今回の通話はタイムアウトで、応答はありませんでした。
トリッガー：call/groupCallにtimeoutを設定し、被招待者がtimeout内に応答しなかった場合、被招待者がCALLING_TIMEOUTイベントコールバックを受け取る

```
let handleCallingTimeout = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.CALLING_TIMEOUT, handleCallingTimeout);
```

TRTCCalling エラーコードリスト

EVENTのERRORフィールドを監視することによって、コンポーネントからスローされたエラーを処理することができます。サンプルコードは次のとおりです。

```
import TRTCCalling from 'trtc-calling-js';
let onError = function(error) {
console.log(error)
};
trtcCalling.on(TRTCCalling.EVENT.ERROR, onError);
```

Error code

code	エラータイプ	意味
60001	メソッドのコールに失敗しました	switchToAudioCallのコールに失敗しました
60002	メソッドのコールに失敗しました	switchToAudioCallのコールに失敗しました
60003	権限の取得に失敗しました	使用可能なマイクデバイスがありません
60004	権限の取得に失敗しました	使用可能なカメラデバイスがありません
60005	権限の取得に失敗しました	ユーザーがデバイスの使用を禁止しています
60006	環境検出に失敗しました	現在の環境はWebRTC（>=v1.0.4バージョン）をサポートしていません

アップグレードガイド

©2013-2019 Tencent Cloud. All rights reserved.

- **TRTCCallingバージョンを>= 1.0.2にアップグレードします **
  - 注意：TSignalingバージョンを>= 0.9.0にアップグレードする必要があります
  - 原因：ログの更新
- **1.0.2にアップグレード > TRTCCallingバージョン>=1.0.0**
  - 注意：TSignalingバージョンを>= 0.8.0にアップグレードする必要があります
  - 原因：ログの更新

よくあるご質問

通話がつながらなかったり、強制オフラインになったりするのはなぜですか。

コンポーネントは現在、マルチインスタンスのログインや**オフラインプッシュのシグナリング**機能をサポートしていません。現在のログインアカウントの一意性をご確認ください。

> 説明：
>
> - **マルチインスタンス**：1つのUserIDで繰り返しログインしたり、異なるターミナルからログインしたりすると、シグナリングの混乱が生じます。
> - **オフラインプッシュ**：インスタンスはオンラインの場合にのみメッセージを受信できます。インスタンスがオフラインのときに受信したシグナリングは、オンラインになった後は再度プッシュされません。
>
> その他よくあるご質問については、TRTCCalling Webに関するご質問をご参照ください。

技術的なお問い合わせ

詳細については、お問い合わせにご連絡をいただくか、colleenyu@tencent.comにメールでご連絡ください。

参考ドキュメント

- TRTCCalling web公式サイトで体験
- TRTCCalling npm
- TRTCCalling web demoソースコード
- TRTCCalling web API
- TRTCCalling webに関するご質問

©2013-2019 Tencent Cloud. All rights reserved.

TRTCCalling API(Flutter)

最終更新日：：2021-10-12 09:59:48

TRTCCallingはTencent CloudのTRTCとIMを組み合わせたサービスで、1v1のオーディオビデオ通話をサポートします。TRTCCallingは、Tencent Cloudの2つのクローズドソースSDKに依存する1つのオープンソースのClassです。具体的な実装プロセスについては、Real-Time Video Call（Flutter）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

TRTCCalling API概要

SDK基本関数

API	説明
sharedInstance	コンポーネントシングルトン。
destroySharedInstance	コンポーネントシングルトンを廃棄します。
registerListener	イベントコールバックを追加します。
unRegisterListener	コールバックインターフェースを削除します。
destroy	関数の廃棄。このインスタンスを実行する必要がなくなった場合は、このインターフェースをコールしてください。
login	コンポーネントインターフェースへのログイン。すべての機能を使用するためには、まずログインする必要があります。
logout	コンポーネントインターフェースからログアウト。ログアウト後にダイヤル操作はできません。

通話操作に関連するインターフェース関数

API	説明
call	1対1通話に招待します。
accept	現在の通話を受信します。
reject	現在の通話を拒否します。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
hangup	現在の通話を終了します。

プッシュプルストリームに関連するインターフェース関数

API	説明
startRemoteView	リモートユーザーのカメラデータを指定のTXCloudVideoViewにレンダリングします。
stopRemoteView	リモートデータのレンダリングを停止します。

音声ビデオ制御に関連するインターフェース関数

API	説明
openCamera	カメラを起動し、指定のTXCloudVideoViewにレンダリングします。
switchCamera	フロント/リアカメラを切り替えます。
closeCamara	カメラを終了します。
setMicMute	ローカルオーディオキャプチャをミュートにします。
setHandsFree	ハンズフリーを設定します。

TRTCCallingDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。

招待者のコールバック

API	説明
onReject	通話拒否時のコールバック。
onNoResp	相手方の応答がない場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onLineBusy	通話中である場合のコールバック。

被招待者のコールバック

API	説明
onInvited	通話に招待された場合のコールバック。
onCallingCancel	現在の通話をキャンセルする場合のコールバック。
onCallingTimeOut	現在の通話がタイムアウトした場合のコールバック。

一般的なコールバック

API	説明
onUserEnter	ユーザーが通話に参加した場合のコールバック。
onUserLeave	ユーザーが通話から退出した場合のコールバック。
onUserAudioAvailable	ユーザーがオーディオアップストリームを開始したかどうかのコールバック。
onUserVideoAvailable	ユーザーがビデオアップストリームを開始したかどうかのコールバック。
onUserVoiceVolume	ユーザー通話音量のコールバック。
onCallEnd	通話終了時のコールバック。

SDK基本関数

sharedInstance

shareInstanceはTRTCCallingのコンポーネントシングルトンです。

```
static Future<TRTCCalling> sharedInstance();
```

destroySharedInstance

コンポーネントシングルトンを廃棄します。

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

©2013-2019 Tencent Cloud. All rights reserved.

destroy

関数の廃棄。このインスタンスを実行する必要がなくなった場合は、このインターフェースをコールしてください。

```
void destroy();
```

registerListener

TRTCCallingイベントコールバック。TRTCCallingDelegateを介してTRTCCallingの各種ステータス通知を受け取ることができます。

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

unRegisterListener

コールバックインターフェースを削除します。

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

login

コンポーネントにログインします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppID	int	TRTCコンソール > 【アプリケーション管理】> アプリケーション情報でSDKAppIDを確認できます。
user	String	現在のユーザーID、文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。

logout

コンポーネントからログアウトします。

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

©2013-2019 Tencent Cloud. All rights reserved.

通話操作に関連するインターフェース関数

call

1対1通話の招待、現在通話中でも引き続き他のユーザーの招待をコールできます。

```
Future<ActionCallback> call(String userId, int type);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーIDを呼び出します。
type	int	1は音声通話、2はビデオ通話を意味します。

accept

現在の通話を受信します。被招待者として `onInvited()` のコールバックを受け取った場合は、この関数を呼び出して通話に応答することができます。

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

reject

現在の通話を拒否します。被招待者として `onInvited()` のコールバックを受け取った場合は、この関数を呼び出して電話を拒否することができます。

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

hangup

現在の通話を終了します。通話中である場合は、この関数を呼び出して通話を終了できます。

```
void hangup();
```

プッシュプルストリームに関連するインターフェース関数

startRemoteView

©2013-2019 Tencent Cloud. All rights reserved.

リモートユーザーのカメラデータを指定のTRTCCloudVideoViewにレンダリングします。

```
void startRemoteView(String userId, int streamType, int viewId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	リモートユーザーID。
streamType	int	視聴を停止するuserIdのビデオストリームタイプを指定します。
viewId	int	ビデオ画像をロードするウィジェットTRTCCloudVideoViewがコールバックするviewId。

stopRemoteView

リモートデータのレンダリングを停止します。

```
void stopRemoteView(String userId, int streamType);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	リモートユーザーID。
streamType	int	視聴を停止するuserIdのビデオストリームタイプを指定します。

音声ビデオ制御に関連するインターフェース関数

openCamera

カメラを起動し、指定のTRTCCloudVideoViewにレンダリングします。

```
void openCamera(bool isFrontCamera, int viewId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFrontCamera	bool	trueはフロントカメラの起動を、falseはリアカメラの起動を意味します。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
viewId	int	ビデオ画像をロードするウィジェットTRTCCloudVideoViewがコールバックするviewId。

switchCamera

フロント/リアカメラを切り替えます。

```
void switchCamera(bool isFrontCamera);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFrontCamera	bool	trueはフロントカメラへの切り替えを、falseはリアカメラへの切り替えを意味します。

closeCamara

カメラをオフにします。

```
void closeCamera();
```

setMicMute

ローカルオーディオキャプチャをミュートにします。

```
void setMicMute(bool isMute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMute	bool	trueはマイクの終了を、falseはマイクの起動を意味します。

setHandsFree

リモートの音声をミュートにします。

```
void setHandsFree(bool isHandsFree);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
isHandsFree	bool	trueはハンズフリーの起動を、falseはハンズフリーの終了を意味します。

TRTCCallingDelegateイベントコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
msg	String	エラー情報。

招待者のコールバック

onReject

通話拒否時のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	拒否したユーザーのID。

onNoResp

©2013-2019 Tencent Cloud. All rights reserved.

相手方が応答しない場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	応答しないユーザーのID。

onLineBusy

通話中である場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話中のユーザーのID。

被招待者のコールバック

onInvited

通話に招待された時のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sponsor	String	発信者のID。
userIds	List&dxlt;String&dxgt;	自分以外の招待IDのリスト。
isFromGroup	bool	多人数の通話への招待かどうか。
type	int	1は音声通話、2はビデオ通話を意味します。

onCallingCancel

現在の通話がキャンセルされた場合のコールバック。被招待者がリクエストを処理しない場合、招待者がキャンセル後にこのコールバックを受け取ります。

onCallingTimeOut

現在の通話がタイムアウトした場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

一般的なコールバック

onUserEnter

ユーザーが通話に参加した場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話に参加したユーザーのID。

onUserLeave

ユーザーが通話から退出した場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話から離脱したユーザーのID。

onUserAudioAvailable

ユーザーがオーディオアップストリームを開始したかどうかのコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話するユーザーID。
available	boolean	ユーザーの音声が使用可能かどうか。

onUserVideoAvailable

ユーザーがビデオアップストリームを開始したかどうかのコールバック。通知を受け取った後、ユーザーはstartRemoteViewをコールしてリモートビデオをレンダリングすることができます。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話するユーザーID。
available	boolean	ユーザーのビデオが使用可能かどうか。

©2013-2019 Tencent Cloud. All rights reserved.

onUserVoiceVolume

ユーザーの通話音量のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userVolumes	List	ディスカッション中のルームの全参加者の音量、値の範囲：0～100。
totalVolume	int	すべてのリモート参加者の総音量。値の範囲：0～100。

onCallEnd

通話終了時のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

音声通話
音声通話(iOS)

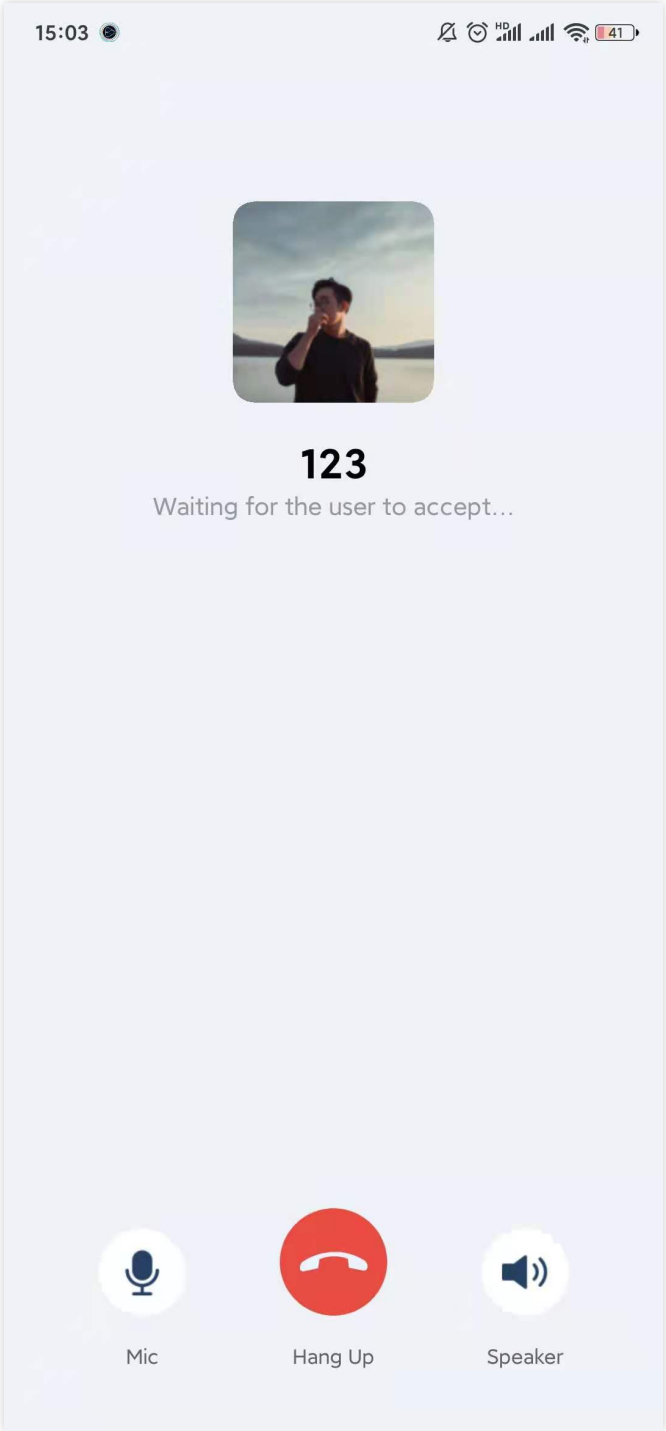
最終更新日：：2022-03-10 11:13:32

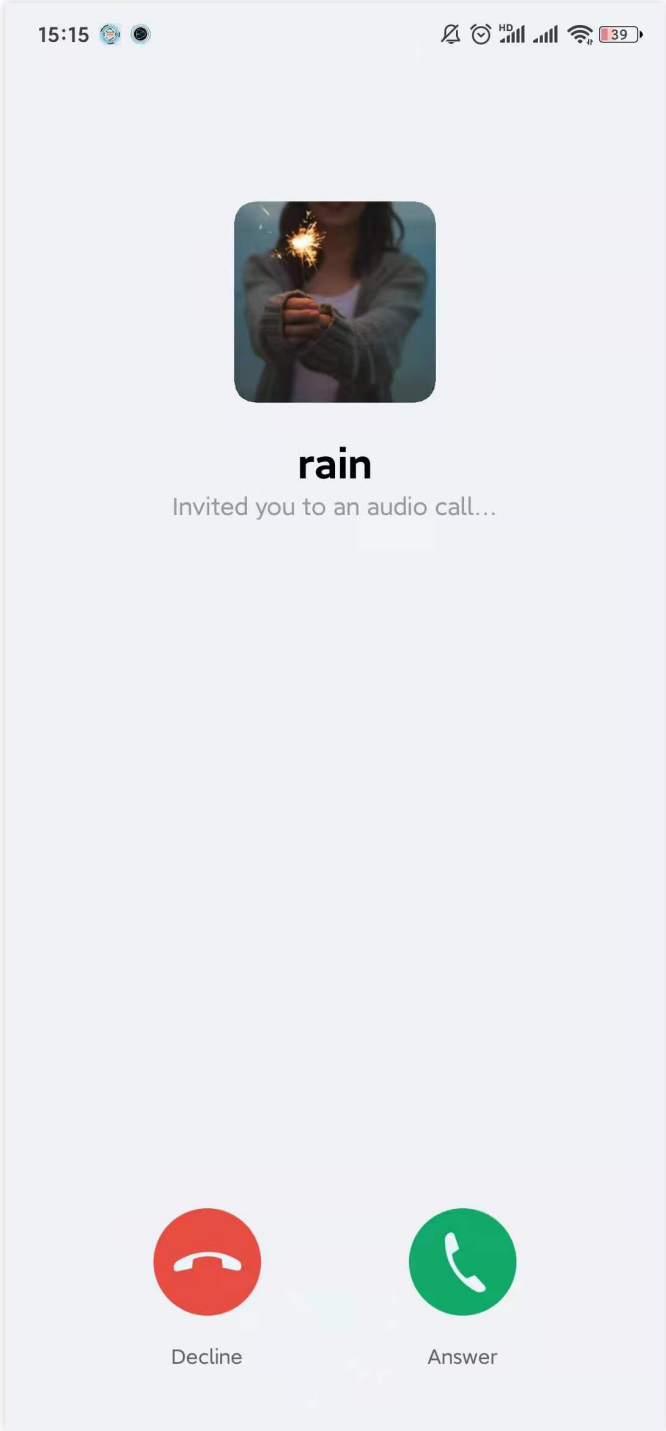
デモンストレーション

Appをダウンロードしてインストールすると、TRTC通話の効果を体験できます。

発呼側	着呼側

©2013-2019 Tencent Cloud. All rights reserved.





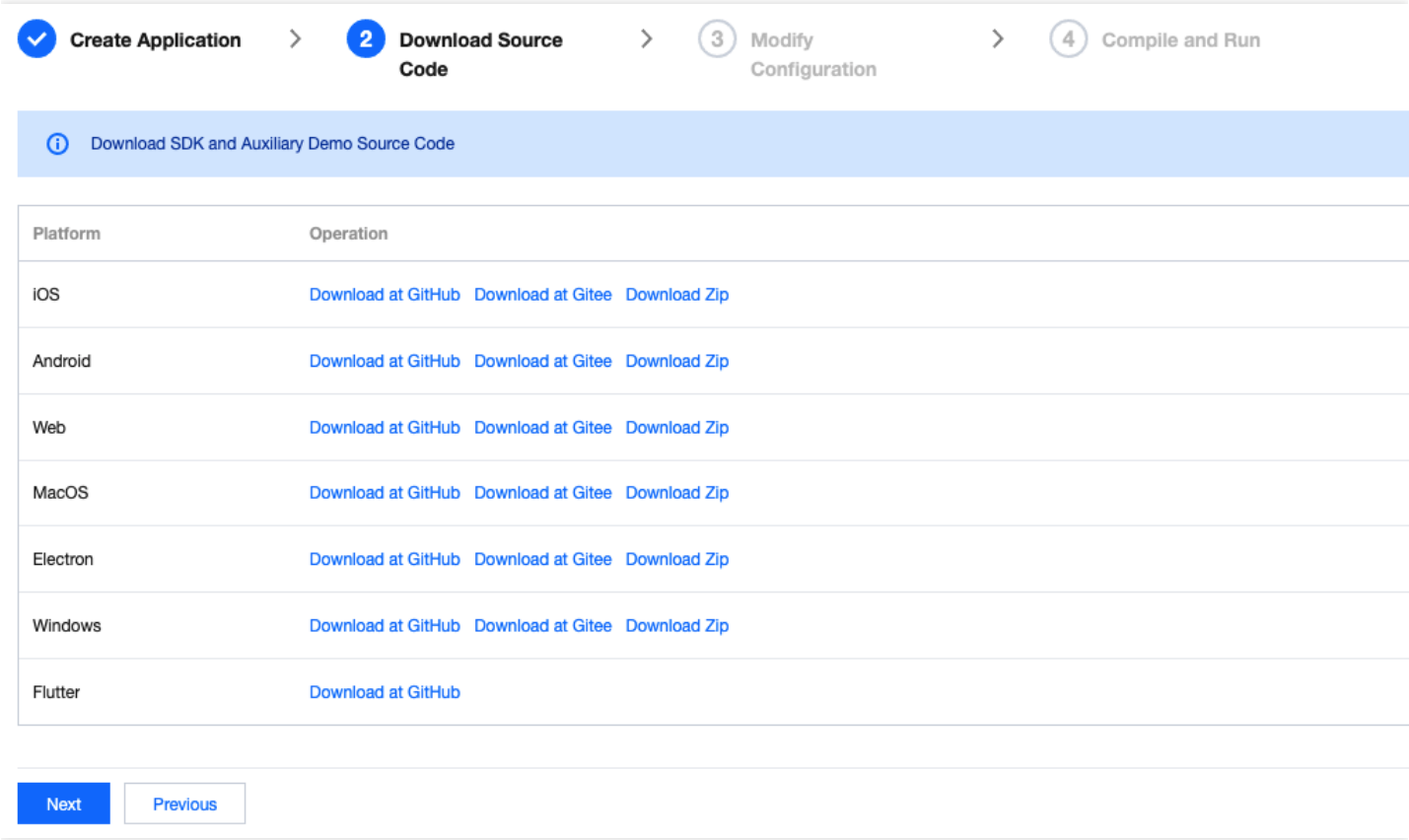
注意：
オーディオビデオ通話機能を素早く実装できるように、当社はTUICallingコンポーネントをリモデルし、通話UIをTUICallingコンポーネントの内部に実装しました。お客様がUIに気をつかう必要はありません。

©2013-2019 Tencent Cloud. All rights reserved.

Appの実行と体験

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援>Demoクイックスタート**を選択します。
   2. **アプリケーション名（例：** `TestVideoCall` **）を入力して、**作成**をクリックします。
2. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUICallingに進み、ソースコードをCloneまたはダウンロードします。

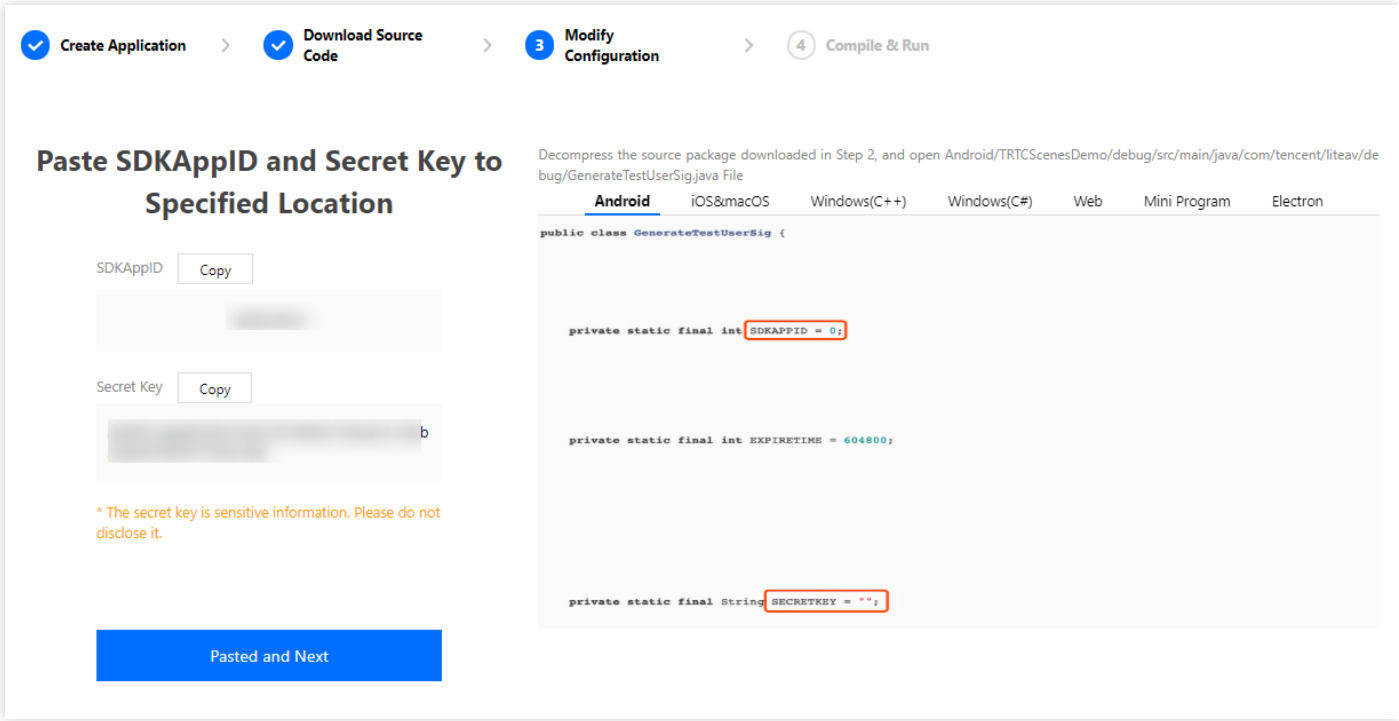
手順3：Appプロジェクトファイルの設定

©2013-2019 Tencent Cloud. All rights reserved.

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `iOS/Example/Debug/GenerateTestUserSig.swift` のファイルを見つけて開きます。

3. `GenerateTestUserSig.swift` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトは0。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトは空文字列。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

©2013-2019 Tencent Cloud. All rights reserved.

手順4：Appの実行

Xcode（バージョン11.0以上）を使用してソースコードプロジェクト `TUICalling/Example/TUICallingApp.xcworkspace` を開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. 電話をかけたいuserIdを入力し、検索をクリックします。
3. **呼出**をクリックし、**ビデオ通話**を選択して電話をかけます（**着呼者がアプリケーション内にいることを保証してください。保証できない場合、電話をかけても失敗する可能性があります**）。

ユーザーB

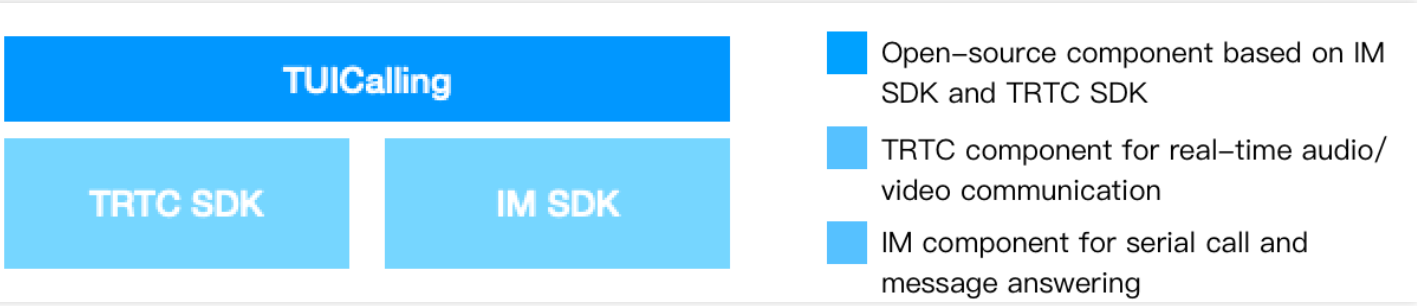
1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. メインページに進み、通話応答を待ちます。

具体的な接続フロー

ソースコードの `Source` フォルダの中にはui、model、Serviceの3つのサブフォルダが含まれています。そのうち、Serviceフォルダには外部に公開しているオープンソースコンポーネントTUICallingManagerが含まれています。 `TUICallingManager.h` ファイルの中でこのコンポーネントが提供するインターフェース関数を見つけるこ

©2013-2019 Tencent Cloud. All rights reserved.

とができます。



オープンソースコンポーネントTUICallingのTUICallingManagerをそのまま使用し、オーディオビデオ通話機能を手軽に実装することができます。もう複雑な通話UIやロジックの実装をご自身で行う必要はありません。

手順1：SDKの統合

通話コンポーネントTRTCCallingは、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

- **方法1：cocoapodsリポジトリを介する依存**

```
pod 'TXIMSDK_iOS'
pod 'TXLiteAVSDK_TRTC'
```

- **方法2：ローカルを介する依存**
  開発環境でのcocoapodsリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定

info.plistファイルに `Privacy - Camera Usage Description` 、 `Privacy - Microphone Usage Description` を追加し、カメラとマイクの権限を申請する必要があります。

手順3：TUICallingコンポーネントをインポート

cocoapodsによってコンポーネントをインポートします。具体的な手順については、以下のとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

1. プロジェクトディレクトリ下の `Source` 、 `Resources` 、 `TXAppBasic` フォルダ、 `TUICalling.podspec` ファイルをプロジェクトディレクトリ下へコピーします。
2. `Podfile` ファイル内に以下の依存関係を追加します。その後、 `pod install` コマンドを実行すると、インポートが完了します。

```
pod 'TXAppBasic', :path =&gt; "../TXAppBasic/"
pod 'TXLiteAVSDK_TRTC'
pod 'TUICalling', :path =&gt; "../", :subspecs =&gt; ["TRTC"]
```

手順4：コンポーネントの初期化およびログイン

1. `TUICallingManager.sharedInstance()` を呼び出し、コンポーネントを初期化します。
2. `TUILogin.initWithSdkAppID(SDKAPPID)` を呼び出し、ログインを初期化します。
3. `TUILogin.login(userId, userSig)` を呼び出し、コンポーネントのログインを完了します。このうちの重要なパラメータの入力については、下表をご参照ください。

パラメータ名	作用
sdkAppID	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。計算方法については UserSigの計算方法をご参照ください。

```
// コンポーネントの初期化
TUICallingManager.sharedInstance();
// ログイン
TUILogin.initWithSdkAppID(SDKAPPID)
TUILogin.login(userId, userSig) {
print("login success")
} fail: { code, errorDes in
print("login failed, code:¥(code), error: ¥(errorDes ?? "nil")")
}
```

手順5：オーディオビデオ通話の実装

1. 発信側：TUICallingManagerの `call();` メソッドを呼び出して通話リクエストを送信し、ユーザーIDの配列（userIDs）と通話タイプ（type）を渡します。通話タイプパラメータは `.audio` （オーディオ通話）または `.video` （ビデオ通話）を渡します。ユーザーID配列（userIDs）のuserIDが1つのみの時は、シングル通話と見なされ、ユーザーID配列（userIDs）のuserIDが複数（>=2）の時は多人数通話と見なされます。

©2013-2019 Tencent Cloud. All rights reserved.

2. 受信側：受信側がログイン状態の場合は、対応するインターフェースが自動的に起動します。受信側がログインしていない状態でも通話リクエストを受け取りたい場合は、オフライン応答をご参照ください。

```
//1. リスナーの登録
TUICallingManager.shareInstance().setCallingListener(listener: TUICallingListerner())

// 2. カスタムページ設定の有無（デフォルトはオフ）
TUICallingManager.shareInstance().enableCustomViewRoute(enable: true)

// 3. リスナーコールバックメソッドの実装
public func shouldShowOnCallView() -&gt; Bool {
return true;
}

public func callStart(userIDs: [String], type: TUICallingType, role: TUICallingRole, viewControll
er: UIViewController?) { if let vc = viewController {
callingVC = vc;
vc.modalPresentationStyle = .fullScreen

if var topController = UIApplication.shared.keyWindow?.rootViewController {
while let presentedViewController = topController.presentedViewController {
topController = presentedViewController
}

if let navigationVC = topController as? UINavigationController {
if navigationVC.viewControllers.contains(self) {
present(vc, animated: false, completion: nil)
}else{
navigationVC.popToRootViewController(animated: false)
navigationVC.pushViewController(self, animated: false)
navigationVC.present(vc, animated: false, completion: nil)
}
}else{
topController.present(vc, animated: false, completion: nil)
}
}
}
}

public func callEnd(userIDs: [String], type: TUICallingType, role: TUICallingRole, totalTime: Flo
at) {
callingVC.dismiss(animated: true, completion: nil)
}

public func onCallEvent(event: TUICallingEvent, type: TUICallingType, role: TUICallingRole, messa
ge: String) {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
}
// 4.電話をかけます
TUICallingManager.shareInstance().call(userIDs, .video)
```

手順6：オフライン応答の実装

説明：
ビジネスの位置付けがオンラインカスタマーサービスなどのオフライン応答機能を必要としないシーンである場合は、上記[手順1]（＃model.step1）-[手順5]（＃model.step5）の対応で問題ありません。 しかし、ビジネスの位置付けがソーシャルシーンである場合は、オフライン応答を実装することをお勧めします。

IM SDKはオフラインプッシュをサポートしていますが、使用可能な基準を満たすためには相応の設定を行う必要があります。

1. Appleプッシュ証明書を申請する場合の具体的な操作については、Appleプッシュ証明書の申請をご参照ください。
2. バックエンドおよびクライアントがオフラインプッシュを設定する場合の具体的な操作については、オフラインプッシュ（iOS）をご参照ください。
3. 現在、TRTCCallingImplのsendModelシグナリング送信関数にオフライン送信関数が統合されており、Appのオフラインプッシュを設定した後、メッセージのオフラインプッシュを実装することができます。

コンポーネントAPIリスト

TUICallingコンポーネントのAPIインターフェースリストは次のとおりです。

インターフェース関数	インターフェースの機能
call	C2C通話に招待します
receiveAPNSCalled	被招待者として着信に応答します
setCallingListener	リスナーを設定します
setCallingBell	着信音の設定(30s以内を推奨)
enableMuteMode	ミュートモードをオンにします

©2013-2019 Tencent Cloud. All rights reserved.

インターフェース関数	インターフェースの機能
enableCustomViewRoute	カスタムビューをオンにします。オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewのインスタンスを受信します。開発者自身が表示方式を決定します。注意：フルスクリーンまたはスクリーンと同じ比率で表示する必要があります。そうしない場合、表示に異常が生じます

©2013-2019 Tencent Cloud. All rights reserved.

音声通話(Android)

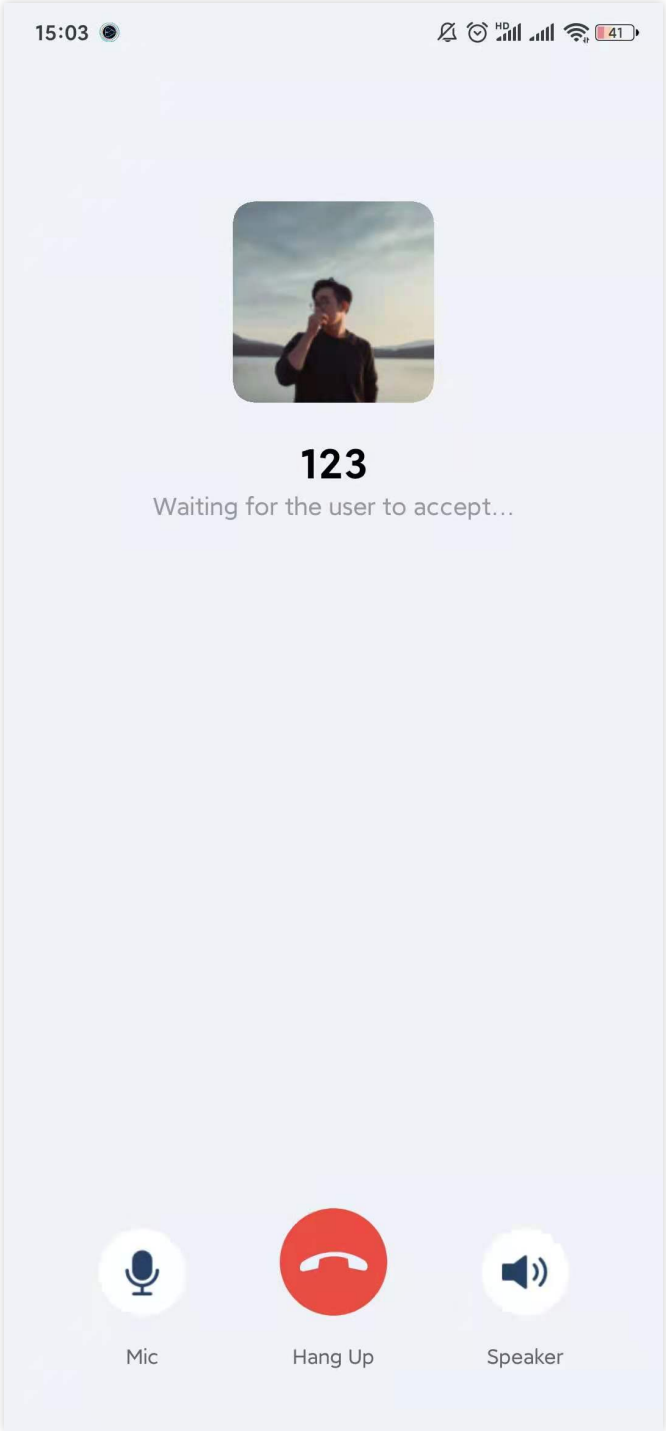
最終更新日：：2022-03-10 11:17:30

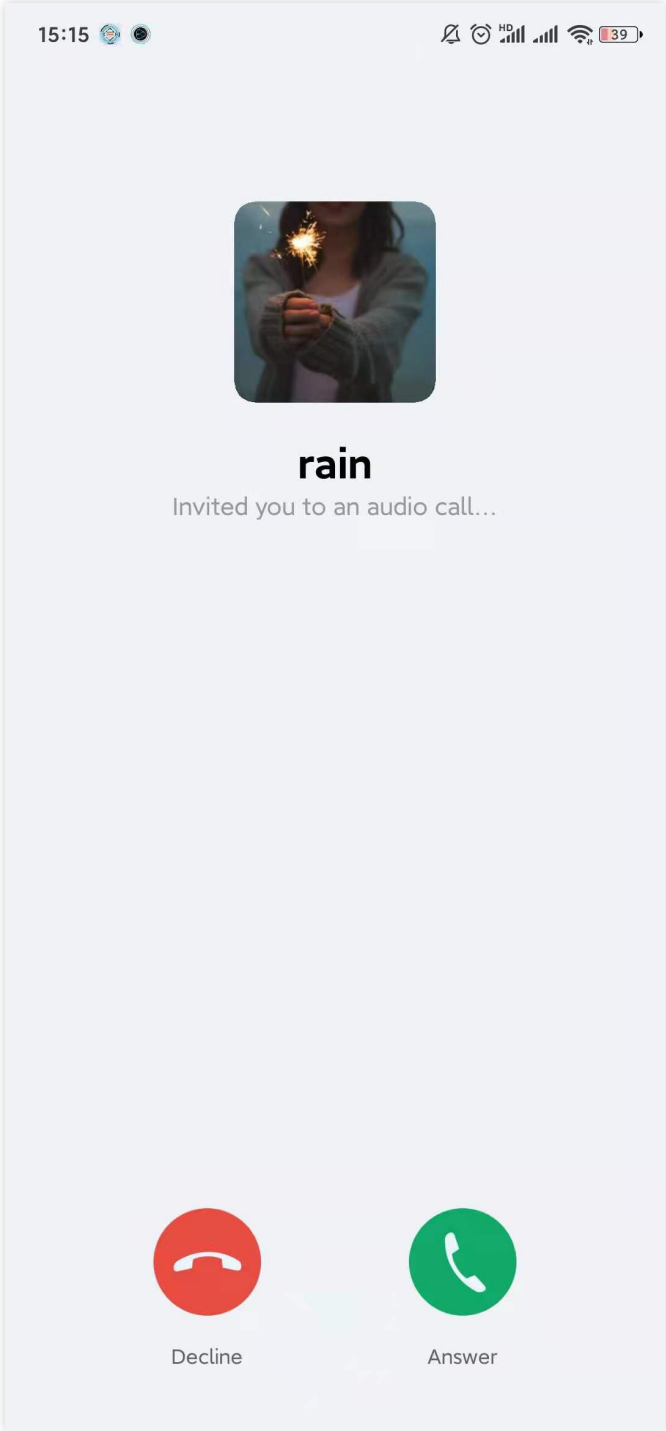
デモンストレーション

Appをダウンロードしてインストールすると、TRTC通話の効果を体験できます。

発呼側	着呼側

©2013-2019 Tencent Cloud. All rights reserved.





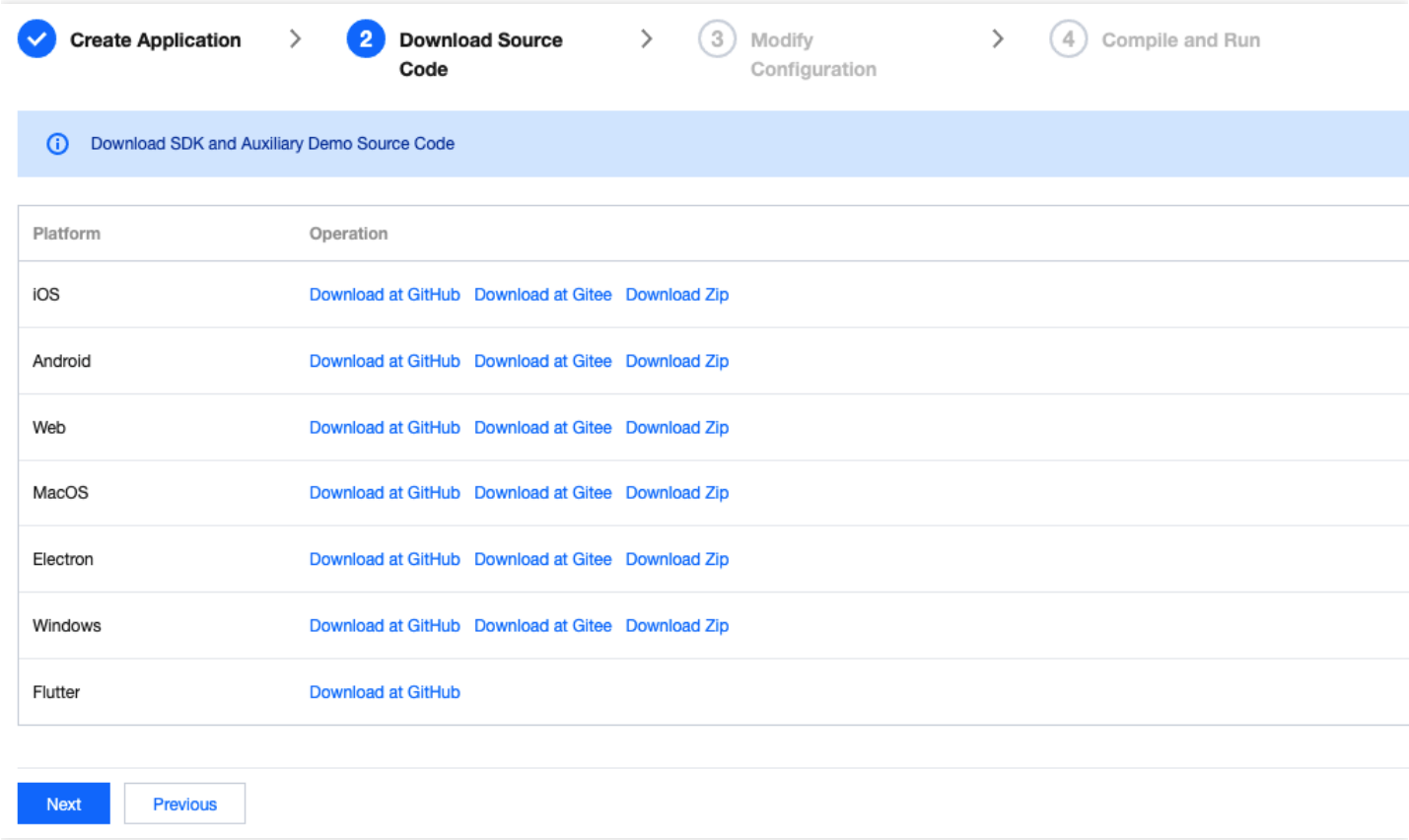
注意：
オーディオビデオ通話機能を素早く実装できるように、当社はTUICallingコンポーネントをリモデルし、通話UIをTUICallingコンポーネントの内部に実装しました。お客様がUIに気をつかう必要はありません。

©2013-2019 Tencent Cloud. All rights reserved.

Appの実行と体験

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援>Demoクイックスタート**を選択します。
   2. **アプリケーション名（例：** `TestVideoCall` **）を入力して、**作成**をクリックします。
2. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。



注意：
本機能はTencent CloudTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの価格説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUICallingに進み、ソースコードをCloneまたはダウンロードします。

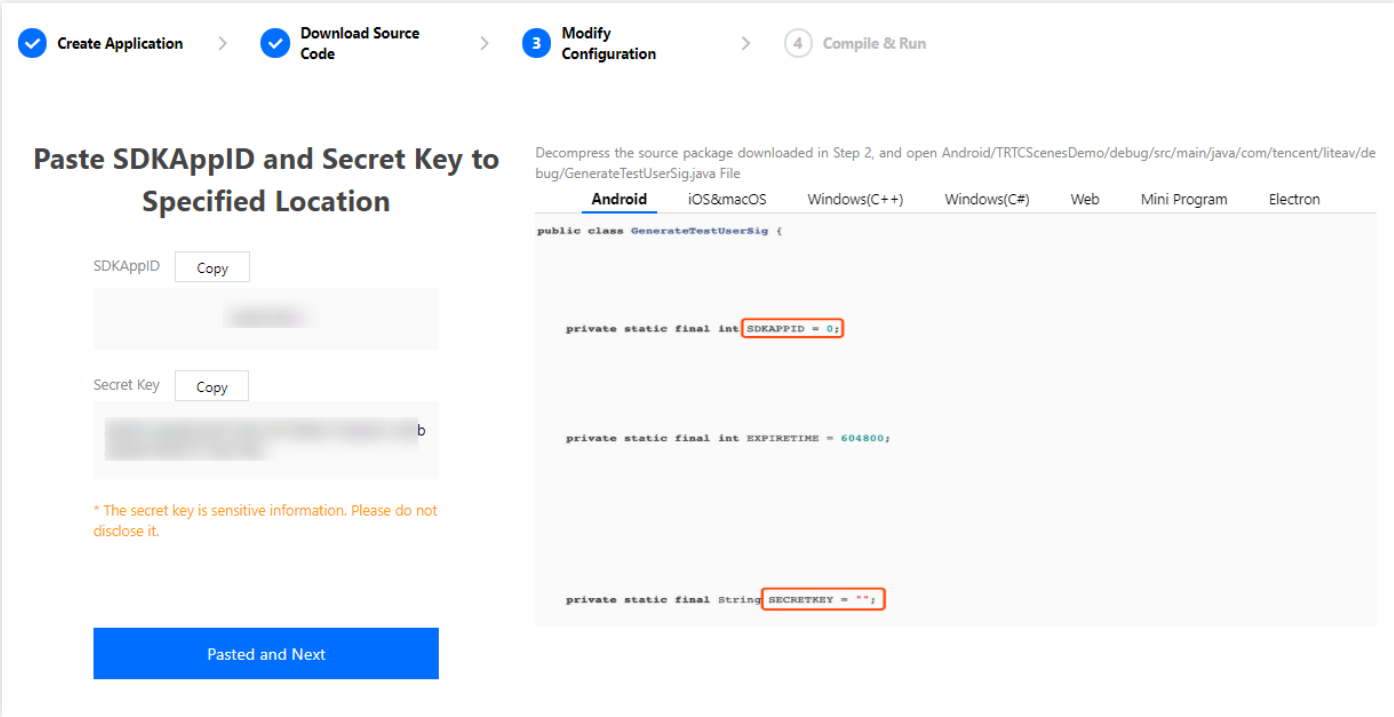
手順3：Appプロジェクトファイルの設定

©2013-2019 Tencent Cloud. All rights reserved.

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `Android/Debug/src/main/java/com/tencent/liteav/debug/GenerateTestUserSig.java` のファイルを見つけて開きます。

3. `GenerateTestUserSig.java` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはプレースホルダー(PLACEHOLDER)。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトはプレースホルダー(PLACEHOLDER)。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを

©2013-2019 Tencent Cloud. All rights reserved.

取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Android Studio（バージョン3.5以降）を使用してソースコードプロジェクト `TUICalling` を開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

体験アプリケーション

注意：
体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. 電話をかけたいuserIdを入力し、検索をクリックします。
3. **呼出**をクリックし、**ビデオ通話**を選択して電話をかけます（**着呼者がアプリケーション内にいることを保証してください。保証できない場合、電話をかけても失敗する可能性があります**）。

ユーザーB

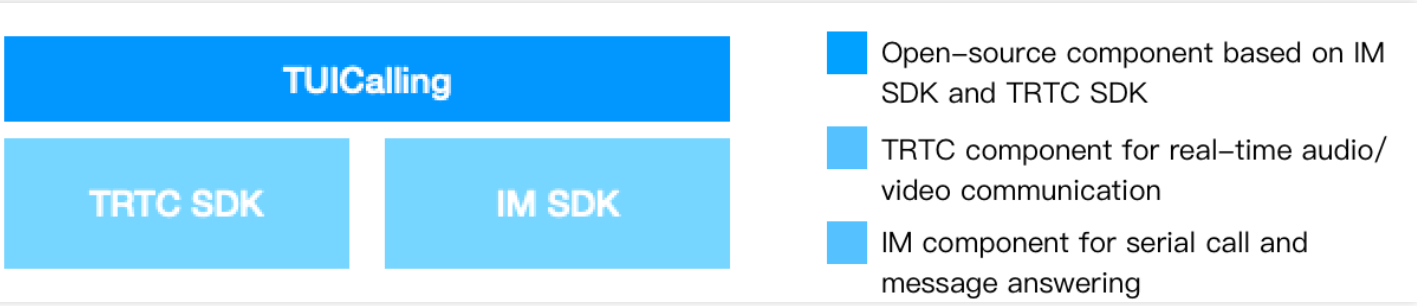
1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. メインページに進み、通話応答を待ちます。

具体的な接続フロー

ソースコードの `Source` フォルダの中にはuiとmodelの2つのサブフォルダが含まれています。そのうち、modelフォルダには外部に公開しているオープンソースコンポーネントTUICallingManagerが含まれています。 `TUICalling.java` ファイルの中でこのコンポーネントが提供するインターフェース関数を見つけることがで

©2013-2019 Tencent Cloud. All rights reserved.

きます。



オープンソースコンポーネントTUICallingのTUICallingManagerをそのまま使用し、オーディオビデオ通話機能を手軽に実装することができます。もう複雑な通話UIやロジックの実装をご自身で行う必要はありません。

手順1：SDKの統合

オーディオビデオ通話コンポーネントTUICallingは、TRTC SDKとIM SDKに依存しています。次の手順で2つのSDKをプロジェクトに統合することができます。

方法1：MavenCentralリポジトリを介する依存

1. dependenciesにTRTCSDKとIMSDKの依存を追加します。

```
dependencies {
compile "com.tencent.liteav:LiteAVSDK_TRTC:latest.release"
compile 'com.tencent.imsdk:imsdk:latest.release'

// 当社はgson解析を使用していますので、さらにgoogle のGsonに依存する必要はありません。
compile 'com.google.code.gson:gson:latest.release'
}
```

2. defaultConfigでAppが使用するCPUアーキテクチャを指定します。

```
defaultConfig {
ndk {
abiFilters "armeabi-v7a"
}
}
```

3. **Sync Now**をクリックし、SDKを同期します。

説明：

> mavenCentralへのネットワーク接続に問題がない場合、SDKは自動的にダウンロードされ、プロジェクトに統合されます。

方法2：ローカルAARを介する依存

開発環境でのmavenCentralリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定および難読化ルール

AndroidManifest.xmlにAppの権限を設定します。SDKには次の権限が必要です（6.0以上のAndroidシステムではカメラ、読み取りストレージの権限を動的に申請する必要があります）。

```
<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" />
```

proguard-rules.proファイルでは、SDK関連を非難読化リストに追加します。

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

手順3：TUICallingコンポーネントをインポート

Sourceディレクトリをプロジェクトにコピーし、 `setting.gradle` へのインポートを完了します。以下をご参照ください。

```
include ':Source'
```

©2013-2019 Tencent Cloud. All rights reserved.

手順4：コンポーネントの初期化およびログイン

1. `TUICallingManager.sharedInstance()` を呼び出し、コンポーネントを初期化します。
2. `TUILogin.init(context, SDKAppID, config, listener)` を呼び出し、ログインを初期化します。
3. `TUILogin.login(userId, userSig, callback)` を呼び出し、コンポーネントのログインを完了します。このうちの重要なパラメータの入力については、下表をご参照ください。

パラメータ名	作用
SDKAppID	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。計算方法については UserSigの計算方法をご参照ください。
config	SDKの設定。ログレベルおよびログコールバック（nullを渡すことも可能）の設定に使用します。詳細については、下記サンプルコードをご参照ください。
listener	IMリスナー。いくつかの必要なシステムコールバック通知の受信に使用します。例：キックアウトによるオフライン、userSig期限切れなど。詳細については、下記サンプルコードをご参照ください。
callback	ログイン結果リスナー。ログイン成功の有無を通知します。詳細については、下記サンプルコードをご参照ください。

```
// コンポーネントの初期化
TUICallingManager manager = TUICallingManager.sharedInstance();
// ログイン
V2TIMSDKConfig config = new V2TIMSDKConfig();
config.setLogLevel(V2TIMSDKConfig.V2TIM_LOG_DEBUG);
config.setLogListener(new V2TIMLogListener() {
@Override
public void onLog(int logLevel, String logContent) {

}
});
TUILogin.init(this, ${あなたのSDKAPPID}, config, new V2TIMSDKListener() {
@Override
public void onKickedOffline() { // ログインがキックアウトされたオフライン通知
mIsKickedOffline = true;
checkUserStatus();
}
@Override
```

©2013-2019 Tencent Cloud. All rights reserved.

```
public void onUserSigExpired() { // suerSig期限切れ通知
mIsUserSigExpired = true;
checkUserStatus();
}
});
TUILogin.login("${あなたのuserId}", "${あなたのuserSig}", new V2TIMCallback() {
@Override
public void onError(int code, String msg) {
Log.d(TAG, "code: " + code + " msg:" + msg);
}
@Override
public void onSuccess() {
Log.d(TAG, "onSuccess");
}
});
```

手順5：オーディオビデオ通話の実装

1. 発信側：TUICallingManagerの `call();` メソッドを呼び出して通話リクエストを送信し、ユーザーIDの配列（userids）と通話タイプ（type）を渡します。通話タイプパラメータは `TUICalling.Type.AUDIO` （オーディオ通話）または `TUICalling.Type.VIDEO` （ビデオ通話）を渡します。ユーザーID配列（userids）のuserIdが1つのみの時は、シングル通話と見なされ、ユーザーID配列（userids）のuserIdが複数（>=2）の時は多人数通話と見なされます。
2. 受信側：受信側がログイン状態の場合は、対応するインターフェースが自動的に起動します。受信側がログインしていない状態でも通話リクエストを受け取りたい場合は、オフライン応答をご参照ください。

```
// 1. コンポーネントの初期化
TUICallingManager manager = TUICallingManager.sharedInstance();
// 2. リスナーの登録
manager.setCallingListener(new TUICalling.TUICallingListener() {
@Override
public boolean shouldShowOnCallView() {
return true;
}

@Override
public void onCallStart(String[] userIDs, TUICalling.Type type, TUICalling.Role role, final View
tuiCallingView) {
if (!shouldShowOnCallView() || null == tuiCallingView) {
return;
}
runOnUiThread(new Runnable() {
@Override
public void run() {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
FrameLayout.LayoutParams params = new FrameLayout.LayoutParams(FrameLayout.LayoutParams.MATCH_PAR
ENT, FrameLayout.LayoutParams.MATCH_PARENT);
mCallingView = tuiCallingView;
addContentView(tuiCallingView, params);
}
});
}

@Override
public void onCallEnd(String[] userIDs, TUICalling.Type type, TUICalling.Role role, long totalTim
e) {
removeView();
}

@Override
public void onCallEvent(TUICalling.Event event, TUICalling.Type type, TUICalling.Role role, Strin
g message) {
if (TUICalling.Event.CALL_FAILED == event) {
removeView();
}
}
});
// 3.電話をかけます
manager.call(userIDs, TUICalling.Type.VIDEO);
```

手順6：オフライン応答の実装

説明：
ビジネスの位置付けがオンラインカスタマーサービスなどのオフライン応答機能を必要としないシーンである場合は、上記[手順1]（＃model.step1）-[手順5]（＃model.step5）の対応で問題ありません。 しかし、ビジネスの位置付けがソーシャルシーンである場合は、オフライン応答を実装することをお勧めします。

IM SDKはオフラインプッシュをサポートしていますが、Androidの各スマートフォンメーカーは各自のオフラインプッシュサービスを備えていることから、iOSプラットフォームよりアクセスが複雑となり、利用基準を満たすためには、相応の設定を行う必要があります。

1. 対応するメーカーのプッシュチャネルに必要な証明書などを申請の上、IMコンソール内に設定し、プッシュ要件に従って証明書とIDなどを追加します。詳細な操作手順については、IM > オフラインプッシュ（Android）をご参照ください。
2. 現在、TRTCCallingImplのsendModelシグナリング送信関数にオフライン送信関数が統合されており、Appのオフラインプッシュを設定した後、メッセージのオフラインプッシュを実装することができます。

©2013-2019 Tencent Cloud. All rights reserved.

コンポーネントAPIリスト

TUICallingコンポーネントのAPIインターフェースリストは次のとおりです。

インターフェース関数	インターフェースの機能
call	C2C通話に招待します
receiveAPNSCalled	被招待者として着信に応答します
setCallingListener	リスナーを設定します
setCallingBell	着信音の設定(30s以内を推奨)
enableMuteMode	ミュートモードをオンにします
enableCustomViewRoute	カスタムビューをオンにします。オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewのインスタンスを受信します。開発者自身が表示方式を決定します。注意：フルスクリーンまたはスクリーンと同じ比率で表示する必要があります。そうしない場合、表示に異常が生じます

©2013-2019 Tencent Cloud. All rights reserved.

音声通話(Web)

最終更新日：：2022-04-02 16:27:03

このドキュメントは2つのパートから成り、ブラウザで実行できる一連の音声通話ソリューションを実装する方法について説明します。

- パート1：サービスをアクティブ化する方法および提供するDemoのクイックスタートについて説明します。
- パート2：TRTCCallingコンポーネントを使用して、独自の音声通話機能をすばやく構築する方法について説明します。

環境要件

最新バージョンのChromeブラウザをご使用ください。現在、デスクトップのChromeブラウザはTRTC Web SDKをサポートしており、関連機能は比較的整っていますので、Chromeブラウザを使用して体験することをお勧めします。

- TRTCCallingは以下のポートに依存してデータ送信を行っていますので、これらをファイアウォール・ホワイトリストに追加してください。設定が完了したら、公式サイト Demoにアクセスして体験していただけば、設定が有効かどうかをチェックすることができます。
  - TCPポート：8687
  - UDPポート：8000、8080、8800、843、443、16285
  - ドメイン名：qcloud.rtc.qq.com
    具体的には、ファイアウォール制限の対処関連をご参照ください。

サポートプラットフォーム

現在、このソリューションは、次のプラットフォームをサポートしています。

OS	ブラウザタイプ	ブラウザの最小バージョン要件
Mac OS	デスクトップ版Safariブラウザ	11+
Mac OS	デスクトップ版Chromeブラウザ	56+
Mac OS	デスクトップ版Firefoxブラウザ	56+
Mac OS	デスクトップ版Edgeブラウザ	80+
Windows	デスクトップ版Chromeブラウザ	56+

©2013-2019 Tencent Cloud. All rights reserved.

OS	ブラウザタイプ	ブラウザの最小バージョン要件
Windows	デスクトップ版QQブラウザ（超高速コア）	10.4+
Windows	デスクトップ版Firefoxブラウザ	56+
Windows	デスクトップ版Edgeブラウザ	80+

説明：
互換性照会の詳細については、ブラウザサポート状況をご参照ください。また、TRTC検査ページでオンライン検査することも可能です。

URLドメイン名プロトコルの制限

ユースケース	プロトコル	受信（再生）	送信（マイク・オン）	画面共有	備考
本番環境	HTTPSプロトコル	サポートあり	サポートあり	サポートあり	推奨
本番環境	HTTPプロトコル	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	http://localhost	サポートあり	サポートあり	サポートあり	推奨
ローカル開発環境	http://127.0.0.1	サポートあり	サポートあり	サポートあり	-
ローカル開発環境	http://[ローカルマシンIP]	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	file:///	サポートあり	サポートあり	サポートあり	-

Demoのクイックスタートテスト

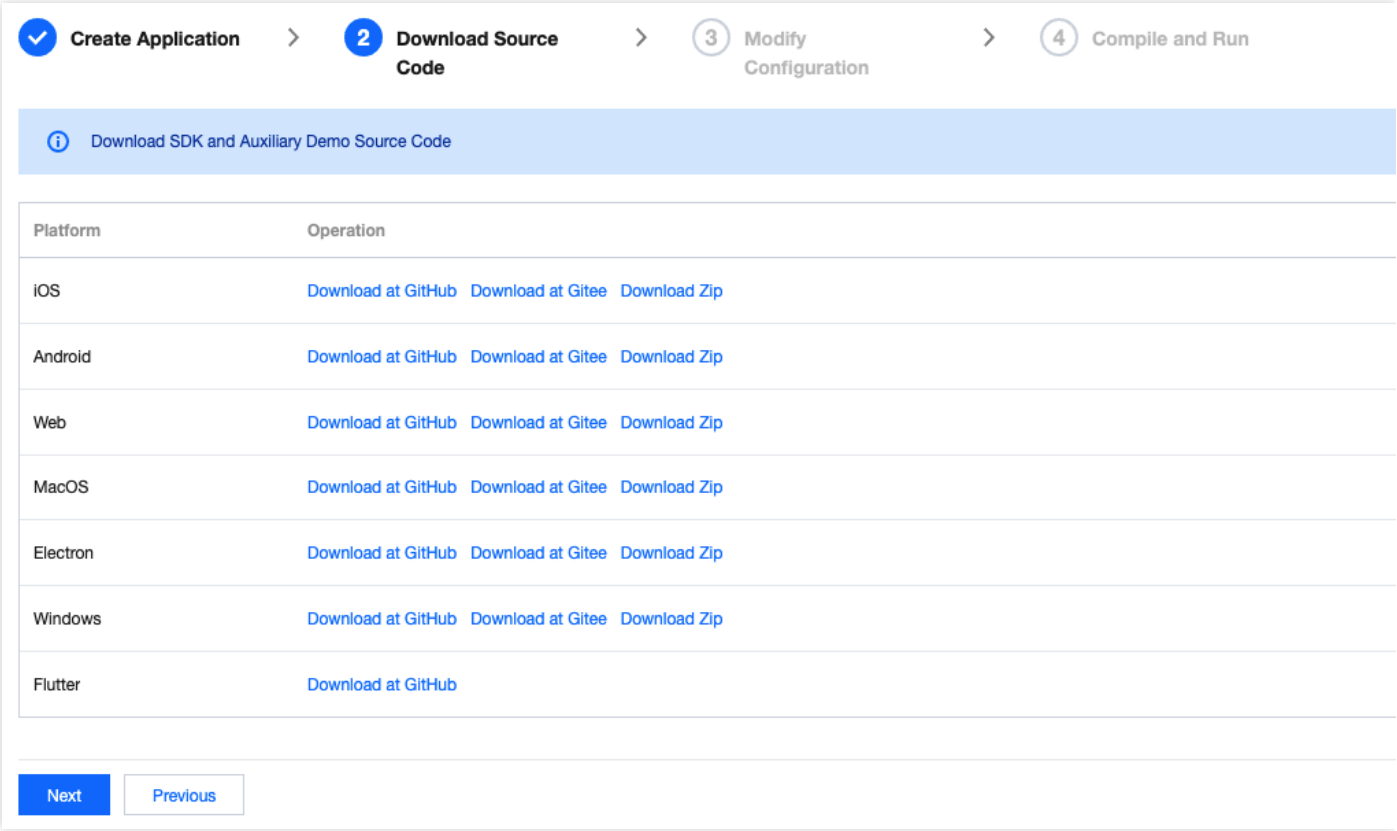
手順1：アプリケーションの新規作成

©2013-2019 Tencent Cloud. All rights reserved.

1.Tencent Cloudアカウントの登録を行い、実名認証を完了します。
2. TRTCコンソールにログインし、**開発支援** > **Demoクイックスタート**を選択します。
3. アプリケーション名（例： `TestTRTC` ）を入力し、**作成**をクリックします。

手順2：Demoのダウンロード

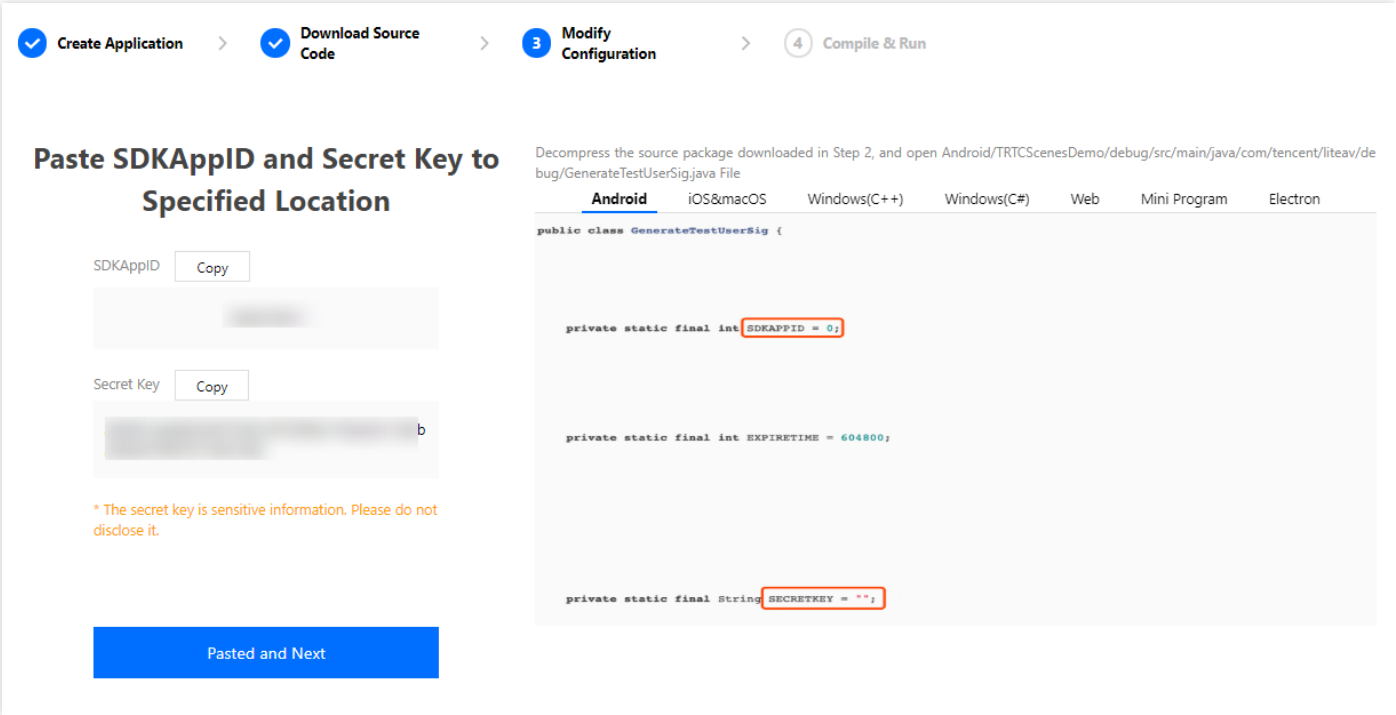
1. 実際のサービスニーズに応じてDemoソースコードをダウンロードするか、または、クリックしてTUICallingに進み、Demoソースコードをダウンロードします。
2. ダウンロード完了後、**ダウンロード完了、次のステップへ**をクリックします。



ステップ3：Demo プログラムファイルの設定

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。
2. `Web/public/debug/GenerateTestUserSig.js` ファイルを見つけて開きます。
3. `GenerateTestUserSig.js` のファイルの関連するパラメータを設定します。
   - SDKAPPID：デフォルトでは0です。実際のSDKAppIDに設定してください。
   - SECRETKEY：デフォルトでは空文字列です。実際のキー情報に設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、**貼り付け完了、次のステップへ**をクリックすると、作成が完了します。
5. コンパイル完了後、**コンソール概要に戻る**をクリックすると、終了します。

注意：

- ここで言及したUserSigの作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appからサービスサーバーにリクエストを発出し動的にUserSigを取得します。詳細はサーバーでのUserSig作成をご参照ください。

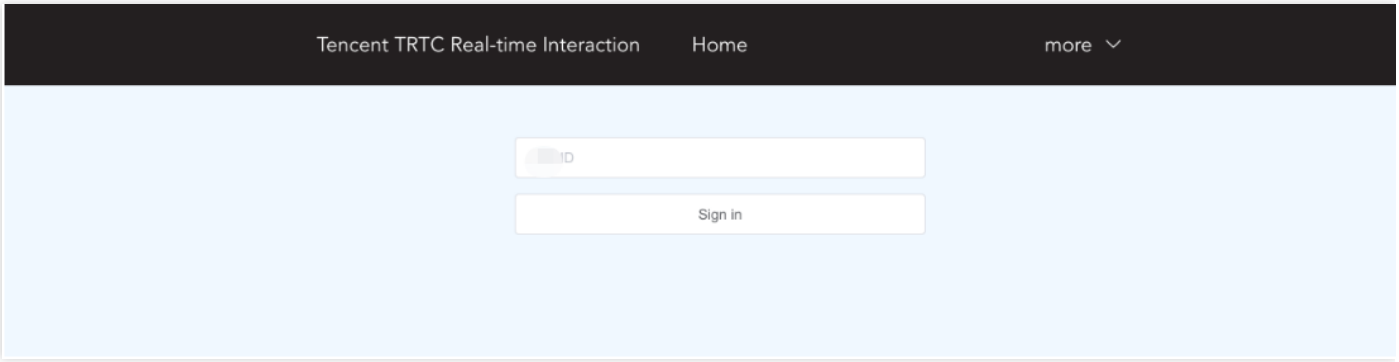
手順4：Demoの実行

1. Demoディレクトリに次のコマンドを順番に入力します。

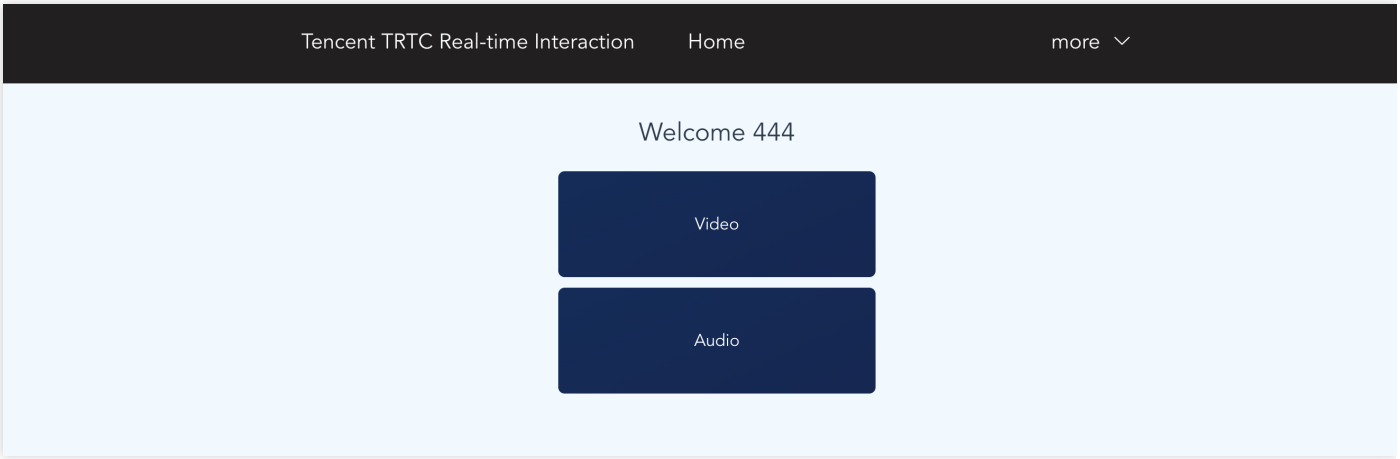
```
npm install
npm run serve
```

©2013-2019 Tencent Cloud. All rights reserved.

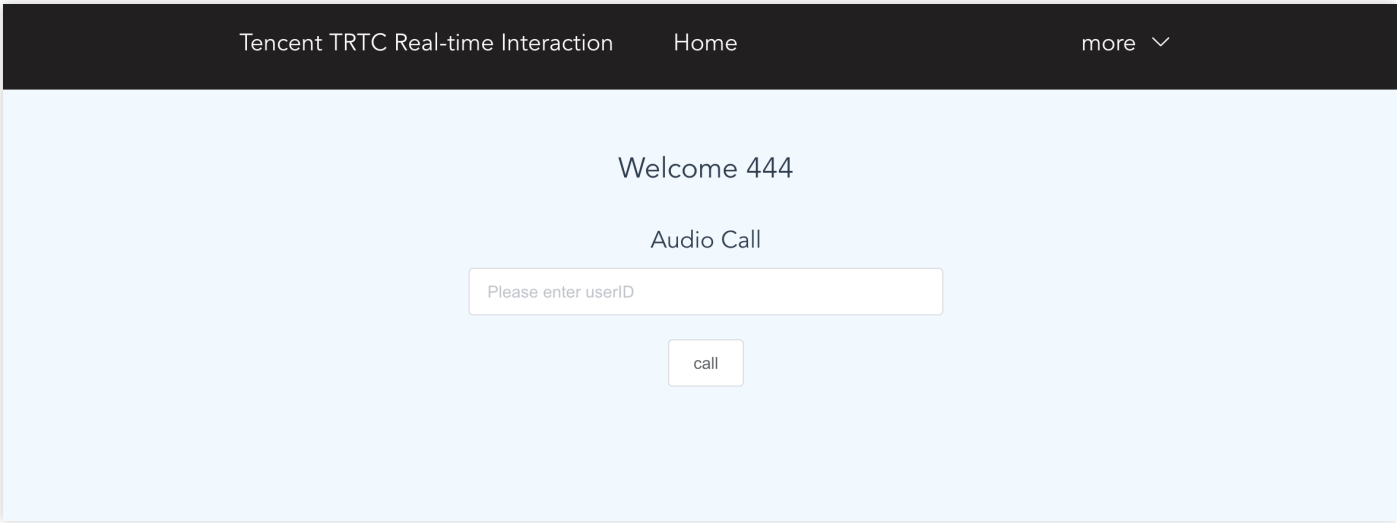
2. Chromeブラウザを起動し、リンク `http://localhost:8080/` を開きます。すべてが正常であれば、Demo実行インターフェースは次の図のようになります。



3. useridを入力し、**ログイン**をクリックして、**音声通話**を選択します。

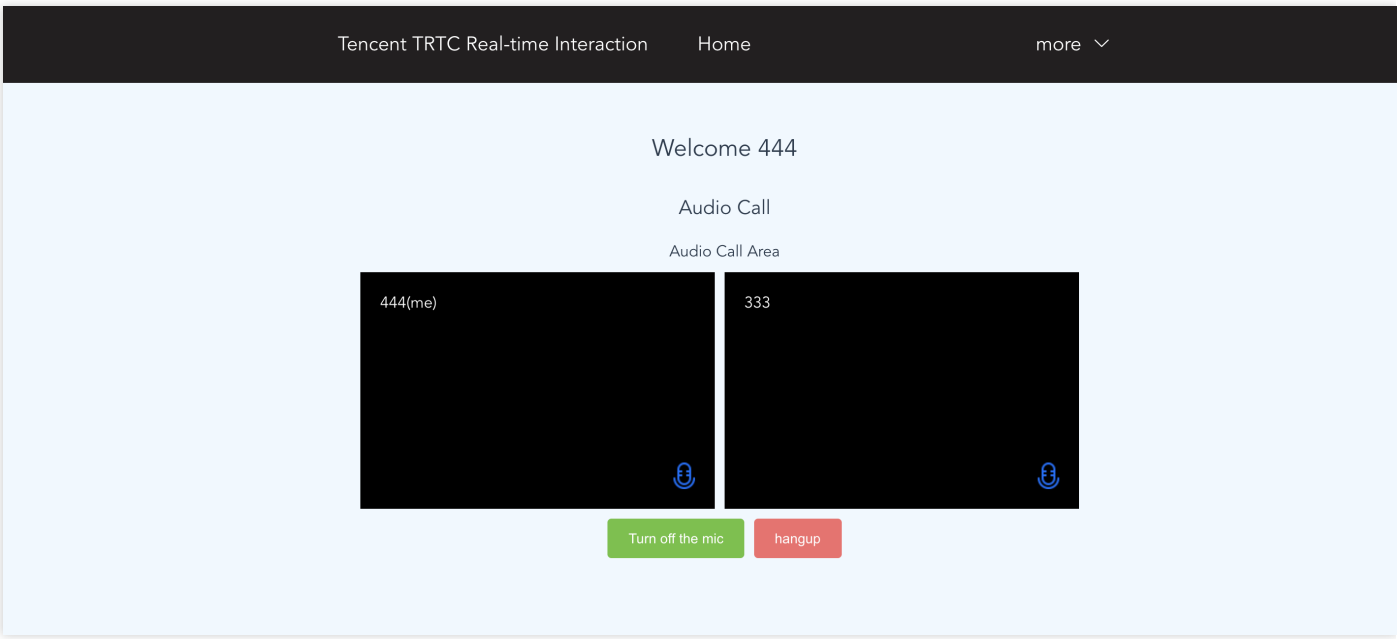


4. 呼び出し先のuseridを入力し、**呼出**をクリックします。



©2013-2019 Tencent Cloud. All rights reserved.

5. 音声通話を開始できます。



独自の音声通話の作成

手順1：TRTCCallingコンポーネントの統合

説明：

- 0.6.0以降は、trtc-js-sdk、tim-js-sdkおよびtsignalingの依存関係を手動でインストールする必要があります。
- trtc-calling-js.jsのボリュームを縮小し、アクセス側で使用されているtrtc-js-sdk、tim-js-sdkおよびtsignalingのバージョン競合を回避するために、trtc-calling-js.jsは、外部依存関係としてtrtc-js-sdk、tim-js-sdk、tsignalingをパッケージ化しているので、使用する前に依存関係を手動でインストールする必要があります。

```
// npm方式によるインストール
npm install trtc-js-sdk --save

npm install tim-js-sdk --save

npm install tsignaling --save

npm install trtc-calling-js --save
```

©2013-2019 Tencent Cloud. All rights reserved.

```
//スクリプト方式によってtrtc-calling-jsを使用する必要がある場合は、順序どおりに次のリソースをインポートする必要があります。

<script src="./trtc.js"></script>
<script src="./tim-js.js"></script>
<script src="./tsignaling.js"></script>
<script src="./trtc-calling-js.js"></script>
```

手順2：TRTCCallingオブジェクトの作成

TRTCCallingオブジェクトを作成し、SDKAppIDパラメータをお客様のSDKAppIDに設定します。

```
import TRTCCalling from 'trtc-calling-js';

let options = {
SDKAppID: 0, // アクセスするときは、0をお客様のSDKAppIDに置き換える必要があります。
// v0.10.2から、timパラメータを追加します
// TIMインスタンスの一意性を保証するために、timパラメータはサービス内にすでに存在するTIMインスタンスに使用されます。
tim: tim
};
const trtcCalling = new TRTCCalling(options);
```

手順3：ログインの完了

```
trtcCalling.login({
userID,
userSig,
});
```

手順4：1v1通話の実現

- **発呼者：特定ユーザーを呼び出す**

```
trtcCalling.call({
userID, //ユーザーID
type: 1, //通話タイプ、0-不明、1-音声通話、2-ビデオ通話
timeout //招待タイムアウト時間、単位s（秒）
});
```

- **着呼者：新しい呼び出しに応答**

©2013-2019 Tencent Cloud. All rights reserved.

```
// 応答
trtcCalling.accept();
//拒否します
trtcCalling.reject()
```

- **通話終了**

```
trtcCalling.hangup()
```

よくあるご質問

通話がつながらなかったり、強制オフラインになったりするのはなぜですか。

コンポーネントは現在、マルチインスタンスのログインや**オフラインプッシュのシグナリング**機能をサポートしていません。現在のログインアカウントの一意性をご確認ください。

説明：

- **マルチインスタンス**：1つのUserIDで繰り返しログインしたり、異なるターミナルからログインしたりすると、シグナリングの混乱が生じます。
- **オフラインプッシュ**：インスタンスはオンラインの場合にのみメッセージを受信できます。インスタンスがオフラインのときに受信したシグナリングは、オンラインになった後は再度プッシュされません。その他よくあるご質問については、TRTCCalling Webに関するご質問をご参照ください。

技術的なお問い合わせ

詳細については、お問い合わせにご連絡をいただくか、colleenyu@tencent.comにメールでご連絡ください。

参考ドキュメント

- TRTCCalling web公式サイトで体験
- TRTCCalling npm
- TRTCCalling web demoソースコード
- TRTCCalling web API
- TRTCCalling webに関するご質問

©2013-2019 Tencent Cloud. All rights reserved.

音声通話(Flutter)

最終更新日：：2022-02-21 17:22:11

音声通話機能をすばやく実装する必要がある場合、当社が提供するDemoをもとにアダプターを修正するか、または当社が提供するTRTCCalling コンポーネントでUIのカスタマイズを実装することができます。

DemoのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援>Demoクイックスタート**を選択します。
2. `TestAudioCall` などアプリケーション名を入力して、作成**をクリックします。

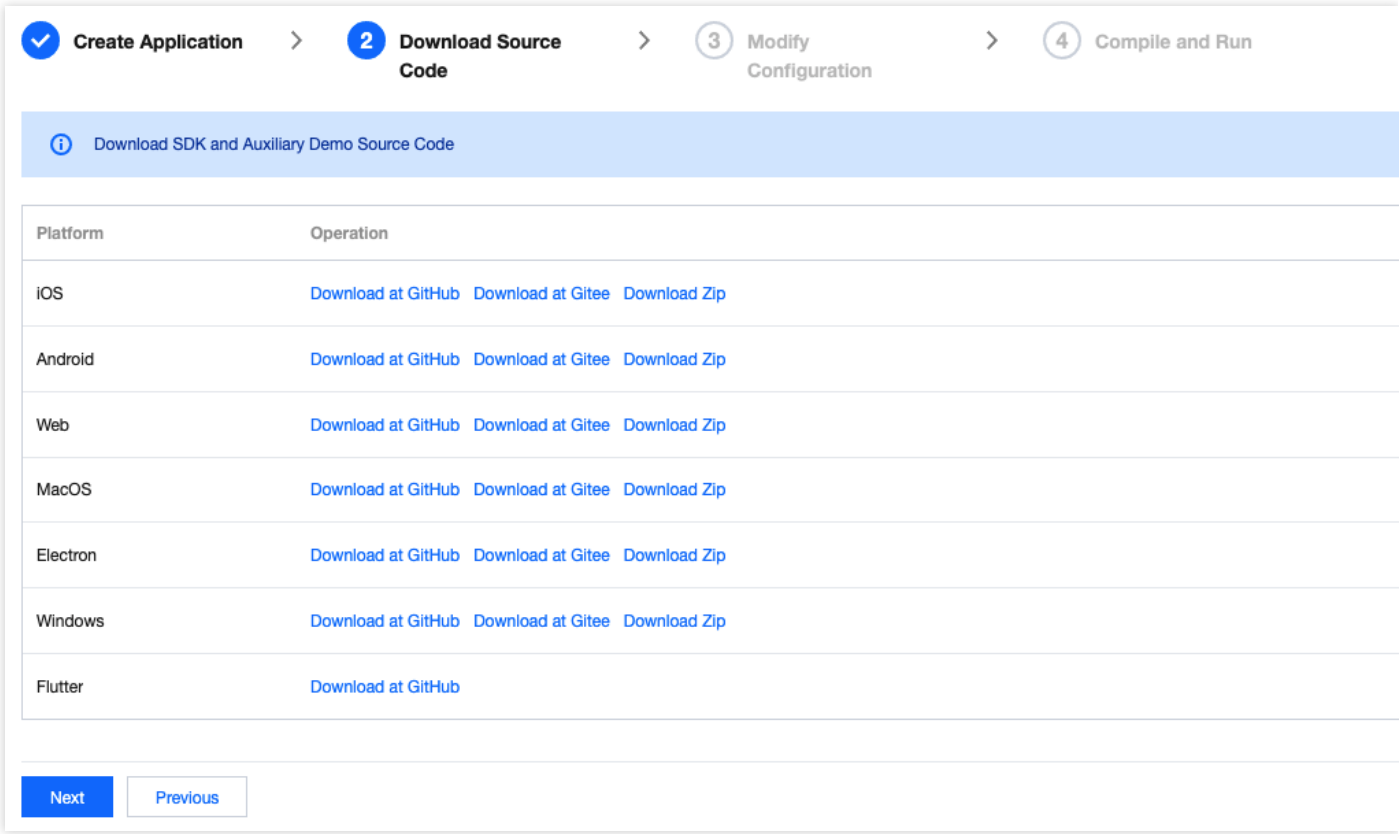
> 注意：
> 本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

ステップ2：SDKおよびDemoソースコードをダウンロード

1. 実際のビジネスニーズに応じて、SDKと付属のDemoソースコードをダウンロードします。

©2013-2019 Tencent Cloud. All rights reserved.

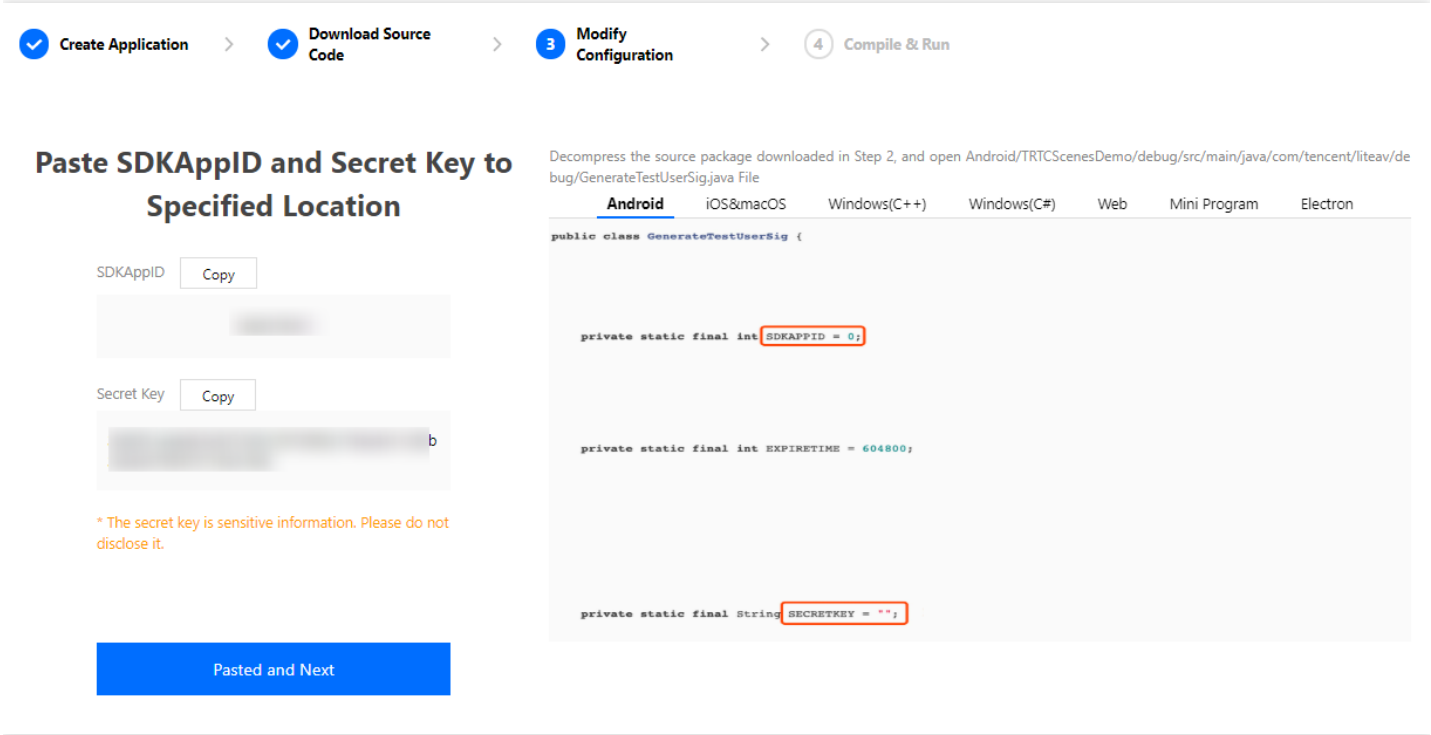
2. ダウンロード完了後、**ダウンロードしました。次のステップ**をクリックします。



ステップ3：Demo プログラムファイルの設定

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。
2. `/example/lib/debug/GenerateTestUserSig.dart` ファイルを見つけて開きます。
3. `GenerateTestUserSig.dart` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはPLACEHOLDER、実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトはPLACEHOLDER、実際のキー情報を設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。
5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及したUserSigの新規作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Demoの動作

注意：
Android端末は実際のマシンで実行する必要があり、エミュレーターのデバッグはサポートされていません。

©2013-2019 Tencent Cloud. All rights reserved.

1. `flutter pub get` を実行します。
2. コンパイルを実行し、デバッグを行います。
   - Android 端末
   - iOS 端末
   i. `flutter run` を実行します。
   ii. Android Studio（3.5およびそれ以降のバージョン）を使用して、ソースプロジェクトを開き、**実行**をクリックすれば完了です。

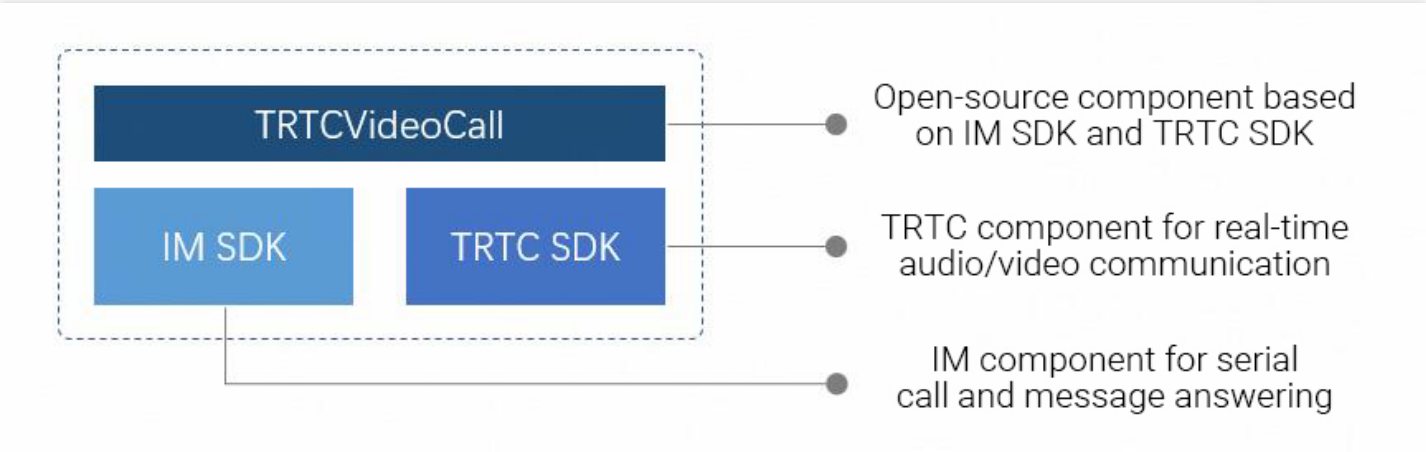
手順5：Demoソースコードの修正

ソースコードフォルダ `TRTCCallingDemo` には2つのサブフォルダuiとmodelが含まれ、そのうちuiフォルダにはインターフェースコードが含まれています。

ファイルまたはフォルダ	機能の説明
TRTCCallingVideo.dart	オーディオビデオ通話のメインインターフェースが表示され、このインターフェースで通話の応答と拒否が完了します。
TRTCCallingContact.dart	連絡先を選択するためのインターフェースを表示するために使用され、このインターフェースを介して登録済みユーザーを検索し、通話を開始できます

UIカスタマイズの実装

ソースコードフォルダ `TRTCCallingDemo` には2つのサブフォルダuiとmodelが含まれ、そのうちmodelフォルダには当社が実装した再利用可能なオープンソースコンポーネントTRTCCallingが含まれています。このコンポーネントが提供するインターフェース関数は `TRTCCalling.dart` ファイルで確認できます。



オープンソースコンポーネントTRTCCallingを使用すれば、自分のUIを実装することができます。すなわちmodel パーツを再利用するだけで、自分でUIパーツを実装できます。

©2013-2019 Tencent Cloud. All rights reserved.

手順1：SDKの統合

オーディオビデオ通話コンポーネントTRTCCallingは、TRTC SDKとIM SDKに依存しています。 `pubspec.yaml` を設定することで、更新を自動的にダウンロードできます。

プロジェクトの `pubspec.yaml` に次の依存関係を記述します。

```
dependencies:
tencent_trtc_cloud: 最新バージョン番号
tencent_im_sdk_plugin: 最新バージョン番号
```

手順2：権限の設定および難読化ルール

- iOS 端末
- Android 端末

`Info.plist` にカメラとマイクの権限申請を追加する必要があります。

```
<key>NSMicrophoneUsageDescription</key>
<string>通常の音声通話が行えるようにマイクの権限を承認します</string>
```

手順3：TRTCCalling コンポーネントをインポート

modelディレクトリ内の全ファイルをプロジェクトにコピーします。

```
/lib/TRTCCallingDemo/model
```

手順4：コンポーネントの初期化およびログイン

1. `TRTCCalling.sharedInstance()` を呼び出して、コンポーネントインスタンスを取得します。
2. `login(SDKAppID, userId, userSig, callback)` を呼び出し、コンポーネントのログインを完了します。このうちいくつかの重要パラメータの入力については、下表をご参照ください。

パラメータ名	作用
SDKAppID	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。計算方法については、UserSigの計算、使用方法をご参照ください。

©2013-2019 Tencent Cloud. All rights reserved.

```
// 初期化
sCall = await TRTCCalling.sharedInstance();
sCall.login(1400000123, "userA", "xxxx");
```

手順5：1v1音声通話を実現

1. 発信者： `TRTCCalling` の `call()` メソッドを呼び出して通話リクエストを開始し、ユーザーID（userid）と通話タイプ（type）を渡して、通話タイプパラメータを `TRTCCalling.typeAudioCall` に渡します。
2. 受信者：受信者がログイン状態の場合は、 `onInvited()` という名称のイベント通知を受け取ります。コールバックにおける `callType` のパラメータは発信者が入力した通話タイプであり、このパラメータを介して対応するインターフェースを起動することができます。
3. 受信者：電話に対応したい場合、受信者は `accept()` 関数をコールすると同時に `openCamera()` 関数を呼び出して、自身のローカルCCDカメラを起動します。受信者は `reject()` を呼び出し、この通話を拒否することもできます。
4. 双方のオーディオビデオチャネルが確立した後、通話する双方は `onUserVideoAvailable()` という名称のイベント通知を受け取ります。これは、相手方のビデオ画面を取得済みであることを示しています。このとき、双方のユーザーは `startRemoteView()` を呼び出し、リモートのビデオ画面を表示することができます。リモートの音声はデフォルトで自動再生に設定されています。

コンポーネントAPIリスト

TRTCCallingコンポーネントのAPIインターフェースリストは次のとおりです。

インターフェース関数	インターフェースの機能
registerListener	TRTCCalling監視装置を追加します。ユーザーはこの監視装置を介してステータス通知を取得することができます
unRegisterListener	監視装置を削除します
destroy	インスタンスを破棄します
login	IMにログインします。すべての機能を使用するためには、まずログインする必要があります
logout	IMからログアウトします。ログアウト後はダイヤル操作ができません
call	C2Cの通話に招待します。被招待者はonInvitedのイベント通知を受け取ります
accept	被招待者として通話に応答します
reject	被招待者として通話を拒否します

©2013-2019 Tencent Cloud. All rights reserved.

インターフェース関数	インターフェースの機能
hangup	通話を終了します
setMicMute	micをミュート/非ミュートに設定します
setHandsFree	ハンズフリーを起動するかどうかを設定します

©2013-2019 Tencent Cloud. All rights reserved.

TUICalling API(iOS)

最終更新日：：2021-12-07 10:29:44

TUICallingは、Tencent CloudのTRTCとIMを組み合わせたサービスであり、1v1および多人数のビデオ通話をサポートしています。TUICallingはTencent Cloudの2つのクローズドソースSDKに依存するオープンソースのClassです。具体的な実装プロセスについては、リアルタイム音声通話（iOS）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

TUICalling API概要

SDK 基本関数

API	説明
sharedInstance	コンポーネントシングルトン。
call	C2C通話に招待します。
receiveAPNSCalled	被招待者として着信に応答します。
setCallingListener	リスナーを設定します。
setCallingBell	着信音の設定（30s以内を推奨）
enableMuteMode	ミュートモードをオンにします
enableCustomViewRoute	カスタムビューの有効化

TUICallingListener API概要

イベントコールバック

API	説明
shouldShowOnCallView	呼び出された時の応答ページのプルリクエスト
onCallStart	呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます
onCallEnd	通話コールバック。発呼側、着呼側ともにトリガーされます

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onCallEvent	通話イベントコールバック

Type API概要

通話タイプ

enum	説明
TUICallingTypeAudio	オーディオ通話
TUICallingTypeVideo	ビデオ通話

Role API概要

ユーザーロールタイプ

enum	説明
TUICallingRoleCall	通話発信者（発呼側）
TTUICallingRoleCalled	通話受信者（着呼側）

Event API概要

イベントタイプ

enum	説明
TUICallingEventCallStart	通話開始
TUICallingEventCallSucceed	通話接続成功
TUICallingEventCallEnd	通話終了
TUICallingEventCallFailed	通話失敗

SDK基本関数

©2013-2019 Tencent Cloud. All rights reserved.

sharedInstance

shareInstanceはTUICallingManagerのコンポーネントシングルトンです。

```
+ (instancetype)shareInstance;
```

call

C2C通話に招待します。

```
- (void)call:(NSArray<NSString *> *)userIDs type:(TUICallingType)type;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ

receiveAPNSCalled

被招待者として着信に応答します。

```
- (void)receiveAPNSCalled:(NSArray<NSString *> *)userIDs type:(TUICallingType)type;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ

setCallingListener

リスナーを設定します。

```
- (void)setCallingListener:(id<TUICallingListerner>)listener;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
listener	TUICallingListener	TUIcallingコンポーネントリスナー

©2013-2019 Tencent Cloud. All rights reserved.

setCallingBell

着信音を設定します（30s以内を推奨）。

```
- (void)setCallingBell:(NSString *)filePath;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
filePath	NSString	着信音リソースパス

enableMuteMode

ミュートモードをオンにします。

```
- (void)enableMuteMode:(BOOL)enable;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	BOOL	ミュートモード有効化の有無

enableCustomViewRoute

カスタムビューをオンにします。
オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewControllerのインスタンスを受信します。開発者自身が表示方式を決定します。

> 注意：
> フルスクリーン表示にする必要があります。そうしない場合、表示に異常が生じます。

```
- (void)enableCustomViewRoute:(BOOL)enable;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	BOOL	カスタムビュー有効化の有無

©2013-2019 Tencent Cloud. All rights reserved.

TUICallingListenerコールバック関数

shouldShowOnCallView

呼び出された時の応答ページのプルリクエストに対する同意の有無。

```
- (BOOL)shouldShowOnCallView;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
戻り値	BOOL	同意の有無

onCallStart

呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます。

```
- (void)callStart:(NSArray<NSString *> *)userIDs type:(TUICallingType)type role:(TUICallingRole)r
ole viewController:(UIViewController * _Nullable)viewController;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ
role	TUICallingRole	ユーザーロールタイプ：発呼側/着呼側
viewController	UIViewController	通話ビューViewController

onCallEnd

通話終了コールバック。発呼側、着呼側ともにトリガーされます。enableCustomViewRouteの設定がNOの時、このコールバックメソッドはトリガーされません。

```
- (void)callEnd:(NSArray<NSString *> *)userIDs type:(TUICallingType)type role:(TUICallingRole)rol
e totalTime:(CGFloat)totalTime;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userIDs	NSArray	通話ユーザーIDリスト
type	TUICallingType	通話タイプ：オーディオ/ビデオ
role	TUICallingRole	ユーザーロールタイプ：発呼側/着呼側
totalTime	CGFloat	通話時間。単位：秒

onCallEvent

通話イベントコールバック。enableCustomViewRouteの設定がNOの時、このコールバックメソッドはトリガーされません。

```
- (void)onCallEvent:(TUICallingEvent)event type:(TUICallingType)type role:(TUICallingRole)role message:(NSString *)message;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
event	TUICallingEvent	通話イベントタイプ。
type	TUICallingType	通話タイプ：オーディオ/ビデオ
role	TUICallingRole	ユーザーロールタイプ：発呼側/着呼側
message	NSString	イベントの説明情報

©2013-2019 Tencent Cloud. All rights reserved.

TUICalling API(Android)

最終更新日：：2021-12-07 10:29:44

TUICallingは、Tencent CloudのTRTCとIMを組み合わせたサービスであり、1v1および多人数のビデオ通話をサポートしています。TUICallingはTencent Cloudの2つのクローズドソースSDKに依存するオープンソースのClassです。具体的な実装プロセスについては、リアルタイム音声通話（Android）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

TUICalling API概要

SDK 基本関数

API	説明
sharedInstance	コンポーネントシングルトン
call	C2C通話に招待します
receiveAPNSCalled	被招待者として着信に応答します
setCallingListener	リスナーを設定します。
setCallingBell	着信音の設定（30s以内を推奨）
enableMuteMode	ミュートモードをオンにします
enableCustomViewRoute	カスタムビューの有効化

TUICallingListener API概要

イベントコールバック

API	説明
shouldShowOnCallView	呼び出された時の応答ページのプルリクエスト
onCallStart	呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます
onCallEnd	通話コールバック。発呼側、着呼側ともにトリガーされます

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onCallEvent	通話イベントコールバック

Type API概要

通話タイプ

enum	説明
AUDIO	オーディオ通話
VIDEO	ビデオ通話

Role API概要

ユーザーロールタイプ

enum	説明
CALL	通話発信者（発呼側）
CALLED	通話受信者（着呼側）

Event API概要

イベントタイプ

enum	説明
CALL_START	通話開始
CALL_SUCCEED	通話接続成功
CALL_END	通話終了
CALL_FAILED	通話失敗

SDK基本関数

©2013-2019 Tencent Cloud. All rights reserved.

sharedInstance

shareInstanceはTUICallingのコンポーネントシングルトンです。

```
public static TUICallingManager sharedInstance();
```

call

C2C通話に招待します。

```
void call(String[] userIDs, Type type);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ

receiveAPNSCalled

被招待者として着信に応答します。

```
void receiveAPNSCalled(String[] userIDs, Type type);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ

setCallingListener

リスナーを設定します。

```
void setCallingListener(TUICallingListener listener);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
listener	TUICallingListener	TUIcallingコンポーネントリスナー

©2013-2019 Tencent Cloud. All rights reserved.

setCallingBell

着信音を設定します(30s以内を推奨)。

```
void setCallingBell(String filePath);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
filePath	String	着信音リソースパス

enableMuteMode

ミュートモードをオンにします。

```
void enableMuteMode(boolean enable);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	boolean	ミュートモード有効化の有無

enableCustomViewRoute

カスタムビューをオンにします。
オンにすると、呼び出し/被呼び出し開始コールバックの中で、CallingViewのインスタンスを受信します。開発者自身が表示方式を決定します。

> 注意：
> フルスクリーンまたはスクリーンと同じ比率で表示する必要があります。そうしない場合、表示に異常が生じます。

```
void enableCustomViewRoute(boolean enable);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	boolean	カスタムビュー有効化の有無

©2013-2019 Tencent Cloud. All rights reserved.

TUICallingListenerコールバック関数

shouldShowOnCallView

呼び出された時の応答ページのプルリクエストに対する同意の有無。

```
boolean shouldShowOnCallView();
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
戻り値	boolean	同意の有無

onCallStart

呼び出し開始コールバック。発呼側、着呼側ともにトリガーされます。

```
void onCallStart(String[] userIDs, TUICalling.Type type, TUICalling.Role role, View tuiCallingVie
w);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト。
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ
role	TUICalling.Role	ユーザーロールタイプ：発呼側/着呼側
tuiCallingView	View	通話ビューView。enableCustomViewRouteの設定がfalseの時、viewはnullになります。

onCallEnd

通話終了コールバック。発呼側、着呼側ともにトリガーされます。

```
void onCallEnd(String[] userIDs, TUICalling.Type type, TUICalling.Role role, long totalTime);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDs	String[]	通話ユーザーIDリスト

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ
role	TUICalling.Role	ユーザーロールタイプ：発呼側/着呼側
totalTime	long	通話時間。単位：秒

onCallEvent

通話イベントコールバック。

```
void onCallEvent(TUICalling.Event event, TUICalling.Type type, TUICalling.Role role, String message);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
event	TUICalling.Event	通話イベントタイプ
type	TUICalling.Type	通話タイプ：オーディオ/ビデオ
role	TUICalling.Role	ユーザーロールタイプ：発呼側/着呼側
message	String	イベントの説明情報

©2013-2019 Tencent Cloud. All rights reserved.

TRTCCalling API(Web)

最終更新日：：2022-04-02 16:27:03

TRTCCallingの概要

TRTCCallingコンポーネントは、Tencent CloudのTencent Real-Time Communication（TRTC）とInstant Messaging（IM）サービスの組み合わせにより構成されており、1v1および複数人でのビデオ/音声通話をサポートしています。具体的な実装プロセスについては、リアルタイム音声通話（Web）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

Demoのダウンロード

クリックしてTUICallingに進み、実際のサービスのニーズに応じてDemoソースコードをダウンロードします。

環境要件

最新バージョンのChromeブラウザをご使用ください。現在、デスクトップのChromeブラウザはTRTC Web SDKをサポートしており、関連機能は比較的整っていますので、Chromeブラウザを使用して体験することをお勧めします。具体的な内容については 環境要件をご参照ください。

URLドメイン名プロトコルの制限

ユースケース	プロトコル	受信（再生）	送信（マイク・オン）	画面共有	備考
本番環境	HTTPSプロトコル	サポートあり	サポートあり	サポートあり	推奨
本番環境	HTTPプロトコル	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	http://localhost	サポートあり	サポートあり	サポートあり	推奨

©2013-2019 Tencent Cloud. All rights reserved.

ユースケース	プロトコル	受信（再生）	送信（マイク・オン）	画面共有	備考
ローカル開発環境	http://127.0.0.1	サポートあり	サポートあり	サポートあり	-
ローカル開発環境	http://[ローカルマシンIP]	サポートあり	サポートなし	サポートなし	-
ローカル開発環境	file:///	サポートあり	サポートあり	サポートあり	-

TRTCCalling API

イベントサブスクリプション/サブスクリプションのキャンセルに関するインターフェース関数

このコンポーネントはイベントのディスパッチに基づいて管理します。アプリケーション層は、コンポーネントからディスパッチされるイベントにインタラクティブに応じて変更することができます。

API	説明
on(eventName, callback, context)	イベントのサブスクリプション
off(eventName, callback, context)	イベントサブスクリプションの取消

SDK 基本関数

API	説明
login({userID, userSig})	IMログインインターフェース。すべての機能を使用するためには、まずログインする必要があります
logout()	インターフェースからログアウトします。ログアウトした後は、ダイヤル操作ができなくなります

通話操作に関連するインターフェース関数

API	説明
call({userID, type, offlinePushInfo}))	シングル通話の招待
groupCall({userIDList, type, groupID, offlinePushInfo})	グループチャット通話の招待
accept()	通話招待に応答

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
reject()	通話招待を拒否
hangup()	現在の通話の終了

ビデオ制御に関連するインターフェース関数

API	説明
startRemoteView({userID, videoViewDomID})	リモート画面レンダリングの起動
stopRemoteView({userID})	リモート画面レンダリングの停止
startLocalView({userID, videoViewDomID})	ローカル画面レンダリングの起動
stopLocalView({userID})	ローカル画面レンダリングの停止
openCamera()	カメラの起動
closeCamera()	カメラの停止
setMicMute(isMute)	デバイスマイクのミュートの有無
setVideoQuality(profile)	ビデオ画質の設定
switchToAudioCall()	ビデオ通話を音声通話へ切り替え
switchToVideoCall()	音声通話をビデオ通話へ切り替え
getCameras()	カメラデバイスリストの取得
getMicrophones()	マイクデバイスリストの取得
switchDevice({deviceType, deviceId})	カメラまたはマイクデバイスの切り替え

TRTCCallingの詳細

TRTCCallingコンポーネントのインスタンスの作成

まず、Tencent Real-Time Communicationコンソールでアプリケーションを作成し、SDKAppIDを取得する必要があります。
その後、 `new TRTCCalling()` メソッドを使ってTRTCCalling コンポーネントのインスタンスを取得することができます。

©2013-2019 Tencent Cloud. All rights reserved.

```
npm i trtc-js-sdk --save
npm i tim-js-sdk --save
npm i tsignaling --save
npm i trtc-calling-js --save
// nodeを通じてダウンロードする依存関係の場合は、importによって次をインポートします
import TRTCCalling from 'trtc-calling-js';
// スクリプト方式によってtrtc-calling-jsを使用する場合は、順序どおりに
// trtc.jsを手動でインポートします
<script src="./trtc.js"></script>
// 続いて、tim-js.jsを手動でインポートします。
<script src="./tim-js.js"></script>
// 次に、tsignaling.jsを手動でインポートします。
<script src="./tsignaling.js"></script>
// 最後に、trtc-calling-js.jsを手動でインポートします。
<script src="./trtc-calling-js.js"></script>
let options = {
SDKAppID: 0, // アクセスするときは、0をお客様のIMアプリケーションのSDKAppIDに置き換える必要があります
// v0.10.2から、timパラメータを追加します
// timパラメータはサービス内にすでに存在するTIMインスタンスに使用され、TIMインスタンスの一意性を保証します。
tim: tim
};
let trtcCalling = new TRTCCalling(options);
```

イベントのサブスクリプション/サブスクリプションキャンセルに関するインターフェース関数

on(eventName, callback, context)

コンポーネントからディスパッチされたイベントを監視するために使用します。詳細については、イベントリストをご参照ください。

```
let handleInvite = function ({inviteID, sponsor, inviteData}) {
console.log(`inviteID: ${inviteID}, sponsor: ${sponsor}, inviteData: ${JSON.stringify(inviteData)}`);
};
trtcCalling.on('onInvited', handleInvite, this);
```

off(eventName, callback, context)

イベント監視をキャンセルするために使用します。

```
let handleInvite = function ({inviteID, sponsor, inviteData}) {
console.log(`inviteID: ${inviteID}, sponsor: ${sponsor}, inviteData: ${JSON.stringify(inviteData)}`);
```

©2013-2019 Tencent Cloud. All rights reserved.

```
};
trtcCalling.off('onInvited', handleInvite, this);
```

SDK基本関数

login({userID, userSig})

インターフェースにログインします。

```
trtcCalling.login({userID, userSig})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	現在のユーザーのID。文字列タイプ。アルファベット（a-zおよびA-Z）、数字（0-9）、ハイフン（-）、アンダーバー（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算、使用方法をご参照ください。

logout()

インターフェースからログアウトします。

```
trtcCalling.logout()
```

通話操作に関連するインターフェース関数

call({userID, type, offlinePushInfo})

1v1通話招待、そのうちtypeは通話タイプ、1は音声通話、2はビデオ通話です。

説明：

- v1.0.0以降のバージョンで、timeoutパラメータを取り消しました
- v1.0.0以降のバージョンに、offlinePushInfoパラメータを新規追加しました（**オフラインプッシュはAndroidまたはiOS端末にのみ適用されます。WebおよびWeChat Mini Programはサポートしていません**）。

```
// v1.0.0以前
trtcCalling.call({userID, type, timeout});
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// v1.0.0以降のバージョン
const offlinePushInfo = {
title: '',
description: '通話リクエストが1件あります',
}
trtcCalling.call({userID, type, offlinePushInfo})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	被招待者のuserID
type	Number	1：音声通話、2：ビデオ通話
timeout	Number	0はタイムアウトしていないことを意味し、単位はs（秒）。**v1.0.0より前のバージョンのみ**
offlinePushInfo	Object	メッセージのオフラインプッシュをカスタマイズします（オプション）。**v1.0.0以降のバージョンのみ**

offlinePushInfoパラメータ（v1.0.0以降のバージョンのみ）

パラメータ	タイプ	意味
title	String	オフラインプッシュタイトル（オプション）
description	String	オフラインプッシュコンテンツ（オプション）
androidOPPOChannelID	String	オフラインプッシュのためのOPPO携帯（システム8.0以上）のチャンネルIDを設定します（オプション）。
extension	String	オフラインプッシュパススルーコンテンツ（オプション）。**TRTCCallingバージョン>=1.0.2、tsignalingバージョン>=0.9.0のみ**

groupCall({userIDList, type, groupID, offlinePushInfo})

groupID パラメータは IM SDKにおけるグループ IDです。このパラメータを設定すると、通話リクエストメッセージがグループメッセージシステムを介してブロードキャストされます。このメッセージブロードキャスト方式は簡単で信頼性の高い方法です。設定しない場合は、 `TRTCCalling` コンポーネントが単発メッセージで1人1人に通知を行います。

説明：

©2013-2019 Tencent Cloud. All rights reserved.

v1.0.0以降のバージョンに、offlinePushInfoパラメータを新規追加しました（**オフラインプッシュはAndroidまたはiOS端末にのみ適用されます。WebおよびWeChat Mini Programはサポートしていません**）。

```
// v1.0.0より前
trtcCalling.groupCall({userIDList, type, groupID});

// v1.0.0以降
const offlinePushInfo = {
title: '',
description: '通話リクエストが1件あります',
}
trtcCalling.groupCall({userIDList, type, groupID, offlinePushInfo})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIDList	Array	招待リスト
type	Number	1：音声通話、2：ビデオ通話
groupID	String	IMグループID（オプション）
offlinePushInfo	Object	メッセージのオフラインプッシュをカスタマイズします（オプション）。**v1.0.0以降のバージョンのみ**

offlinePushInfoパラメータ（v1.0.0以降のバージョンのみ）

パラメータ	タイプ	意味
title	String	オフラインプッシュタイトル（オプション）
description	String	オフラインプッシュコンテンツ（オプション）
androidOPPOChannelID	String	オフラインプッシュのためのOPPO携帯（システム8.0以上）のチャンネルIDを設定します（オプション）。
extension	String	オフラインプッシュパススルーコンテンツ（オプション）。**TRTCCallingバージョン>=1.0.2、tsignalingバージョン>=0.9.0のみ**

accept()

©2013-2019 Tencent Cloud. All rights reserved.

招待を受け取った後、このインターフェースを呼び出すと、現在の招待を受け入れます。

説明：

- 前回のinvitationの処理が未完了の場合、コンポーネントはデフォルトでビジー状態になり、その後のすべての招待はビジーを返します。
- v1.0.0以降のバージョンで、paramsパラメータを取り消しました。

```
import TRTCCalling from 'trtc-calling-js';
trtcCalling.on(TRTCCalling.EVENT.INVITED, ({inviteID, sponsor, inviteData}) => {
// ...
// v1.0.0以前
const { roomID, callType } = inviteData;
trtcCalling.accept({inviteID, roomID, callType})
// v1.0.0以降
trtcCalling.accept();
})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
inviteID	String	招待ID。1回の招待を標識します（イベントINVITEDコールバックデータのinviteIDを監視）。**v1.0.0より前のバージョンのみ**
roomID	Number	通話ルームナンバーID（イベントINVITEDコールバックデータのinviteData.roomIDを監視）。**v1.0.0より前のバージョンのみ**
callType	Number	1：音声通話、2：ビデオ通話（イベントINVITEDコールバックデータのinviteData.callTypeを監視）。**v1.0.0より前のバージョンのみ**

reject()

招待を受け取った後、このインターフェースを呼び出すと、現在の招待が拒否されます。

説明：
v1.0.0以降のバージョンで、paramsパラメータを取り消しました。

```
import TRTCCalling from 'trtc-calling-js';
trtcCalling.on(TRTCCalling.EVENT.INVITED, ({inviteID, sponsor, inviteData}) => {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// ...
// v1.0.0より前
const { callType } = inviteData;
trtcCalling.reject({inviteID, isBusy, callType})
// v1.0.0以降
trtcCalling.reject();
})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
inviteID	String	招待ID。1回の招待を識別します（イベントINVITEDコールバックデータのinviteIDを監視）。**v1.0.0より前のバージョンのみ**
isBusy	Boolean	回線ビジー状態の有無。**v1.0.0より前のバージョンのみ**
callType	Number	1：音声通話、2：ビデオ通話（イベントINVITEDコールバックデータのinviteData.callTypeを監視）。**v1.0.0より前のバージョンのみ**

hangup()

1. 通話中にこの関数を呼び出せば、通話を終了することができます。
2. 応答がない状態では、通話をキャンセルするために用いることができます。

```
trtcCalling.hangup()
```

ビデオ制御に関連するインターフェース関数

startRemoteView({userID, videoViewDomID})

リモートユーザーのカメラデータを指定されたDOM IDノードにレンダリングします。

```
trtcCalling.startRemoteView({userID, videoViewDomID})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	このユーザーデータは、このDOM IDノードにレンダリングされたvideoタグを介して再生されます

stopRemoteView({userID})

©2013-2019 Tencent Cloud. All rights reserved.

リモートユーザーのカメラデータによってレンダリングされたDOMノードを削除します。

説明：
v1.0.0以降のバージョンで、videoViewDomIDパラメータを削除しました。

```
// v1.0.0以前
trtcCalling.stopRemoteView({userID, videoViewDomID});
// v1.0.0以降
trtcCalling.stopRemoteView({userID});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	このDOM IDノードのvideoタグを削除し、ビデオ再生を停止します。**v1.0.0より前のバージョンのみ**

startLocalView({userID, videoViewDomID})

ローカルユーザーのカメラデータを指定されたDOM IDノードにレンダリングします。

```
trtcCalling.startLocalView({userID, videoViewDomID})
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	ローカルユーザーデータは、このDOM IDノードにレンダリングされたvideoタグを介して再生されます

stopLocalView({userID})

ローカルユーザーのカメラデータによってレンダリングされたDOMノードを削除します。

説明：
v1.0.0以降のバージョンで、videoViewDomIDパラメータを削除しました。

©2013-2019 Tencent Cloud. All rights reserved.

```
// v1.0.0以前
trtcCalling.stopLocalView({userID, videoViewDomID});
// v1.0.0以降
trtcCalling.stopLocalView({userID});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
videoViewDomID	String	このDOM IDノードのvideoタグを削除し、ビデオ再生を停止します。 **v1.0.0より前のバージョンのみ**

openCamera()

ローカルカメラをオンにします。

```
trtcCalling.openCamera()
```

closeCamera()

カメラをオフにします。

```
trtcCalling.closeCamera()
```

setMicMute(isMute)

マイクをオン/オフします。

```
trtcCalling.setMicMute(true) // マイクオフ
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMute	Boolean	• true: マイクのオフ • false: マイクのオン

setVideoQuality(profile)

ビデオの画質を設定します。

©2013-2019 Tencent Cloud. All rights reserved.

説明：

- v0.8.0以降のバージョンは、このメソッドが新規追加されました。
- このメソッドは、call、groupCall、acceptの前に設定する必要があり、その後の設定は有効になりません。

```
trtcCalling.setVideoQuality('720p') // ビデオの画質を720pに設定します
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
profile	String	• 480p：640 × 480 • 720p：1280 × 720 • 1080p：1920 × 1080

switchToAudioCall()

ビデオ通話を音声通話へ切り替えます。

説明：

- v0.10.0以降のバージョンは、このメソッドが新規追加されました。
- 1v1通話中のみ使用をサポートします。
- ERRORイベントの監視に失敗しました。code：60001。

```
trtcCalling.switchToVideoCall() // ビデオ通話を音声通話へ切り替え
```

switchToVideoCall()

音声通話をビデオ通話へ切り替えます。

説明：

- v0.10.0以降のバージョンは、このメソッドが新規追加されました。
- 1v1通話中のみ使用をサポートします。
- ERRORイベントの監視に失敗しました。code：60002。

©2013-2019 Tencent Cloud. All rights reserved.

```
trtcCalling.switchToVideoCall() // 音声通話をビデオ通話へ切り替え
```

getCameras()

このインターフェースを呼び出すと、カメラデバイスリストを取得することができます。

> 説明：
> v1.0.0以降のバージョンに、このメソッドが新規追加されました。

```
trtcCalling.getCameras() // カメラリストの取得
```

getMicrophones()

このインターフェースを呼び出すと、マイクデバイスリストを取得することができます。

> 説明：
> v1.0.0以降のバージョンに、このメソッドが新規追加されました。

```
trtcCalling.getMicrophones() // マイクリストの取得
```

switchDevice({deviceType, deviceId})

このインターフェースを呼び出すと、カメラまたはマイクのデバイスを切り替えることができます。

> 説明：
> v1.0.0以降のバージョンに、このメソッドが新規追加されました。

```
trtcCalling.switchDevice({deviceType: 'audio', deviceId: deviceId}) // デバイスの切り替え
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
deviceType	String	video: カメラ、audio: マイク

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
deviceId	String	• カメラデバイスIDはgetCameras()で取得します • マイクデバイスIDはgetMicrophones()で取得します

TRTCCallingイベントリスト

次のコードを参照して、TRTCCalling コンポーネントイベントを監視できます。

```
import TRTCCalling from 'trtc-calling-js';
// etc
function handleInviteeReject({userID}) {

}
trtcCalling.on(TRTCCalling.EVENT.REJECT, handleInviteeReject)
```

招待側イベント

CODE	イベント受信側	説明
REJECT	招待側	被招待ユーザーは通話拒否
NO_RESP	招待側	被招待ユーザーの応答がなくタイムアウト
LINE_BUSY	招待側	被招待ユーザーは通話中、ビジー状態
INVITED	被招待側	招待通知を受信済み
CALLING_CANCEL	被招待側	相手側による今回通話のキャンセル
CALLING_TIMEOUT	被招待側	今回の通話に応答せずにタイムアウト
USER_ENTER	招待側と被招待側	ユーザー入室
USER_LEAVE	招待側と被招待側	通話から離脱したユーザーあり
CALL_END	招待側と被招待側	今回の通話終了
KICKED_OUT	招待側と被招待側	ログイン重複により、ルームから強制退出
USER_VIDEO_AVAILABLE	招待側と被招待側	リモートユーザーによるカメラのオン/オフ
USER_AUDIO_AVAILABLE	招待側と被招待側	リモートユーザーによるマイクのオン/オフ

©2013-2019 Tencent Cloud. All rights reserved.

一般的なイベントコールバック

SDK_READY

SDKがready状態に入るとこのコールバックを受信します

> 説明：
> v1.0.0以降のバージョンに、このイベントが新規追加されました。

```
let onSDKReady = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.SDK_READY, onSDKReady);
```

USER_ENTER

ユーザーが入室しました。

トリッガー：ユーザーが通話に参加する。

```
let handleUserEnter = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.USER_ENTER, handleUserEnter);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID

USER_LEAVE

ユーザーが退室しました。

トリッガー：ユーザーが通話を退出する。

```
let handleUserLeave = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.USER_LEAVE, handleUserLeave);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userID	String	ユーザーID

GROUP_CALL_INVITEE_LIST_UPDATE

グループチャットで招待リストを更新するとこのコールバックを受信します

> 説明：
> v1.0.0以降のバージョンに、このイベントが新規追加されました。

```
let handleGroupInviteeListUpdate = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.GROUP_CALL_INVITEE_LIST_UPDATE, handleGroupInviteeListUpdate);
```

CALL_END

今回の通話は終了しました。
トリッガー：今回の通話を終了する。

```
let handleCallingEnd = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.CALL_END, handleCallingEnd);
```

KICKED_OUT

ログインを繰り返したことにより、強制退室されました。
トリッガー：他のページに繰り返しログインする。

```
let handleKickedOut = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.KICKED_OUT, handleKickedOut);
```

USER_VIDEO_AVAILABLE

リモートユーザーがカメラをオン/オフにします。
トリッガー：リモートユーザーがカメラをオン/オフにする。

©2013-2019 Tencent Cloud. All rights reserved.

```
let handleUserVideoChange = function({userID, isVideoAvailable}) {
console.log(userID, isVideoAvailable)
};
trtcCalling.on(TRTCCalling.EVENT.USER_VIDEO_AVAILABLE, handleUserVideoChange);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
isVideoAvailable	Boolean	• true：リモートユーザーによるカメラオン • false：リモートユーザーによるカメラオフ

USER_AUDIO_AVAILABLE

リモートユーザーがマイクをオン/オフにしました。

トリッガー：リモートユーザーがマイクをオン/オフする。

```
let handleUserAudioChange = function({userID, isAudioAvailable}) {
console.log(userID, isAudioAvailable)
};
trtcCalling.on(TRTCCalling.EVENT.USER_AUDIO_AVAILABLE, handleUserAudioChange);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
isAudioAvailable	Boolean	• true：リモートユーザーによるマイクオン • false：リモートユーザーによるマイクオフ

招待側イベントコールバック

REJECT

ユーザーが通話を拒否しました。

トリッガー：被招待者が通話を拒否し、発信者がREJECTイベントコールバックを受け取る。

```
let handleInviteeReject = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.REJECT, handleInviteeReject);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID

NO_RESP

招待されたユーザーは応答しませんでした。
トリッガー：call/groupCallにtimeoutを設定し、被招待者がtimeout内に応答しなかった場合、発信者がNO_RESPイベントコールバックを受け取る。

```
let handleNoResponse = function({userID, userIDList}) {
console.log(userID, userIDList)
};
trtcCalling.on(TRTCCalling.EVENT.NO_RESP, handleNoResponse);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID
userIDList	Array	タイムアウトユーザーリスト

LINE_BUSY

被招待側は通話中で、ビジー状態です。
トリッガー：被招待者が通話中である場合、発信者がLINE_BUSYイベントコールバックを受け取る。

```
let handleLineBusy = function({userID}) {
console.log(userID)
};
trtcCalling.on(TRTCCalling.EVENT.LINE_BUSY, handleLineBusy);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID

被招待側イベントコールバック

INVITED

©2013-2019 Tencent Cloud. All rights reserved.

招待通知を受領しました。
トリッガー：招待通知があった場合、被招待者がINVITEDイベントコールバックを受け取る。

```
let handleNewInvitationReceived = function({
sponsor, userIDList, isFromGroup, inviteData, inviteID
}) {
console.log(sponsor, userIDList, isFromGroup, inviteData, inviteID)
};
trtcCalling.on(TRTCCalling.EVENT.INVITED, handleNewInvitationReceived);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sponsor	String	招待者
userIDList	Array	同時に招待された人
isFromGroup	Boolean	IMグループの招待の有無
inviteData	Object	新規ユーザーへの招待：{version, callType, roomID}
inviteID	String	招待ID。1回の招待を識別

CALLING_CANCEL

今回の通話はキャンセルされました。
トリッガー：発信者がコール中に通話をキャンセルし、被招待者がCALLING_CANCELイベントコールバックを受け取る。

```
let handleCallingCancel = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.CALLING_CANCEL, handleCallingCancel);
```

CALLING_TIMEOUT

今回の通話はタイムアウトで、応答はありませんでした。
トリッガー：call/groupCallにtimeoutを設定し、被招待者がtimeout内に応答しなかった場合、被招待者がCALLING_TIMEOUTイベントコールバックを受け取る

```
let handleCallingTimeout = function(event) {
console.log(event)
};
trtcCalling.on(TRTCCalling.EVENT.CALLING_TIMEOUT, handleCallingTimeout);
```

©2013-2019 Tencent Cloud. All rights reserved.

TRTCCalling エラーコードリスト

EVENTのERRORフィールドを監視することによって、コンポーネントからスローされたエラーを処理することができます。サンプルコードは次のとおりです。

```
import TRTCCalling from 'trtc-calling-js';
let onError = function(error) {
console.log(error)
};
trtcCalling.on(TRTCCalling.EVENT.ERROR, onError);
```

Error code

code	エラータイプ	意味
60001	メソッドのコールに失敗しました	switchToAudioCallのコールに失敗しました
60002	メソッドのコールに失敗しました	switchToAudioCallのコールに失敗しました
60003	権限の取得に失敗しました	使用可能なマイクデバイスがありません
60004	権限の取得に失敗しました	使用可能なカメラデバイスがありません
60005	権限の取得に失敗しました	ユーザーがデバイスの使用を禁止しています
60006	環境検出に失敗しました	現在の環境はWebRTC（>=v1.0.4バージョン）をサポートしていません

アップグレードガイド

- **TRTCCallingバージョンを>= 1.0.2にアップグレードします **
  - 注意：TSignalingバージョンを>= 0.9.0にアップグレードする必要があります
  - 原因：ログの更新
- **1.0.2にアップグレード >TRTCCallingバージョン>=1.0.0**
  - 注意：TSignalingバージョンを>= 0.8.0にアップグレードする必要があります
  - 原因：ログの更新

©2013-2019 Tencent Cloud. All rights reserved.

よくあるご質問

通話がつながらなかったり、強制オフラインになったりするのはなぜですか。

コンポーネントは現在、マルチインスタンスのログインや**オフラインプッシュのシグナリング**機能をサポートしていません。現在のログインアカウントの一意性をご確認ください。

説明：

- **マルチインスタンス**：1つのUserIDで繰り返しログインしたり、異なるターミナルからログインしたりすると、シグナリングの混乱が生じます。
- **オフラインプッシュ**：インスタンスはオンラインの場合にのみメッセージを受信できます。インスタンスがオフラインのときに受信したシグナリングは、オンラインになった後は再度プッシュされません。その他よくあるご質問については、TRTCCalling Webに関するご質問をご参照ください。

技術的なお問い合わせ

詳細については、お問い合わせにご連絡をいただくか、colleenyu@tencent.comにメールでご連絡ください。

参考ドキュメント

- TRTCCalling web公式サイトで体験
- TRTCCalling npm
- TRTCCalling web demoソースコード
- TRTCCalling web API
- TRTCCalling webに関するご質問

©2013-2019 Tencent Cloud. All rights reserved.

TRTCCalling API(Flutter)

最終更新日：：2021-10-12 09:40:09

TRTCCallingはTencent CloudのTencent Real-Time Communication（TRTC）とIMを組み合わせたサービスで、1v1オーディオビデオ通話をサポートします。TRTCCallingは、Tencent Cloudの2つのクローズドソースSDKに依存する1つのオープンソースのClassです。具体的な実装プロセスについては、Real-time Voice Call（Flutter）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオ通話コンポーネントとして使用します。
- IM SDK： IM SDKをシグナリング情報の送信と処理に使用します。

TRTCCalling API概要

SDK基本関数

API	説明
sharedInstance	コンポーネントシングルトン。
destroySharedInstance	コンポーネントシングルトンを廃棄します。
registerListener	イベントコールバックを追加します。
unRegisterListener	コールバックインターフェースを削除します。
destroy	関数の廃棄。このインスタンスを実行する必要がなくなった場合は、このインターフェースをコールしてください。
login	コンポーネントインターフェースへのログイン。すべての機能を使用するためには、まずログインする必要があります。
logout	コンポーネントインターフェースからログアウト。ログアウト後にダイヤル操作はできません。

通話操作に関連するインターフェース関数

API	説明
call	1対1通話に招待します。
accept	現在の通話を受信します。
reject	現在の通話を拒否します。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
hangup	現在の通話を終了します。

オーディオ制御に関するインターフェース関数

API	説明
setMicMute	ローカルオーディオキャプチャをミュートにします。
setHandsFree	ハンズフリーを設定します。

TRTCCallingDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。

招待者のコールバック

API	説明
onReject	通話拒否時のコールバック。
onNoResp	相手方の応答がない場合のコールバック。
onLineBusy	通話中である場合のコールバック。

被招待者のコールバック

API	説明
onInvited	通話に招待された場合のコールバック。
onCallingCancel	現在の通話をキャンセルする場合のコールバック。
onCallingTimeOut	現在の通話がタイムアウトした場合のコールバック。

一般的なコールバック

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onUserEnter	ユーザーが通話に参加した場合のコールバック。
onUserLeave	ユーザーが通話から退出した場合のコールバック。
onUserAudioAvailable	ユーザーがオーディオアップストリームを開始したかどうかのコールバック。
onUserVoiceVolume	ユーザー通話音量のコールバック。
onCallEnd	通話終了時のコールバック。

SDK基本関数

sharedInstance

shareInstanceはTRTCCallingのコンポーネントシングルトンです。

```
static Future<TRTCCalling> sharedInstance();
```

destroySharedInstance

コンポーネントシングルトンを廃棄します。

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

destroy

関数の廃棄。このインスタンスを実行する必要がなくなった場合は、このインターフェースをコールしてください。

```
void destroy();
```

registerListener

TRTCCallingイベントコールバック。TRTCCallingDelegateを介してTRTCCallingの各種ステータス通知を受け取ることができます。

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

unRegisterListener

©2013-2019 Tencent Cloud. All rights reserved.

コールバックインターフェースを削除します。

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

login

コンポーネントにログインします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppID	int	TRTCコンソール > 【アプリケーション管理】> アプリケーション情報でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については UserSigの計算方法をご参照ください。

logout

コンポーネントからログアウトします。

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

通話操作に関連するインターフェース関数

call

1対1通話の招待、現在通話中でも引き続き他のユーザーの招待をコールできます。

```
Future<ActionCallback> call(String userId, int type);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーIDを呼び出します。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
type	int	1は音声通話、2はビデオ通話を意味します。

accept

現在の通話を受信します。被招待者として `onInvited()` のコールバックを受け取った場合は、この関数を呼び出して通話に応答することができます。

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

reject

現在の通話を拒否します。被招待者として `onInvited()` のコールバックを受け取った場合は、この関数を呼び出して電話を拒否することができます。

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

hangup

現在の通話を終了します。通話中である場合は、この関数を呼び出して通話を終了できます。

```
void hangup();
```

オーディオ制御に関するインターフェース関数

setMicMute

ローカルオーディオキャプチャをミュートにします。

```
void setMicMute(bool isMute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMute	bool	trueはマイクの終了を、falseはマイクの起動を意味します。

setHandsFree

リモートの音声をミュートにします。

©2013-2019 Tencent Cloud. All rights reserved.

```
void setHandsFree(bool isHandsFree);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isHandsFree	bool	trueはハンズフリーの起動を、falseはハンズフリーの終了を意味します。

TRTCCallingDelegateイベントコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
msg	String	エラー情報。

招待者のコールバック

onReject

通話拒否時のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	String	拒否したユーザーのID。

onNoResp

相手方が応答しない場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	応答しないユーザーのID。

onLineBusy

通話中である場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話中のユーザーのID。

被招待者のコールバック

onInvited

通話に招待された時のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sponsor	String	発信者のID。
userIds	List&dxlt;String&dxgt;	自分以外の招待IDのリスト。
isFromGroup	bool	多人数の通話への招待かどうか。
type	int	1は音声通話、2はビデオ通話を意味します。

onCallingCancel

©2013-2019 Tencent Cloud. All rights reserved.

現在の通話がキャンセルされた場合のコールバック。被招待者がリクエストを処理しない場合、招待者がキャンセル後にこのコールバックを受け取ります。

onCallingTimeOut

現在の通話がタイムアウトした場合のコールバック。

一般的なコールバック

onUserEnter

ユーザーが通話に参加した場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話に参加したユーザーのID。

onUserLeave

ユーザーが通話から退出した場合のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話から離脱したユーザーのID。

onUserAudioAvailable

ユーザーがオーディオアップストリームを開始したかどうかのコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	通話するユーザーID。
available	boolean	ユーザーのオーディオが使用可能かどうか。

onUserVoiceVolume

ユーザーの通話音量のコールバック。

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userVolumes	List	ディスカッション中のルームの全参加者の音量、値の範囲：0～100。
totalVolume	int	すべてのリモート参加者の総音量。値の範囲：0～100。

onCallEnd

通話終了時のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

ビデオインタラクティブライブストリーミング
ビデオ・インタラクティブストリーミング (iOS)

最終更新日：：2022-03-10 10:33:59

デモンストレーション

ダウンロードからAppをインストールすると、インタラクティブストリーミング機能の効果を体験していただくことができます。これにはマイク接続によるインタラクション、キャスターPK、低レイテンシーの視聴、弾幕チャットなど、TRTCのインタラクティブライブストリーミングシーンにおける関連機能が含まれています。

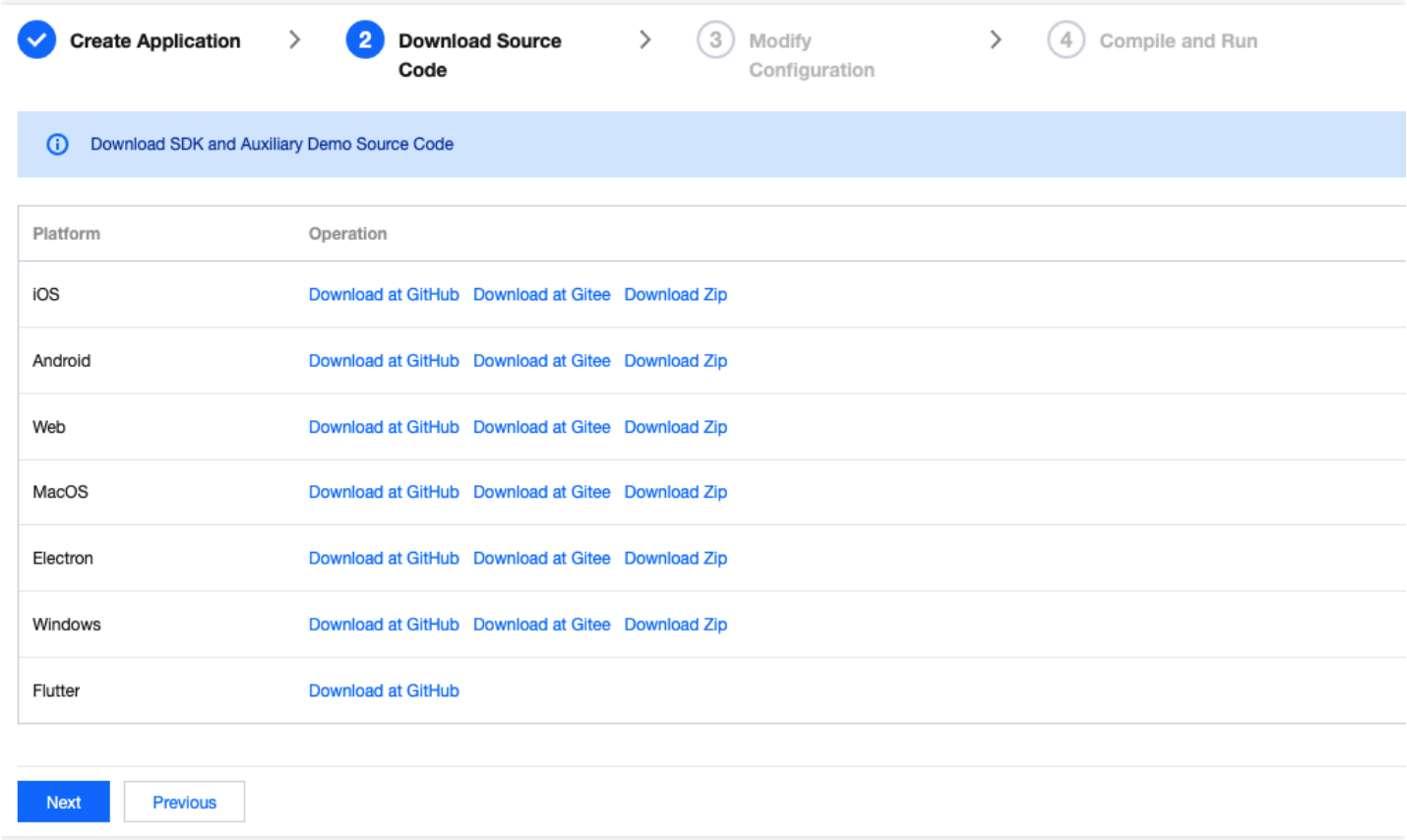
ビデオインタラクティブストリーミングの機能をすばやく実装する必要がある場合、当社が提供するAppをもとに直接変更を加えて適応させるか、当社が提供するTUILiveRoomコンポーネントを使用し、カスタマイズしたUIを実装することができます。

AppのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、【開発支援】>【快速跑通Demo】を選択します。
2. 例えば、`TestLiveRoom` などアプリケーション名を入力して、【作成】をクリックします。
3. 【ダウンロードしました。次のステップ】をクリックすると、この手順をスキップします。

©2013-2019 Tencent Cloud. All rights reserved.



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUILiveRoomに進み、ソースコードをCloneまたはダウンロードします。

手順3：Appプロジェクトファイルの設定

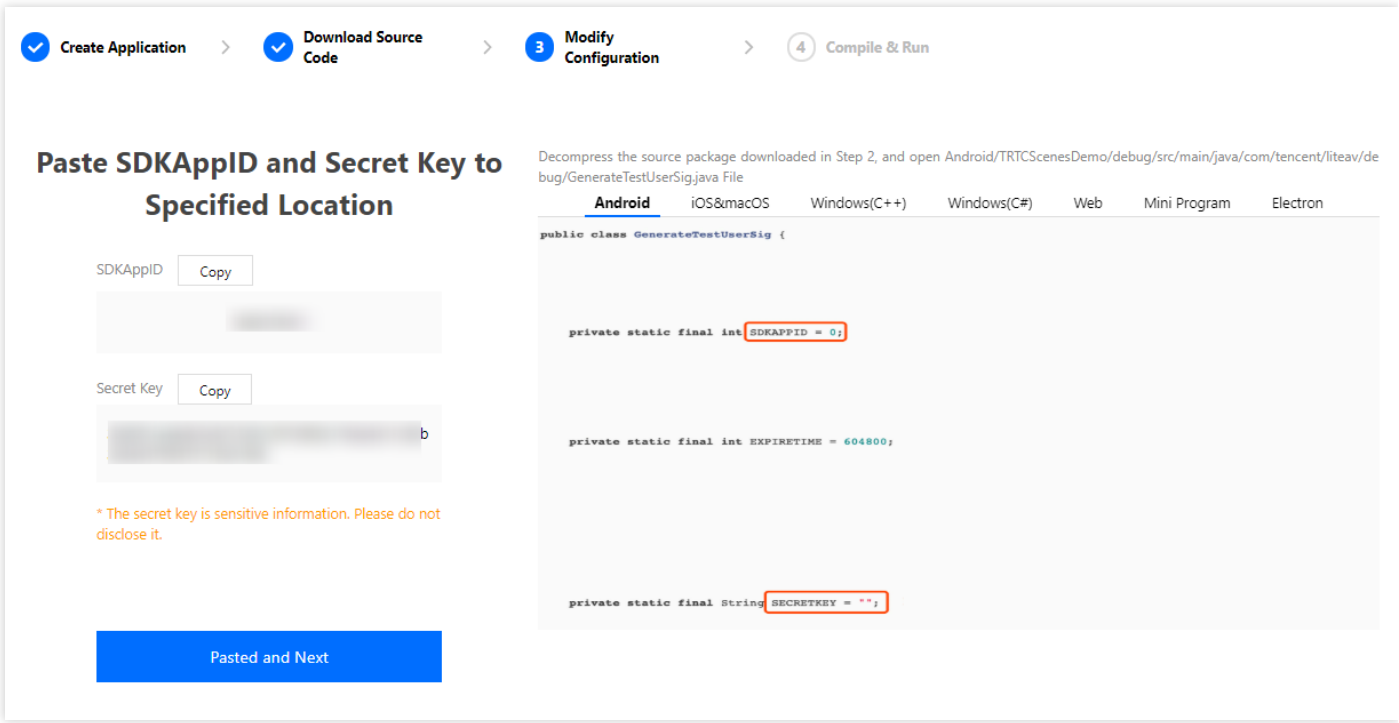
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `TUILiveRoom/Debug/GenerateTestUserSig.swift` ファイルを見つけて開きます。

3. `GenerateTestUserSig.swift` ファイル内の関連パラメータを設定します。

   - SDKAPPID：デフォルトは0。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトは空文字列。実際のキー情報を設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、【貼り付けました。次へ】をクリックすれば、作成が完了します。

5. コンパイル完了後、【コンソールの概要ページに戻る】をクリックすればOKです。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Xcode（11.0以上のバージョン）を使用してソースコードプロセス `TUILiveRoom/TUILiveRoomApp.xcworkspace` を開き、【実行】をクリックすれば、このAppのデバッグが開始されます。

手順5：Appソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードの `Source` フォルダには、uiとmodelの2つのサブフォルダが含まれ、uiフォルダの中はいずれもインターフェースコードです。以下の表には各swiftファイルまたはフォルダおよび対応するUIをリストアップし、二次調整に役立つようにしています。

ファイルまたはフォルダ	機能の説明
Anchor	キャスター側に関するUIの実装コード。
Audience	視聴者側に関するUIの実装コード。
ChatRoom	テキストチャットルームと弾幕メッセージのUI実装コード。
Common	再利用可能なUIコンポーネントの実装コード。
StatusView	ステータスのフローティングレイヤー、ログ情報やビデオのローディングアニメーションを表示するためにビデオ画面上にオーバーレイされます。
LiveRoomMainViewController.swift	ビデオ・インタラクティブストリーミングのトップページ画面UI。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. 【ルームの作成】をクリックします。
3. ルーム名を入力し、【ライブストリーミングを開始】をクリックします。

ユーザーB

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ユーザーAが作成したルーム番号を入力し、【入室する】をクリックします。

> 注意：

©2013-2019 Tencent Cloud. All rights reserved.

ルーム番号は、ユーザーAがルームの作成を完了した後のシステムポップアップボックスから取得します。

ルームのステータスの監視とPKリストへのアクセス

ルームステータスは、次のように `TRTCLiveRoom` を使用して監視することができます。

```
//////////////////////////////////////////////////////////////
//
// ルーム管理関連
//
//////////////////////////////////////////////////////////////
/// ルームを作成（キャスターが呼び出し）します。ルームが存在しない場合、システムが新しいルームを1つ自動作成します。
/// キャスターが配信を開始する際の通常の呼び出しプロセスは以下のとおりです。
/// 1.【キャスター】startCameraPreview()を呼び出し、カメラのプレビューを起動します。この時美顔パラメータを調整できます。
/// 2.【キャスター】 createRoom()を呼び出し、ライブストリーミングルームを作成します。ルーム作成の成功の有無がcallbackでキャスターに通知されます。
/// 3.【キャスター】startPublish() を呼び出し、プッシュを開始します。
/// - Parameters:
/// - roomID: ルームIDは、ご自身でアサインし、一元管理する必要があります。複数の roomID を、1つのライブストリーミングルームリストにまとめることができます。Tencent Cloudではライブストリーミングルームリストの管理サービスを一時的に行いませんので、ご自身でライブルームリストを管理してください。
/// - roomParam: TRTCCreateRoomParam | ルーム情報は、ルームの説明情報として利用します（例：ルーム名、カバー情報など）。ルームリストとルーム情報をご自分で管理する場合は、このパラメータは無視してかまいません。
/// - callback: 入室結果のコールバック。成功時にcodeは0になります。
/// - Note:
/// - キャスターがライブストリーミングを開始する時のコールでは、自分が作成したことがあるルームを再び作成することができます。
- (void)createRoomWithRoomID:(UInt32)roomID
roomParam:(TRTCCreateRoomParam *)roomParam
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(createRoom(roomID:roomParam:callback:));
/// ルームの破棄（キャスターが呼び出し）
/// キャスターはルーム作成後、この関数を呼び出してルームを破棄することができます。
/// - Parameter callback: ルーム破棄結果のコールバック。成功時にcodeは0になります。
/// - Note:
/// - キャスターは、ルーム作成後、この関数を呼び出してルームを破棄します。
- (void)destroyRoom:(Callback _Nullable)callback
NS_SWIFT_NAME(destroyRoom(callback:));
/// ルームリストの詳細情報を取得します。
/// その中の情報は、キャスターがcreateRoom()作成時に roomInfoで設定します。ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。
```

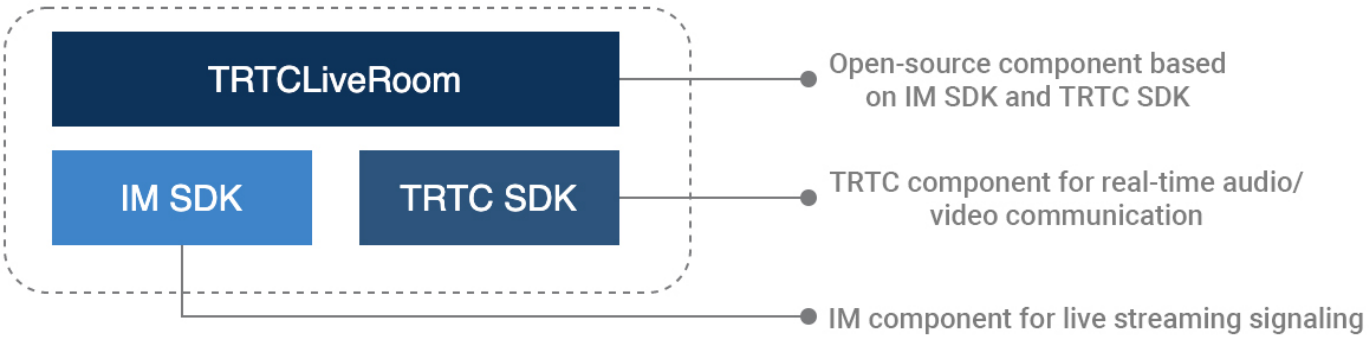
©2013-2019 Tencent Cloud. All rights reserved.

```
/// - Parameter roomIDs: ルームナンバーリスト
/// - Parameter callback: ルーム詳細情報のコールバック
- (void)getRoomInfosWithRoomIDs:(NSArray<nsnumber *=""> *)roomIDs
callback:(RoomInfoCallback _Nullable)callback
NS_SWIFT_NAME(getRoomInfos(roomIDs:callback:));
```

ユーザーのアクセスを容易にするため、Callback方式が採用されています。例えば、ルームリストの取得には非同期操作が必要な場合がありますが、Callback方式を採用すればよりフレキシブルになります。

UIカスタマイズの実装

ソースコードの `Source` フォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTRTCLiveRoomが含まれています。 `TRTCLiveRoom.h` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタマイズしたUIを実装することができます。



手順1：SDKの統合

オーディオビデオ通話コンポーネントTRTCLiveRoomは、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

- **方法1：cocoapodsリポジトリを介する依存**

```
pod 'TXIMSDK_iOS'
pod 'TXLiteAVSDK_TRTC'
```

- **方法2：ローカルを介する依存**
  開発環境でのcocoapodsリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

©2013-2019 Tencent Cloud. All rights reserved.

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定

info.plistファイルに `Privacy > Camera Usage Description` , `Privacy > Microphone Usage Description` を追加し、カメラとマイクの権限をリクエストする必要があります。

ステップ3：TUILiveRoomコンポーネントのインポート

cocoapodsによってコンポーネントをインポートします。具体的な手順については、以下のとおりです。

1. プロジェクトディレクトリの `Source` 、 `Resources` 、 `TCBeautyKit` 、 `TXAppBasic` フォルダ、 `TUILiveRoom.podspec` ファイルをお客様のプロジェクトディレクトリ下へコピーします。
2. `Podfile` ファイル内に以下の依存関係を追加します。その後、 `pod install` コマンドを実行すると、インポートが完了します。

```
pod 'TXAppBasic', :path =&gt; "TXAppBasic/"
pod 'TCBeautyKit', :path =&gt; "TCBeautyKit/"
pod 'TXLiteAVSDK_TRTC'
pod 'TUILiveRoom', :path =&gt; "./", :subspecs =&gt; ["TRTC"]
```

手順4：コンポーネントの作成およびログイン

1. TRTCLiveRoomIの `init` インターフェースを呼び出すと、TRTCLiveRoomコンポーネントのインスタンスオブジェクトを作成することができます。
2. 1つの `TRTCLiveRoomConfig` オブジェクトを作成したら、このオブジェクトにuseCDNFirstとCDNPlayDomainの属性を設定することができます。

- useCDNFirstの属性：視聴者の視聴方式の設定に使用します。trueは、一般視聴者のCDN経由での視聴を表し、料金は安価ですがレイテンシーは高めです。falseは、一般視聴者の低レイテンシーによる視聴を表し、料金はCDN とマイク接続の間ですが、遅延を1s以内に抑えることができます。
- CDNPlayDomainの属性：useCDNFirstの設定をtrueにした時に有効になり、CDN経由の視聴の再生ドメイン名の指定に利用します。CSSコンソール >【ドメイン名管理】ページからログインし、設定することができます。

3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	機能

©2013-2019 Tencent Cloud. All rights reserved.

sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
config	グローバルコンフィギュレーション情報。ログイン時に初期化してください。ログイン後は変更できなくなります。 4. useCDNFirstの属性：視聴者の視聴方式の設定に使用します。trueは、一般視聴者のCDN経由での視聴を表し、料金は安価ですがレイテンシーは高めです。falseは、一般視聴者の低レイテンシーによる視聴を表し、料金は CDN とマイク接続の間ですが、遅延を1s以内に抑えることができます。 5. CDNPlayDomainの属性：useCDNFirstの設定をtrueにした時に有効になり、CDN経由の視聴の再生ドメイン名の指定に利用します。CSSコンソール>【ドメイン名管理】ページからログインして設定を行うことができます。
callback	ログインのコールバック。成功時にcodeは0になります。

```
class LiveRoomController: UIViewController {
let mLiveRoom = TRTCLiveRoom()
}
//useCDNFirst：trueは一般視聴者のCDN経由の視聴、falseは一般視聴者の低レイテンシーによる視聴を表します
//CDNPlayDomain：CDN視聴時に設定する再生ドメイン名を表します
let config = TRTCLiveRoomConfig(useCDNFirst: useCDNFirst, cdnPlayDomain: yourCDNPlayDomain)
mLiveRoom.login(SDKAPPID, userID, userSig, config) { (code, error) in
if code == 0 {
//ログイン成功
}
}
```

手順5：キャスター側での配信開始

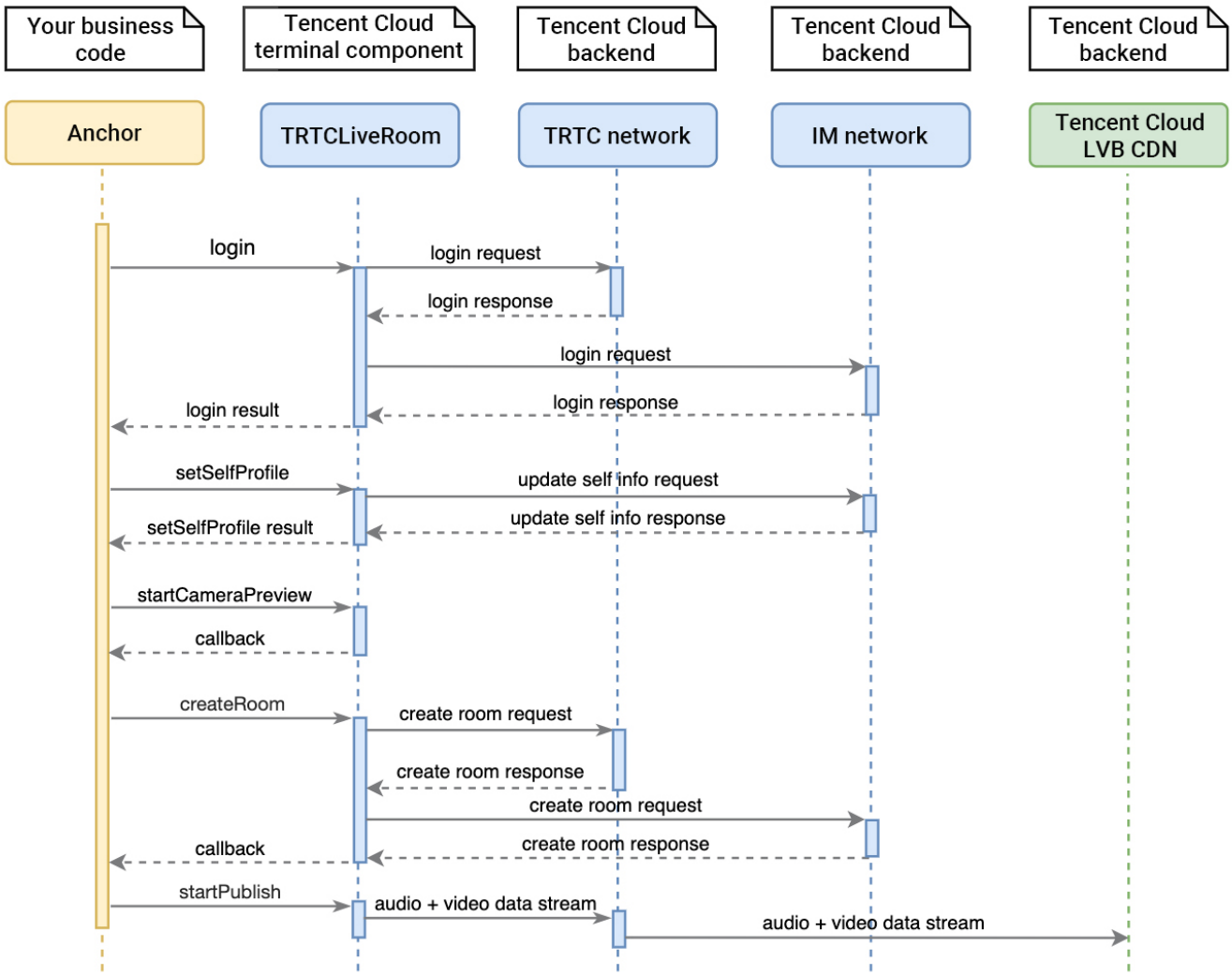
1. キャスターは、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. キャスターは、配信開始前に先に `startCameraPreview` を呼び出してカメラのプレビューを起動させ、インターフェース上に美顔調節ボタンを設置して呼び出し、 `getBeautyManager` で美顔設定を行うことができます。

説明：

©2013-2019 Tencent Cloud. All rights reserved.

エンタープライズ版以外のSDKは美顔やスタンプなどの高度な美顔機能をサポートしていません。

3. 美顔効果の変更後、キャスターは `createRoom` を呼び出して新しいライブストリーミングルームを作成することができます。
4. キャスターが `startPublish` を呼び出し、プッシュを開始します。CDN視聴のサポートが必要な場合は、login 時に渡される `TRTCLiveRoomConfig` パラメータ内で `useCDNFirst` と `CDNPlayDomain` を指定し、 `startPublish` 時にライブストリーミング・プルストリーム用のstreamIDを指定してください。




```
// 1. キャスターは、ニックネームおよびプロフィール画像を設定します
mLiveRoom.setSelfProfile(name: "A", avatarURL: "faceUrl", callback: nil)
// 2. キャスターが配信開始前にプレビューを立ち上げ、美顔パラメータを設定します
let view = UIView()
parentView.add(view)
```

©2013-2019 Tencent Cloud. All rights reserved.

```
mLiveRoom.startCameraPreview(frontCamera: true, view: view, callback: nil)
mLiveRoom.getBeautyManager().setBeautyStyle(.nature)
mLiveRoom.getBeautyManager().setBeautyLevel(6)
// 3.キャスターがルームを作成します
let param = TRTCCreateRoomParam(roomName: "テストルーム", coverUrl: "")
mLiveRoom.createRoom(roomID: 123456789, roomParam: param) { [weak self] (code, error) in
if code == 0 {
// 4.キャスターがプッシュを開始し、ストリームがCDNに配信されます。
self?.mLiveRoom.startPublish(streamID: mSelfUserId + "_stream", callback: nil)
}
}
```

手順6：視聴者側での視聴

1. 視聴者側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 視聴者側が業務バックエンドから最新のライブストリーミングルームリストを取得します。

> 説明：
> App内のライブストリーミングルームリストはデモに使用するためだけのものです。ライブストリーミングルームリストの業務ロジックは千差万別です。現在、Tencent Cloudはライブストリーミングルームリスト管理のサービスを提供していません。各自でご自分のライブストリーミングルームリストを管理してください。

3. 視聴者側は、 `getRoomInfos` を呼び出してルームの詳細情報を取得します。この情報は、キャスター側が `createRoom` を呼び出してライブストリーミングルームを作成するときに設定する簡単な説明情報です。

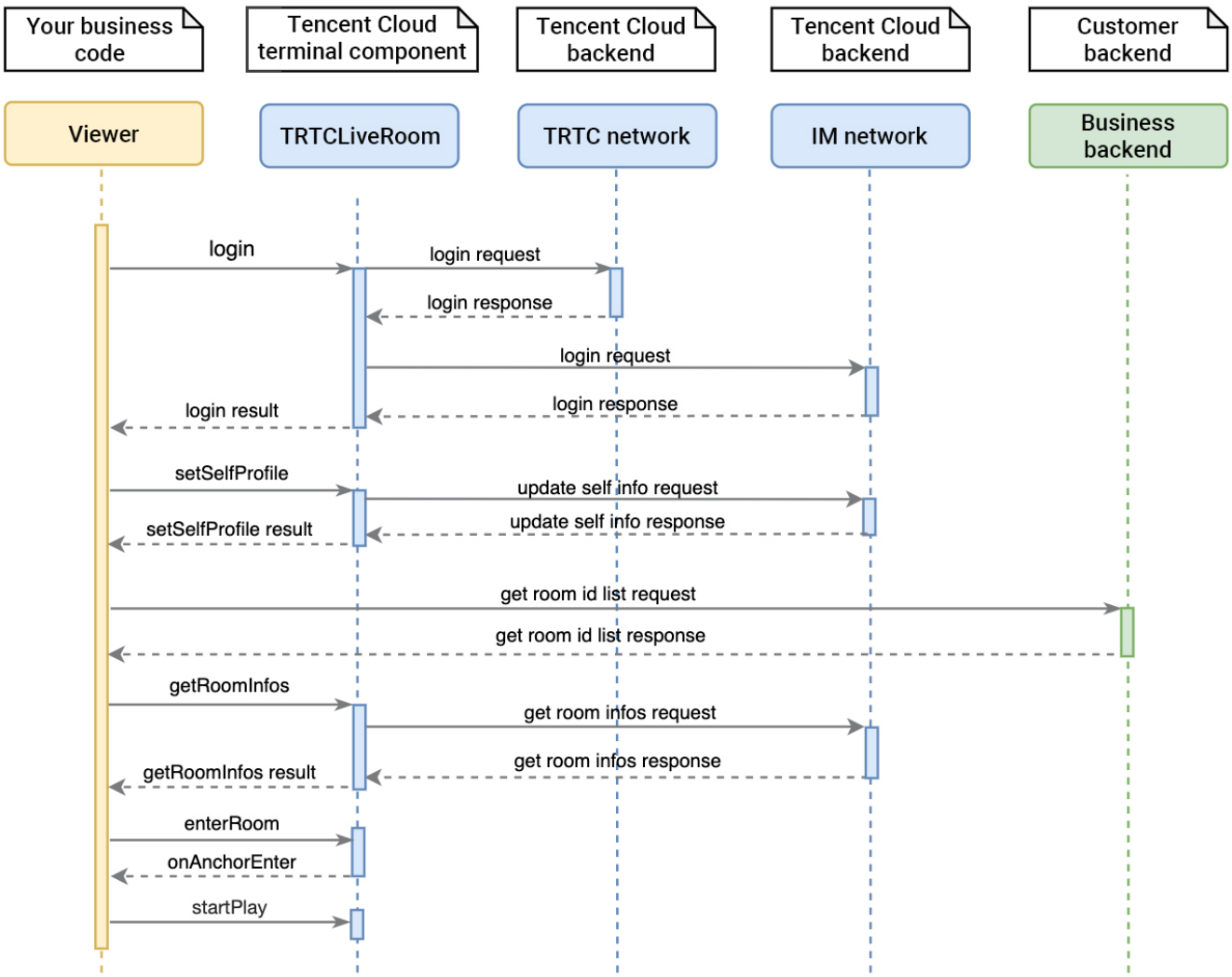
> 注意：
> ライブストリーミングルームのリストに十分に包括的な情報がある場合は、 `getRoomInfos` の呼び出しに関する手順をスキップできます。

4. 視聴者は1つのライブストリーミングルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
5. `startPlay` を呼び出してキャスターのuserIdを渡し、再生を開始します。

- ライブストリーミングルームリストにキャスターのuserID情報がすでに含まれている場合、視聴者は直接 `startPlay` を呼び出してキャスターのuserIdを渡せば、再生を開始できます。

©2013-2019 Tencent Cloud. All rights reserved.

- ルーム参加前にキャスターのuserIDが一時的に取得できない場合、視聴者側はルーム参加後にキャスターの `onAnchorEnter` のイベントコールバックを受信します。このコールバックの中にキャスターのuserID情報が含まれていますので、 `startPlay` を呼び出せば再生できます。



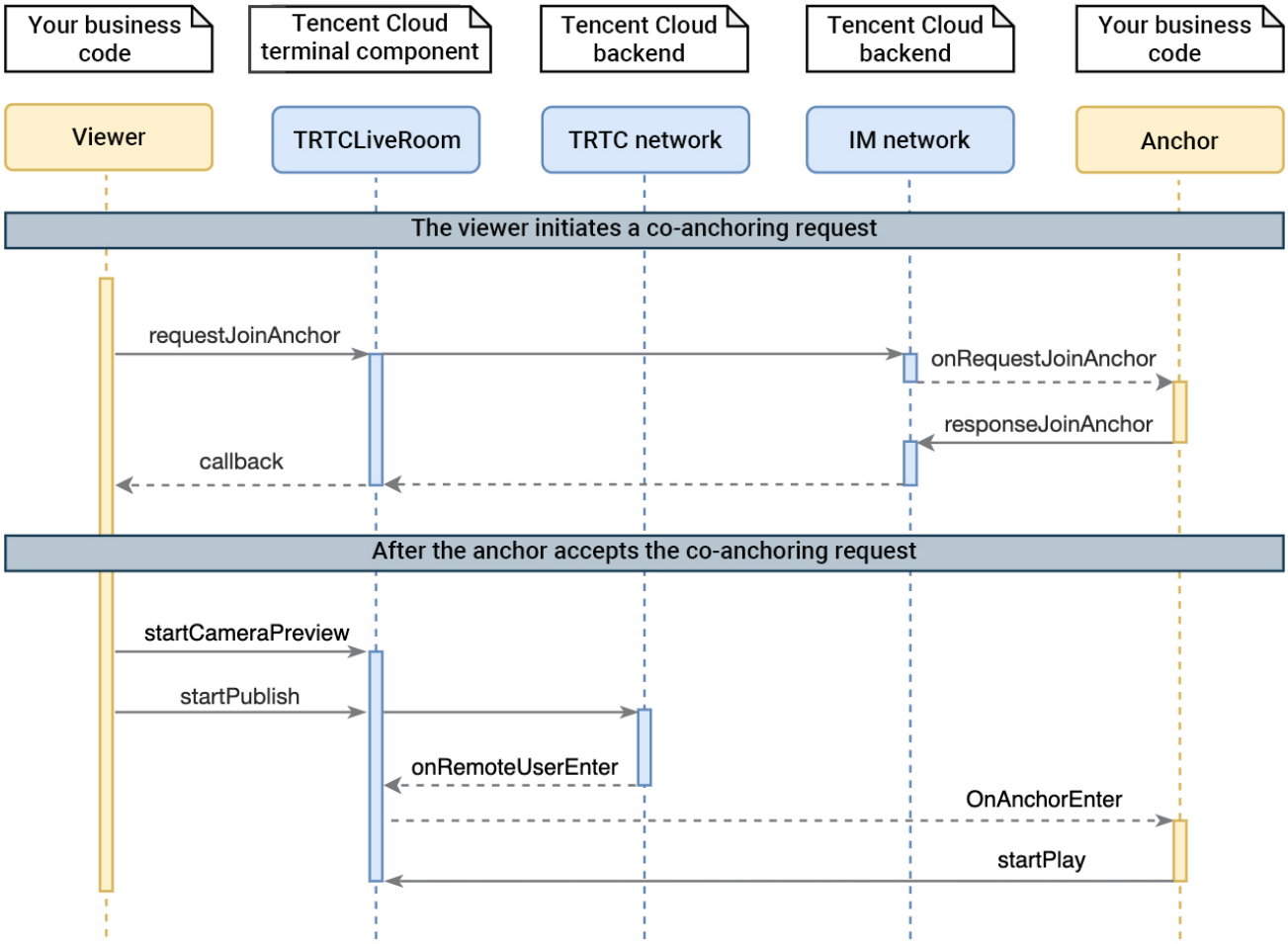

```
// 1.業務バックエンドから取得したルームリストをroomListと仮定します
var roomList: [UInt32] = GetRoomList()
// 2. getRoomInfosを呼び出すことによって、ルームの詳細情報を取得します
mliveRoom.getRoomInfos(roomIDs: roomList, callback: { (code, msg, list) in
if code == 0 {
// ルーム詳細情報の取得後、キャスターリストページでキャスターのニックネーム、プロフィール画像等の関連情報を表示できます
}
})
// 3.roomidを選択しルームに参加します
mliveRoom.enterRoom(roomID: roomID, callback: callback)
```

```
// 4.視聴者がキャスターのルーム参加の通知を受信すると、再生が開始されます
public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onAnchorEnter userID: String) {
// 5.視聴者がキャスターの画面を再生します
mliveRoom.startPlay(userID: userID, view: renderView, callback: nil)
}
```

手順7：視聴者とキャスターとのマイク接続

1. 視聴者側が `requestJoinAnchor` を呼び出し、キャスター側にマイク接続のリクエストを送信します。
2. キャスター側は `TRTCLiveRoomDelegate#onRequestJoinAnchor` （視聴者からのマイク接続のリクエストがあります）のイベント通知を受信します。
3. キャスター側は `responseJoinAnchor` を呼び出し、視聴者からのマイク接続リクエストを受け入れるかどうか決定できます。
4. 視聴者側が `TRTCLiveRoomDelegate#responseCallback` のイベント通知を受信します。この通知にはキャスターの処理結果が含まれています。
5. キャスターがマイク接続リクエストに同意したら、視聴者側は `startCameraPreview` を呼び出し、ローカルカメラを起動させます。その後 `startPublish` を呼び出し、視聴者側のプッシュを開始します。
6. キャスター側は、視聴者側の起動の通知の後に、 `TRTCLiveRoomDelegate#onAnchorEnter` （別のオーディオビデオストリーミングが届いています）の通知を受信します。この通知には視聴者側のuserIdが含まれています。
7. キャスター側が `startPlay` を呼び出すと、マイク接続の視聴者の画面を見ることができます。

©2013-2019 Tencent Cloud. All rights reserved.



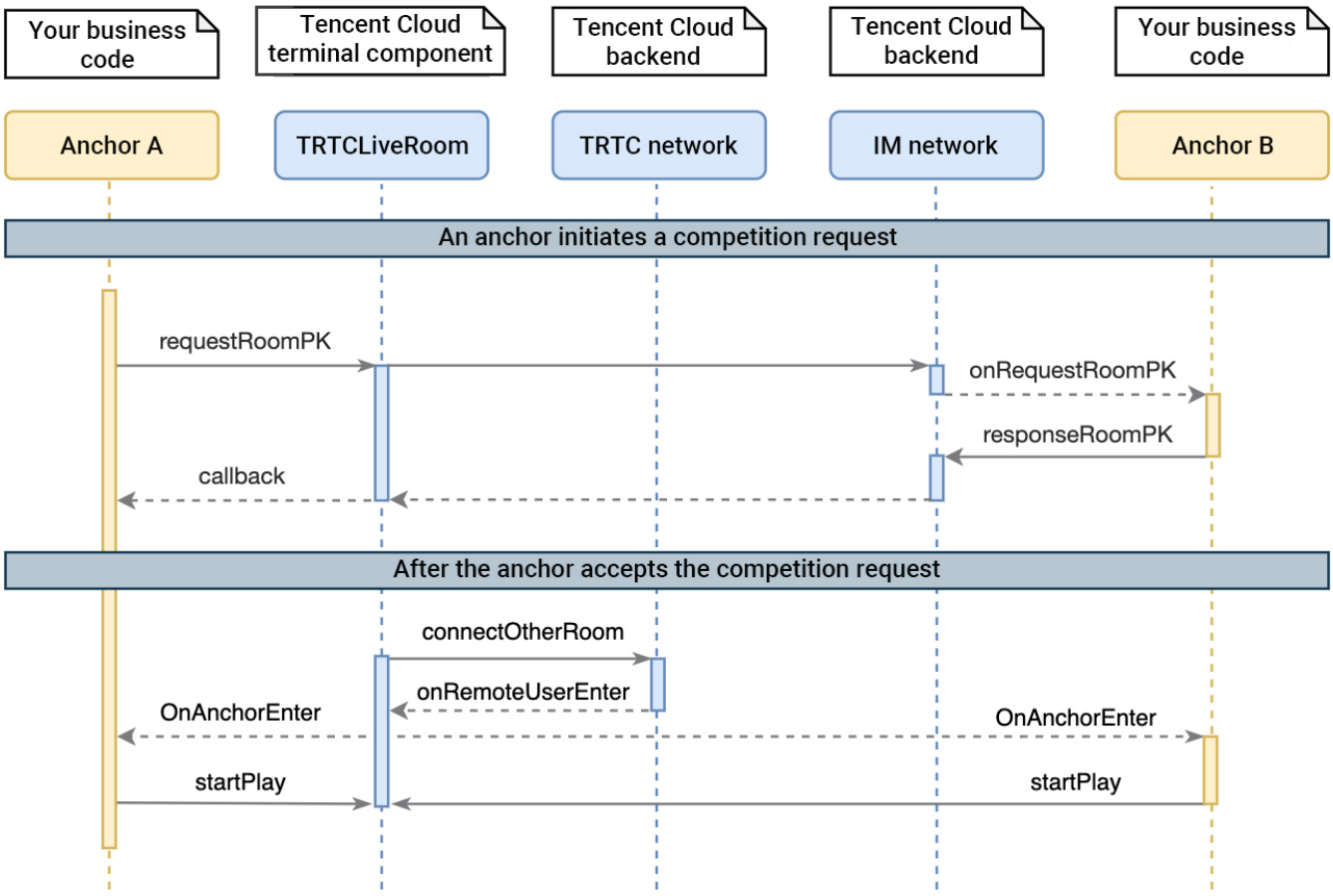

```
// 視聴者側：
// 1.視聴者側がマイク接続のリクエストを送信します
mliveRoom.requestJoinAnchor(reason: mSelfUserId + "あなたとのマイク接続をリクエスト", responseCal
lback: { [weak self] (agreed, reason) in
// 4.キャスターが視聴者のリクエストを受け入れます
if agreed {
// 5.視聴者がプレビューを起動し、プッシュを開始します
self?.mliveRoom.startCameraPreview(frontCamera: true, view: localView, callback: nil)
self?.mliveRoom.startPublish(streamID: streamID, callback: nil)
}
}, callback: callback)
// キャスター側：
// 2.キャスター側がマイク接続のリクエストを受信します
public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onRequestJoinAnchor user: TRTCLiveUserInfo
, reason: String?, timeout: Double) {
// 3.相手側のマイク接続のリクエストに同意します
mliveRoom.responseJoinAnchor(userID: userID, agree: true, reason: "マイク接続に同意")
}
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// 6.キャスターがマイク接続の視聴者のマイク・オンの通知を受信します
public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onAnchorEnter userID: String) {
// 7.キャスターが視聴者の画面を再生します
mliveRoom.startPlay(userID: userID, view: view, callback: nil)
}
```

手順8：キャスターとキャスターPK

1. キャスターAが `requestRoomPK` を呼び出し、キャスターBにPKのリクエストを送信します。
2. キャスターBが `TRTCLiveRoomDelegate onRequestRoomPK` のコールバック通知を受信します。
3. キャスターBが `responseRoomPK` を呼び出し、キャスターA のPKのリクエストを受け入れるか決定します。
4. キャスターBは、キャスターAのリクエストを受け入れたら、 `TRTCLiveRoomDelegate onAnchorEnter` の通知を待ってから、 `startPlay` を呼び出し、キャスターAを表示します。
5. PKのリクエストが同意されたかについて、キャスターAが `responseCallback` のコールバック通知を受信します。
6. キャスターAのリクエストが同意されたら、 `TRTCLiveRoomDelegate onAnchorEnter` の通知を待ってから、 `startPlay` を呼び出し、キャスターBを表示します 。



©2013-2019 Tencent Cloud. All rights reserved.

```
// キャスターA:
// キャスターAがルーム12345を作成します
mLiveRoom.createRoom(roomID: 12345, roomParam: param, callback: nil)

// 1.キャスターAがキャスターBにPKのリクエストを送信します
mLiveRoom.requestRoomPK(roomID: 54321, userID: "B", responseCallback: { (agree, reason) in
// 5.同意の有無のコールバックを受信します
if agree {
}
}, callback: callback)

// キャスターAがキャスターBのルーム参加のコールバックを受信します
public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onAnchorEnter userID: String) {
// 6.Bのルーム参加の通知を受信し、Bの画面を再生します
mLiveRoom.startPlay(userID: userID, view: view, callback: callback)
}

// キャスターB:
// キャスターBがルーム54321を作成します
mLiveRoom.createRoom(roomID: 54321, roomParam: param, callback: nil)

// 2.キャスターBがキャスターAのメッセージを受信します
public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onRequestRoomPK user: TRTCLiveUserInfo, ti
meout: Double) {
// 3.キャスターBがキャスターAに回答し、リクエストを受け入れます
mLiveRoom.responseRoomPK(userID: userID, agree: true, reason: reason)
}

public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onAnchorEnter userID: String) {
// 4.キャスターBがキャスターAのルーム参加の通知を受信し、キャスターAの画面を再生します
mLiveRoom.startPlay(userID: userID, view: view, callback: callback)
}
```

手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、該当するルーム内の全てのキャスターおよび視聴者が `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

```
// 送信側：テキストメッセージの送信
mLiveRoom.sendRoomTextMsg(message: "Hello Word!", callback: callback)
// 受信側：テキストメッセージのモニタリング
mLiveRoom.delegate = self
```

©2013-2019 Tencent Cloud. All rights reserved.

```
public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onRecvRoomTextMsg message: String, from
User user: TRTCLiveUserInfo) {
debugPrint("¥(user.userName)から受信したテキストメッセージ:¥(message)")
}
```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターと視聴者が `onRecvRoomCustomMsg` コールバックを受信できます。
  カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

```
// 送信側：カスタマイズしたCmdで弾幕と「いいね！」のメッセージを区別することができます
// eg:「CMD_DANMU」は弾幕コメントを表し、「CMD_LIKE」は「いいね」情報を表します
mLiveRoom.sendRoomCustomMsg(command: "CMD_DANMU", message: "Hello world", callback: nil)
mLiveRoom.sendRoomCustomMsg(command: "CMD_LIKE", message: "", callback: nil)
// 受信側：カスタムメッセージのモニタリング
mLiveRoom.delegate = self
public func trtcLiveRoom(_ trtcLiveRoom: TRTCLiveRoom, onRecvRoomCustomMsg command: String, me
ssage: String, fromUser user: TRTCLiveUserInfo) {
if "CMD_DANMU" == command {
// 弾幕コメントの受信
debugPrint("¥(user.userName)から受信した弾幕のメッセージ:¥(message)")
} else if "CMD_LIKE" == command {
// 「いいね」情報の受信
debugPrint("¥(user.userName)がいいねを付けました")
}
}
```

©2013-2019 Tencent Cloud. All rights reserved.

ビデオ・インタラクティブストリーミング (Android)

最終更新日：：2022-03-10 10:44:24

デモンストレーション

ダウンロードからAppをインストールすると、インタラクティブストリーミング機能の効果を体験していただくことができます。これにはマイク接続によるインタラクション、キャスターPK、低レイテンシーの視聴、弾幕チャットなど、TRTCのインタラクティブライブストリーミングシーンにおける関連機能が含まれています。

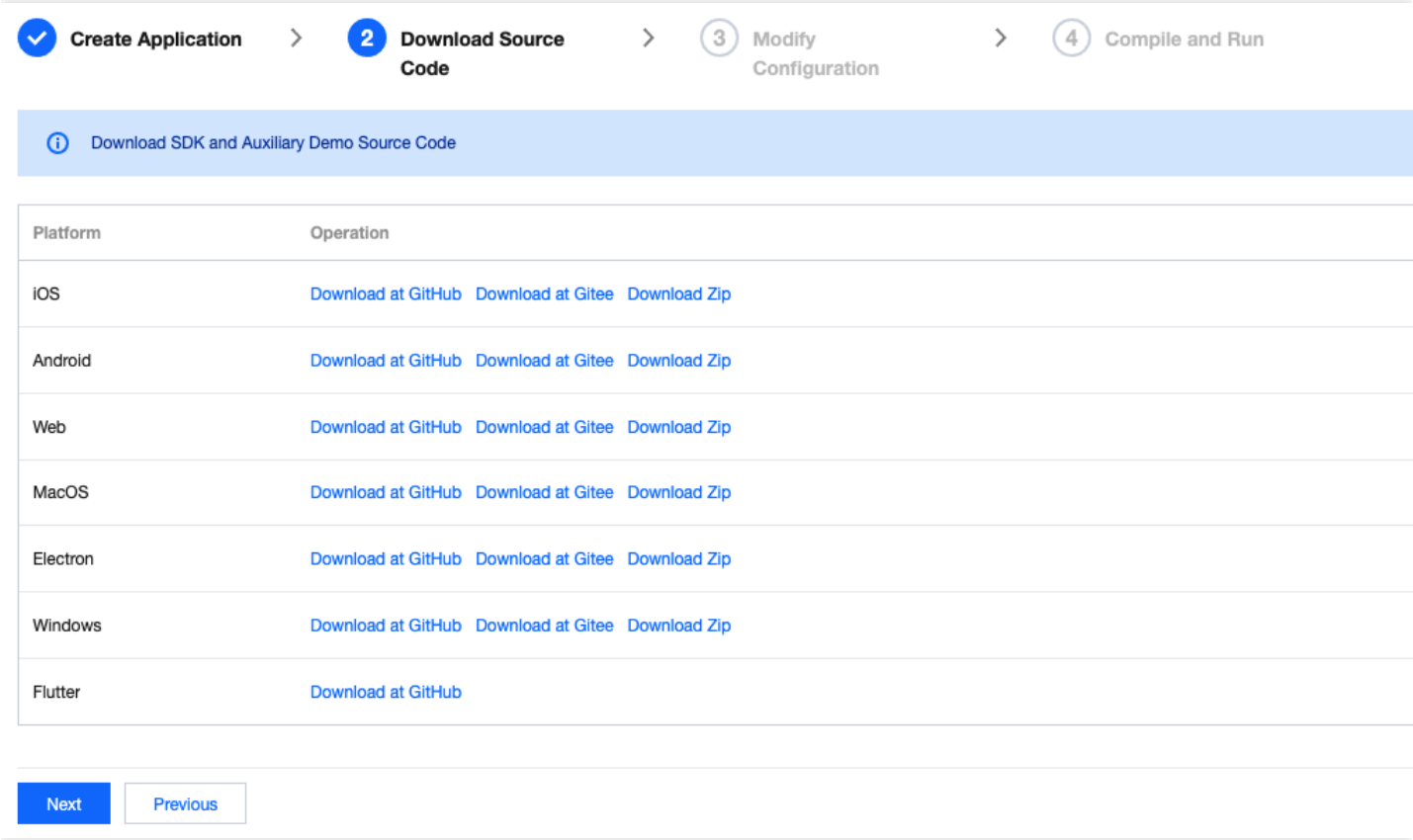
ビデオ・インタラクティブストリーミングの機能をすばやく実装する必要がある場合、当社が提供するAppをもとに直接修正を加えてフィットさせることも、当社が提供するTRTCLiveRoomコンポーネントを使用し、カスタマイズしたUIを実現することも可能です。

AppのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、【開発支援】>【快速跑通Demo】を選択します。
2. 例えば、 `TestLiveRoom` などアプリケーション名を入力して、【作成】をクリックします。
3. 【ダウンロードしました。次のステップ】をクリックすると、この手順をスキップします。

©2013-2019 Tencent Cloud. All rights reserved.



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期してアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUILiveRoomに進み、ソースコードをCloneまたはダウンロードします。

手順3：Appプロジェクトファイルの設定

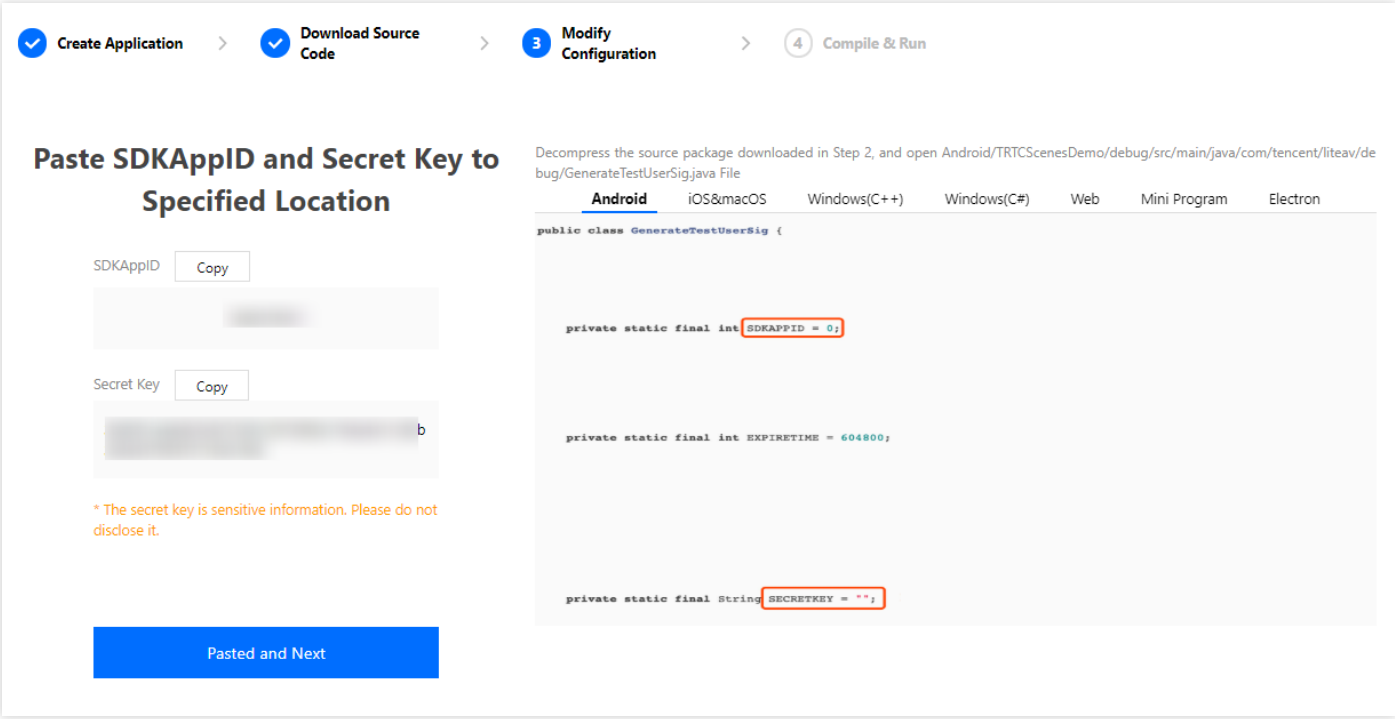
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `Android/Debug/src/main/java/com/tencent/liteav/debug/GenerateTestUserSig.java` のファイルを見つけて開きます。

3. `GenerateTestUserSig.java` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはプレースホルダー(PLACEHOLDER)。実際のSDKAppIDを設定してください。

©2013-2019 Tencent Cloud. All rights reserved.

- SECRETKEY：デフォルトはプレースホルダー(PLACEHOLDER)。実際のキー情報を設定してください。



4. 貼り付け完了後、【貼り付けました。次へ】をクリックすれば、作成が完了します。

5. コンパイル完了後、【コンソールの概要ページに戻る】をクリックすればOKです。

注意：
ここで言及したUserSigの発行方法は、クライアントコードにSECRETKEYを設定しますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
≻ UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。 UserSigが必要なときは、Appから業務サーバーにリクエストを送信し動的にUserSigを取得します。詳細はサーバーでのUserSig新規作成をご参照ください。

手順4：Appの実行

Android Studio（バージョン3.5以上）を使用してソースコード `TUILiveRoom` プロジェクトを開き、【実行】をクリックすれば、このAppのデバッグが開始されます。

手順5：Appソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードのSourceフォルダは、uiとmodelという2つのサブフォルダを含んでいます。uiフォルダにはインターフェースコードが含まれています。以下の表にファイルまたはフォルダおよび対応するUIをリストアップして、二次調整を行いやすくしています。

ファイルまたはフォルダ	機能の説明
anchor	キャスター側に関するUIの実装コード。
audience	視聴者側に関するUIの実装コード。
common	一般的なUIコンポーネントの実装コード。
widget	汎用ウィジェット。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ログイン後、【ルームの作成】をクリックします。
3. ルーム名を入力し、【ライブストリーミングを開始】をクリックします。

ユーザーB

1. ユーザー名を入力します（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）。
2. ユーザーAが作成したルームナンバーを入力し、入室するをクリックします。

> 注意：
> ルームナンバーはユーザーAのルーム上部に表示されます。

ルームのステータスの監視とPKリストへのアクセス

ルームのステータスは、次のように `LiveRoomManager` を使用して監視することができます。

©2013-2019 Tencent Cloud. All rights reserved.

```
LiveRoomManager.getInstance().addCallback(new LiveRoomManager.RoomCallback() {
/**
* ルームの作成
* @param roomId
* @param callbackは、内部処理の結果を通知するために使用します
*/
@Override
public void onRoomCreate(int roomId, final LiveRoomManager.ActionCallback callback) {
// doSomething
}

/**
* ルームの破棄
* @param roomId
* @param callbackは、内部処理の結果を通知するために使用します
*/
@Override
public void onRoomDestroy(int roomId, final LiveRoomManager.ActionCallback callback) {
// doSomething
}

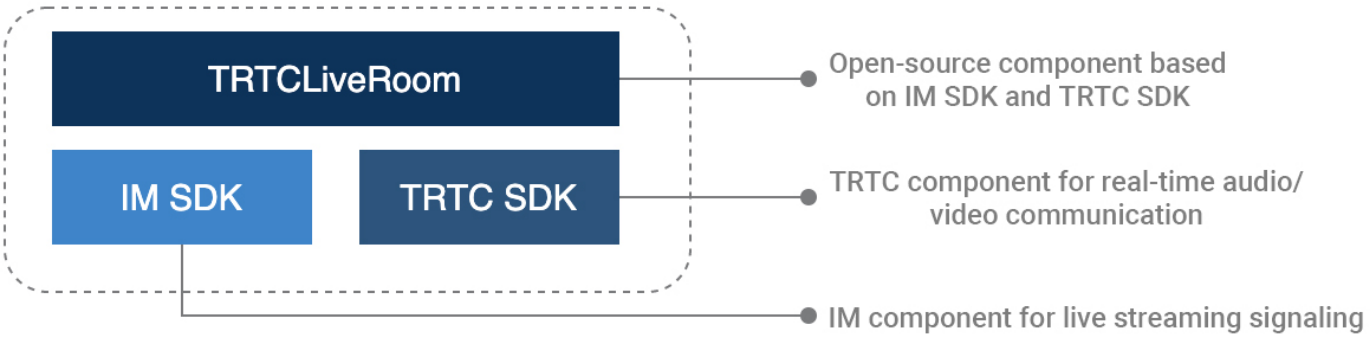
/**
* ルームリストの取得
* @param callback
*/
@Override
public void onGetRoomIdList(final LiveRoomManager.GetCallback callback) {
// ルームリストを取得します。ご自分で管理する必要があります。ユーザー間のPKに使用します
if(callback != null) {
if(success) {
// コールバックに成功しました。ルームリストが渡されます
callback.onSuccess(roomList);
}else{
// ルームリストの取得に失敗しました
callback.onError(code, message);
}
}
}
});
```

ユーザーのアクセスを容易にするため、Callback方式が採用されています。例えば、ルームリストの取得には非同期操作が必要な場合がありますが、Callback方式を採用すればよりフレキシブルになります。

UIカスタマイズの実装

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードのSourceフォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTRTCLiveRoomがあります。 `TRTCLiveRoom.java` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタマイズしたUIを実装することができます。



手順1：SDKの統合

ビデオ・インタラクティブストリーミングコンポーネントTRTCLiveRoomは、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

方法1：Mavenリポジトリを介する依存

1. dependenciesにTRTCSDKとIMSDKの依存を追加します。

```
dependencies {
compile "com.tencent.liteav:LiteAVSDK_TRTC:latest.release"
compile 'com.tencent.imsdk:imsdk:latest.release'
}
```

2. defaultConfigでAppが使用するCPUアーキテクチャを指定します。

```
defaultConfig {
ndk {
abiFilters "armeabi-v7a"
}
}
```

3. 【Sync Now】をクリックし、自動でSDKをダウンロードし、プロジェクトに統合します。

©2013-2019 Tencent Cloud. All rights reserved.

方法2：ローカルAARを介する依存

開発環境でのMavenリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定および難読化ルール

AndroidManifest.xmlにAppの権限を設定します。SDKには次の権限が必要です（6.0以上のAndroidシステムではカメラ、ストレージ読み取りの権限を動的に申請する必要があります）。

```
<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">
```

proguard-rules.proファイルでは、SDK関連を非難読化リストに追加します。

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

手順3：TRTCLiveRoomコンポーネントをインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
src/main/java/com/tencent/liteav/liveroom/model
```

手順4：コンポーネントの作成およびログイン

1. sharedInstance インターフェースを呼び出すと、TRTCLiveRoomコンポーネントのインスタンスオブジェクトを作成できます。

©2013-2019 Tencent Cloud. All rights reserved.

2. 1つの TRTCLiveRoomConfig オブジェクトを作成したら、このオブジェクトにuseCDNFirstとCDNPlayDomainの属性を設定することができます。

- useCDNFirstの属性：視聴者の視聴方式の設定に使用します。trueは、一般視聴者のCDN経由での視聴を表し、料金は安価ですがレイテンシーは高めです。falseは、一般視聴者の低レイテンシーによる視聴を表し、料金はCDN とマイク接続の間ですが、遅延を1s以内に抑えることができます。
- CDNPlayDomainの属性：useCDNFirstの設定をtrueにした時に有効になり、CDN経由の視聴の再生ドメイン名の指定に利用します。CSSコンソール >【ドメイン名管理】ページからログインし、設定することができます。

3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	作用
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
config	グローバルコンフィギュレーション情報。ログイン時に初期化してください。ログイン後は変更できなくなります。 4. useCDNFirstの属性：視聴者の視聴方式の設定に使用します。trueは、一般視聴者のCDN経由での視聴を表し、料金は安価ですがレイテンシーは高めです。falseは、一般視聴者の低レイテンシーによる視聴を表し、料金は CDN とマイク接続の間ですが、遅延を1s以内に抑えることができます。 5. CDNPlayDomainの属性：useCDNFirstの設定をtrueにした時に有効になり、CDN経由の視聴の再生ドメイン名の指定に利用します。CSSコンソール>【ドメイン名管理】ページからログインして設定を行うことができます。
callback	ログインのコールバック。成功時にcodeは0になります。

```
TRTCLiveRoom mLiveRoom = TRTCLiveRoom.sharedInstance(this);
//useCDNFirst：trueは一般視聴者のCDN経由の視聴、falseは一般視聴者の低レイテンシーによる視聴を表します
//yourCDNPlayDomain：CDN視聴時に設定する再生ドメイン名を表します。
TRTCLiveRoomDef.TRTCLiveRoomConfig config =
new TRTCLiveRoomDef.TRTCLiveRoomConfig(useCDNFirst, yourCDNPlayDomain);
mLiveRoom.login(SDKAPPID, userId, userSig, config,
new TRTCLiveRoomCallback.ActionCallback() {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//ログイン成功
}
}
});
```

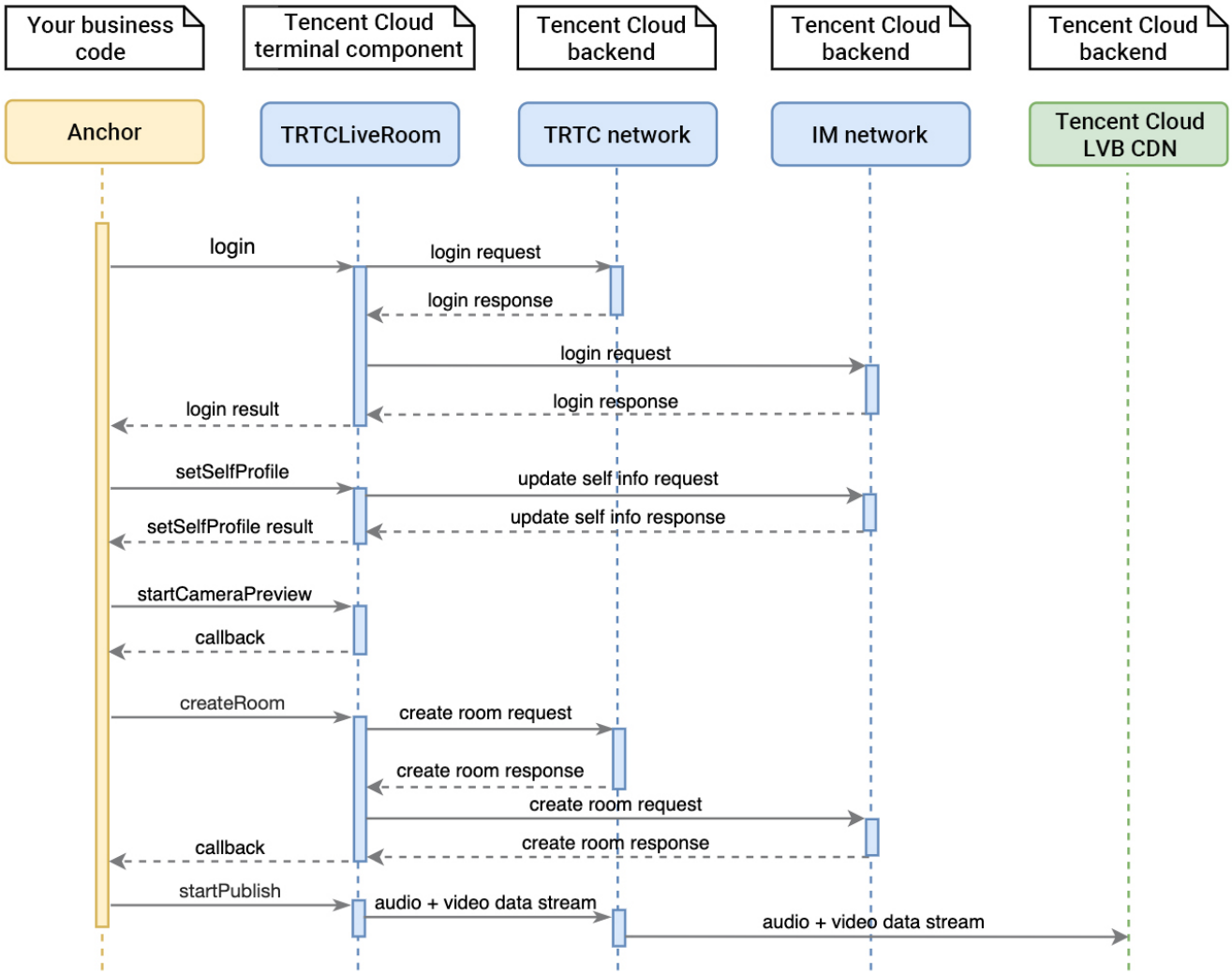
手順5：キャスター側での配信開始

1. キャスターは、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. キャスターは、配信開始前に先に `startCameraPreview` を呼び出してカメラのプレビューを起動させ、インターフェース上に美顔調節ボタンを設置して呼び出し、 `getBeautyManager` で美顔設定を行うことができます。

> 説明：
> エンタープライズ版以外のSDKは美顔やスタンプなどの高度な美顔機能をサポートしていません。

3. 美顔効果の変更後、キャスターは `createRoom` を呼び出して新しいライブストリーミングルームを作成することができます。
4. キャスターが `startPublish` を呼び出し、プッシュを開始します。CDN視聴のサポートが必要な場合は、login時に渡される `TRTCLiveRoomConfig` パラメータ内で `useCDNFirst` と `CDNPlayDomain` を指定し、 `startPublish` 時にライブストリーミング・プルストリーム用のstreamIDを指定してください。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.キャスターは、ニックネームおよびプロフィール画像を設定します
mLiveRoom.setSelfProfile("A", "your_face_url", null);

// 2.キャスターが配信開始前にプレビューを立ち上げ、美顔パラメータを設定します
TXCloudVideoView view = new TXCloudVideoView(context);
parentView.add(view);
mLiveRoom.startCameraPreview(true, view, null);
mLiveRoom.getBeautyManager().setBeautyStyle(1);
mLiveRoom.getBeautyManager().setBeautyLevel(6);

// 3.キャスターがルームを作成します
TRTCLiveRoomDef.TRTCCreateRoomParam param = new TRTCLiveRoomDef.TRTCCreateRoomParam();
param.roomName = 「テストルーム」;
mLiveRoom.createRoom(123456789, param, new TRTCLiveRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
if (code == 0) {
// 4.キャスターがプッシュを開始し、ストリームがCDNに配信されます。
mLiveRoom.startPublish(mSelfUserId + "_stream", null);
}
}
});
```

手順6：視聴者側での視聴

1. 視聴者側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 視聴者側が業務バックエンドから最新のライブストリーミングルームリストを取得します。

> 説明：
> App内のライブストリーミングルームリストはデモに使用するためだけのものです。ライブストリーミングルームリストの業務ロジックは千差万別です。現在、Tencent Cloudはライブストリーミングルームリスト管理のサービスを提供していません。各自でご自分のライブストリーミングルームリストを管理してください。

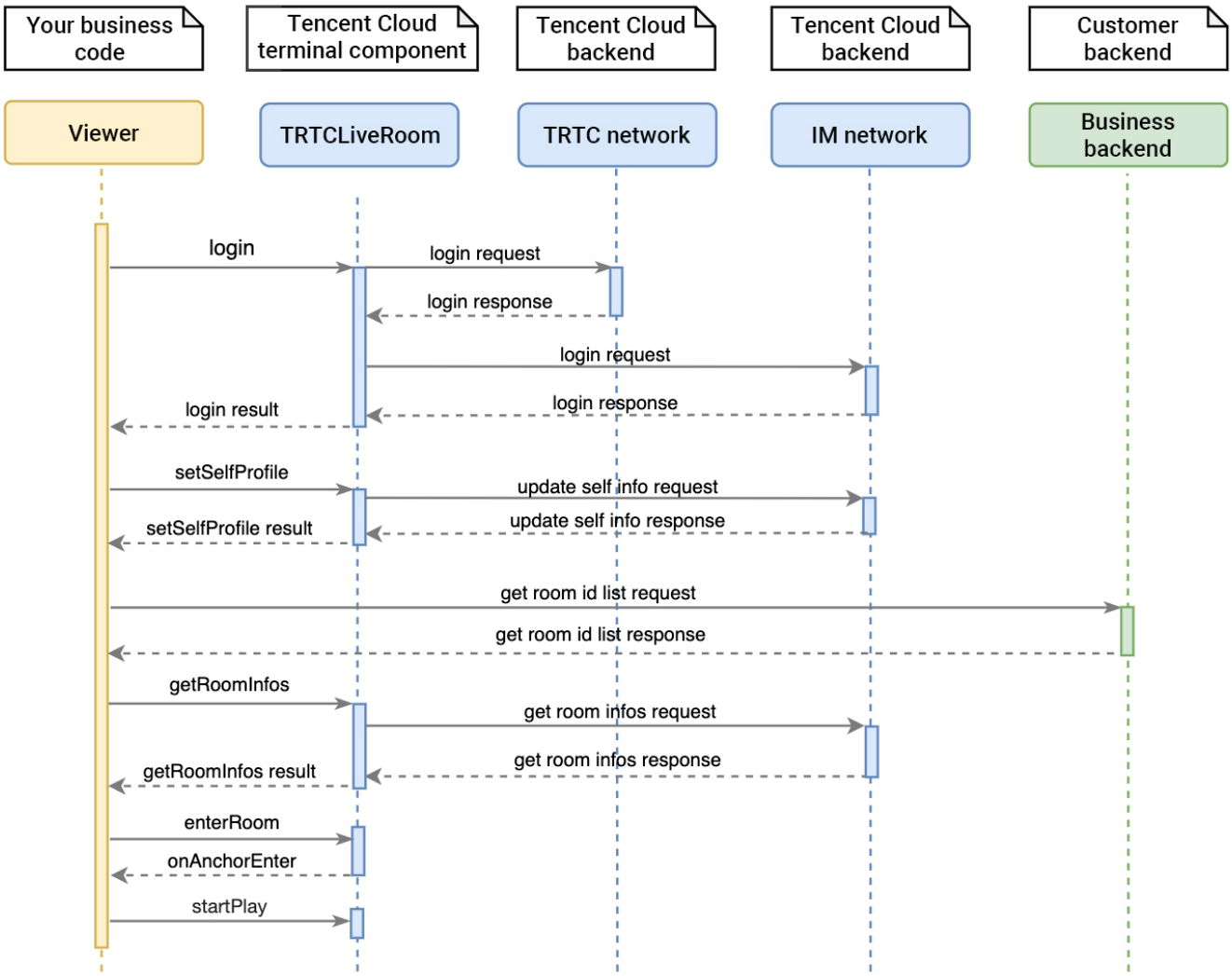
3. 視聴者側は、 `getRoomInfos` を呼び出してルームの詳細情報を取得します。この情報は、キャスター側が `createRoom` を呼び出してライブストリーミングルームを作成するときに設定する簡単な説明情報です。

> 注意：
> ライブストリーミングルームのリストに十分に包括的な情報がある場合は、 `getRoomInfos` の呼び出しに関する手順をスキップできます。

4. 視聴者は1つのライブストリーミングルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
5. `startPlay` を呼び出してキャスターのuserIdを渡し、再生を開始します。

- ライブストリーミングルームリストにキャスターのuserID情報がすでに含まれている場合、視聴者は直接 `startPlay` を呼び出してキャスターのuserIdを渡せば、再生を開始できます。
- ルーム参加前にキャスターのuserIDが一時的に取得できない場合、視聴者側はルーム参加後にキャスターの `onAnchorEnter` のイベントコールバックを受信します。このコールバックの中にキャスターのuserID情報が含まれていますので、 `startPlay` を呼び出せば再生できます。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.業務バックエンドから取得したルームリストをroomListと仮定します
List<integer> roomList = GetRoomList();

// 2. getRoomInfosを呼び出すことによって、ルームの詳細情報を取得します
mLiveRoom.getRoomInfos(roomList, new TRTCLiveRoomCallback.RoomInfoCallback() {
@Override
public void onCallback(int code, String msg, List<trtcliveroomdef.trtcliveroominfo> list) {
if (code == 0) {
// ルーム詳細情報の取得後、キャスターリストページでキャスターのニックネーム、プロフィール画像等の関連情報を表示できます
}
}
})

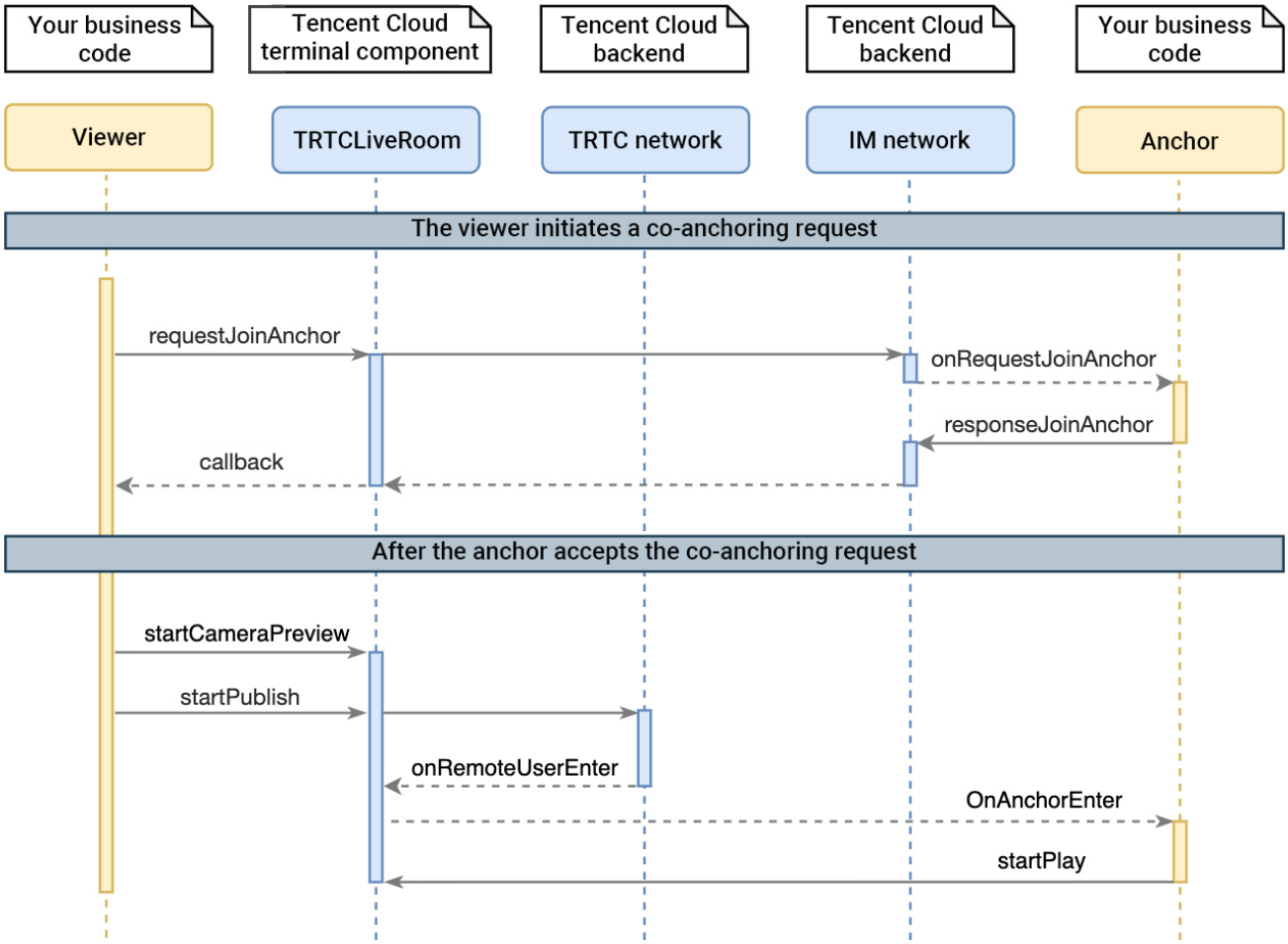
// 3.roomidを選択しルームに参加します
mLiveRoom.enterRoom(roomid, null);
```

```
// 4.視聴者がキャスターのルーム参加の通知を受信すると、再生が開始されます
mLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
@Override
public void onAnchorEnter(final String userId) {
// 5.視聴者がキャスターの画面を再生します
mLiveRoom.startPlay(userId, mTXCloudVideoView, null);
}
});
```

手順7：視聴者とキャスターとのマイク接続

1. 視聴者側が requestJoinAnchor を呼び出し、キャスター側にマイク接続のリクエストを送信します。
2. キャスター側は TRTCLiveRoomDelegate#onRequestJoinAnchor（視聴者からのマイク接続のリクエストがあります）のイベント通知を受信します。
3. キャスター側は responseJoinAnchor を呼び出し、視聴者からのマイク接続リクエストを受け入れるかどうか決定できます。
4. 視聴者側が `TRTCLiveRoomDelegate#responseCallback` のイベント通知を受信します。この通知にはキャスターの処理結果が含まれています。
5. キャスターがマイク接続リクエストに同意したら、視聴者側は `startCameraPreview` を呼び出し、ローカルカメラを起動させます。その後 startPublish を呼び出し、視聴者側のプッシュを開始します。
6. キャスター側は、視聴者側の起動の通知の後に、 `TRTCLiveRoomDelegate#onAnchorEnter` （別のオーディオ・ビデオストリーミングが届いています）の通知を受信します。この通知には視聴者側のuserIdが含まれています。
7. キャスター側が startPlay を呼び出すと、マイク接続の視聴者の画面を見ることができます。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.視聴者側がマイク接続のリクエストを送信します
mLiveRoom.requestJoinAnchor(mSelfUserId + "あなたとのマイク接続をリクエスト",
new TRTCLiveRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
// 4.キャスターが視聴者のリクエストを受け入れます
TXCloudVideoView view = new TXCloudVideoView(context);
parentView.add(view);
// 5.視聴者がプレビューを起動し、プッシュを開始します
mLiveRoom.startCameraPreview(true, view, null);
mLiveRoom.startPublish(mSelfUserId + "_stream", null);
}
}
});

// 2.キャスター側がマイク接続のリクエストを受信します
mLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
```

©2013-2019 Tencent Cloud. All rights reserved.

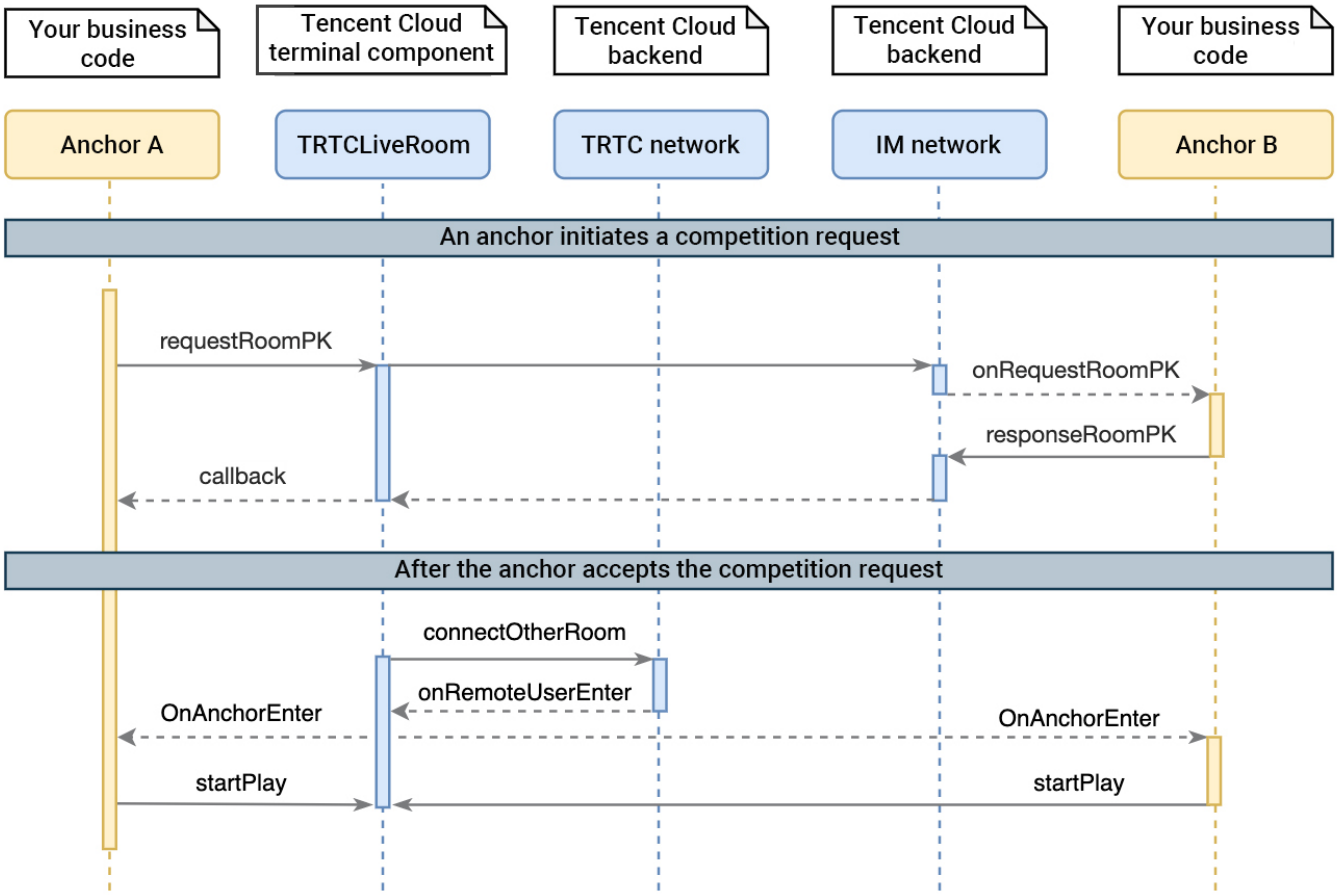
```
@Override
public void onRequestJoinAnchor(final TRTCLiveRoomDef.TRTCLiveUserInfo userInfo,
String reason, final int timeout) {
// 3.相手側のマイク接続のリクエストに同意します
mLiveRoom.responseJoinAnchor(userInfo.userId, true, "マイク接続に同意");
}

@Override
public void onAnchorEnter(final String userId) {
// 6.キャスターがマイク接続の視聴者のマイク・オンの通知を受信します
TXCloudVideoView view = new TXCloudVideoView(context);
parentView.add(view);
// 7.キャスターが視聴者の画面を再生します
mLiveRoom.startPlay(userId, view, null);
}
});
```

手順8：キャスターとキャスターPK

1. キャスターAが requestRoomPK を呼び出し、キャスターBにPKのリクエストを送信します。
2. キャスターBが `TRTCLiveRoomDelegate onRequestRoomPK` のコールバック通知を受信します。
3. キャスターBが `responseRoomPK` を呼び出し、キャスターA のPKのリクエストを受け入れるか決定します。
4. キャスターBは、キャスターAのリクエストを受け入れたら、 `TRTCLiveRoomDelegate onAnchorEnter` の通知を待ってから、 `startPlay` を呼び出し、キャスターAを表示します。
5. PKのリクエストが同意されたかについて、キャスターAが `responseCallback` のコールバック通知を受信します。
6. キャスターAのリクエストが同意されたら、 `TRTCLiveRoomDelegate onAnchorEnter` の通知を待ってから、 `startPlay` を呼び出し、キャスターBを表示します 。

©2013-2019 Tencent Cloud. All rights reserved.




```
// キャスターA:
// キャスターAがルーム12345を作成します
mLiveRoom.createRoom(12345, param, null);

mLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
@Override
public void onAnchorEnter(final String userId) {
// 6.Bのルーム参加の通知を受信します
mLiveRoom.startPlay(userId, mTXCloudVideoView, null);
}
});

// 1.キャスターAがキャスターBにPKのリクエストを送信します
mLiveRoom.requestRoomPK(54321, "B",
new TRTCLiveRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
// 5.同意の有無のコールバックを受信します
if (code == 0) {
// ユーザー受け入れ
```

©2013-2019 Tencent Cloud. All rights reserved.

```
}else{
// ユーザー拒否
}
}
});

// キャスターB：
// キャスターBがルーム54321を作成します
mLiveRoom.createRoom(54321, param, null);

// 2.キャスターBがキャスターAのメッセージを受信します
mLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
@Override
public void onRequestRoomPK(
final TRTCLiveRoomDef.TRTCLiveUserInfo userInfo, final int timeout) {
// 3.キャスターBがキャスターAに回答し、リクエストを受け入れます
mLiveRoom.responseRoomPK(userInfo.userId, true, "");
}
@Override
public void onAnchorEnter(final String userId) {
// 4.キャスターBがキャスターAのルーム参加の通知を受信し、キャスターAの画面を再生します
mLiveRoom.startPlay(userId, mTXCloudVideoView, null);
}
});
```

手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、該当するルーム内の全てのキャスターおよび視聴者が `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

  ```
  // 送信側：テキストメッセージの送信
  mLiveRoom.sendRoomTextMsg("Hello Word!", null);
  // 受信側：テキストメッセージのモニタリング
  mLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
  @Override
  public void onRecvRoomTextMsg(String roomId,
  String message, TRTCLiveRoomDef.TRTCLiveUserInfo userInfo) {
  Log.d(TAG、 + userInfo.userName + 「からのメッセージを受信:」 + message);
  }
  });
  ```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターと視聴者が `onRecvRoomCustomMsg` コールバックを受信できます。

©2013-2019 Tencent Cloud. All rights reserved.

カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

```
// eg:「CMD_DANMU」は弾幕コメントを表し、「CMD_LIKE」は「いいね」情報を表します
mLiveRoom.sendRoomCustomMsg("CMD_DANMU", "Hello world", null);
mLiveRoom.sendRoomCustomMsg("CMD_LIKE", "", null);
// 受信側：カスタムメッセージのモニタリング
mLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
@Override
public void onRecvRoomCustomMsg(String roomId, String cmd,
String message, TRTCLiveRoomDef.TRTCLiveUserInfo userInfo) {
if ("CMD_DANMU".equals(cmd)) {
// 弾幕コメントの受信
Log.d(TAG、 + userInfo.userName + 「からの弾幕コメントを受信:」 + message);
} else if ("CMD_LIKE".equals(cmd)) {
// 「いいね」情報の受信
Log.d(TAG、 userInfo.userName + 「いいねを付けました！」);
}
}
});
```

©2013-2019 Tencent Cloud. All rights reserved.

ビデオ・インタラクティブストリーミング (Flutter)

最終更新日：：2022-03-31 11:39:17

ビデオ・インタラクティブストリーミングの機能をすばやく実装する必要がある場合、当社が提供するDemoをもとに直接修正を加えてフィットさせることも、当社が提供するTRTCLiveRoomコンポーネントを使用し、カスタマイズしたUIを実現することも可能です。

DemoのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、【開発支援】>【快速跑通Demo】を選択します。
2. `TestLive` などのアプリケーション名を入力して、【作成】をクリックします。

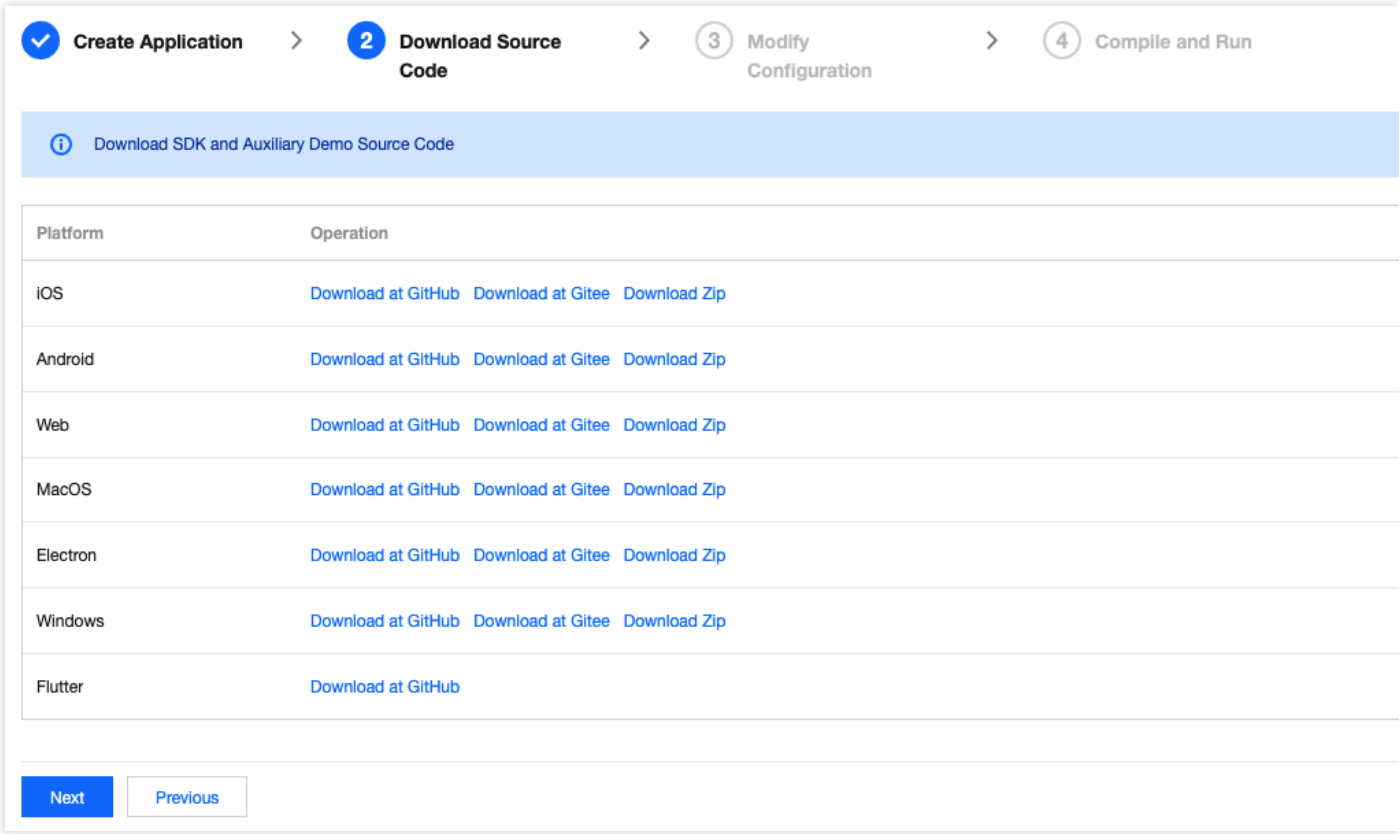
> 注意：
> 本機能はTencent CloudTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの価格説明をご参照ください。

ステップ2：SDKおよびDemoソースコードをダウンロード

1. 実際のビジネスニーズに基づき、SDKおよび付属のDemoソースコードをダウンロードします。

©2013-2019 Tencent Cloud. All rights reserved.

2. ダウンロード完了後、【ダウンロード済み。次へ】をクリックします。



ステップ3：Demo プログラムファイルの設定

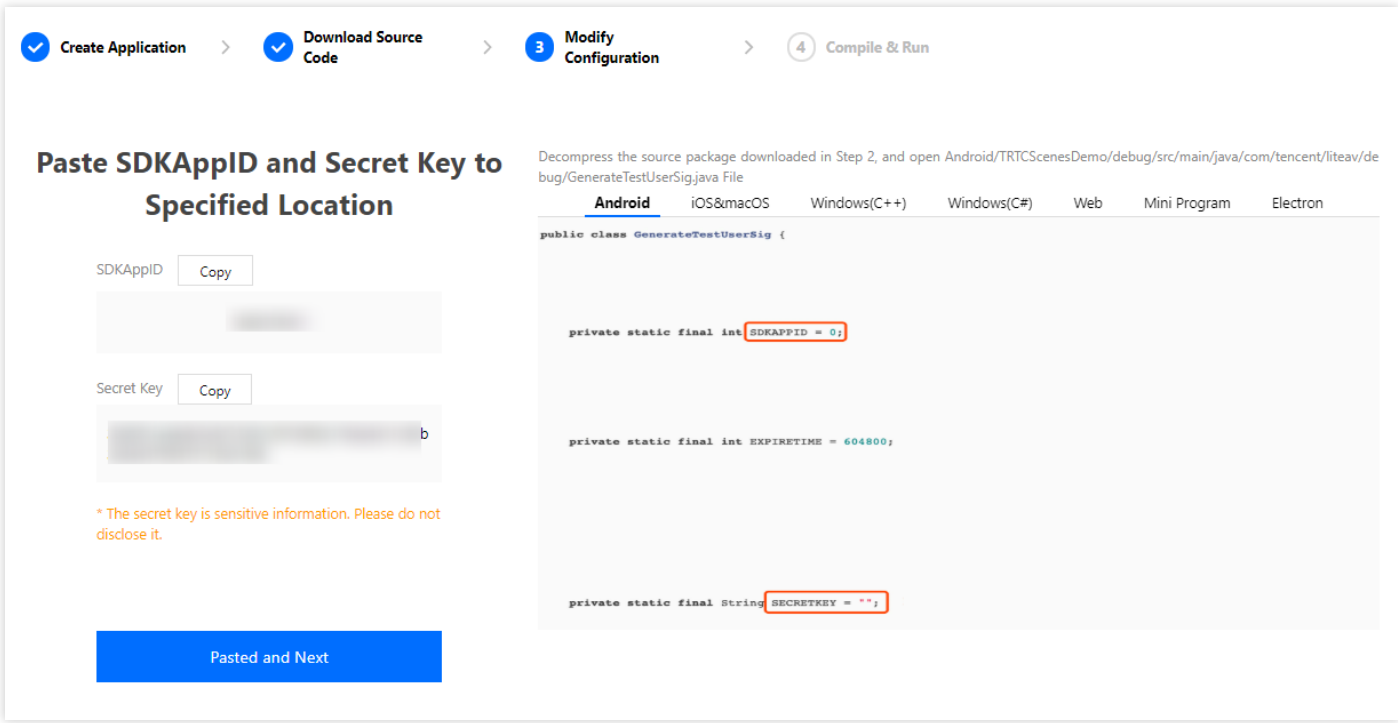
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `/example/lib/debug/GenerateTestUserSig.dart` ファイルを見つけて開きます。

3. `GenerateTestUserSig.dart` ファイル内の関連パラメータを設定します。

   - SDKAPPID：デフォルトはPLACEHOLDER。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトはPLACEHOLDER。実際のキー情報を設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、【貼り付けました。次へ】をクリックすれば、作成が完了します。

5. コンパイル完了後、【コンソールの概要ページに戻る】をクリックすればOKです。

注意：

- ここで言及したUserSigの新規作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：コンパイル実行

注意：
Android端末は実際のマシンで実行する必要があり、エミュレーターのデバッグはサポートされていません。

©2013-2019 Tencent Cloud. All rights reserved.

1. `flutter pub get` を実行します。
2. コンパイルを実行し、デバッグを行います。
   - Android 端末
   - iOS 端末
   i. `flutter run` を実行します。
   ii. Android Studio（3.5およびそれ以降のバージョン）を使用して、ソースプロジェクトを開き、【実行】をクリックすれば完了です。

手順5：Demoソースコードの修正

ソースコードのTRTCLiveRoomフォルダは、uiとmodelの2つのサブフォルダを含んでいます。uiフォルダにはインターフェースコードが含まれています。以下の表にはファイルまたはフォルダおよび対応するUIをリストアップして、二次調整を行いやすくしています。

ファイルまたはフォルダ	機能の説明
base	UIに使用される基礎となるクラス。
list	リストページおよびルームページの作成。
room	メインルームページは、キャスターと視聴者という2種類のインターフェースがあります。

UIカスタマイズの実装

ソースコードフォルダ `TRTCLiveRoom` にはuiとmodelの2つのサブフォルダが含まれ、そのうちmodelフォルダには再利用可能なオープンソースコンポーネントTRTCLiveRoomが含まれています。このコンポーネントが提供するインターフェース関数は `TRTCLiveRoom.dart` ファイルで確認でき、対応するインターフェースを使用してカスタマイズUIを実装することができます。

手順1：SDKの統合

インタラクティブライブストリーミングコンポーネントTRTCLiveRoomは、TRTC SDKとIM SDKに依存しています。 `pubspec.yaml` を設定することで、自動的にダウンロードして更新できます。

プロジェクトの `pubspec.yaml` に次の依存関係を記述します。

```
dependencies:
tencent_trtc_cloud: 最新バージョン番号
tencent_im_sdk_plugin: 最新バージョン番号
```

手順2：権限の設定および難読化ルール

©2013-2019 Tencent Cloud. All rights reserved.

- iOS 端末
- Android 端末

`Info.plist` にカメラとマイクの権限申請を追加する必要があります。

```
<key>NSMicrophoneUsageDescription</key>
<string>通常の音声通話が行えるようにマイクの権限を承認します</string>
```

手順3：TRTCLiveRoomコンポーネントをインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
lib/TRTCLiveRoom/model/
```

手順4：コンポーネントの作成およびログイン

1. `sharedInstance` インターフェースを呼び出すと、TRTCLiveRoomコンポーネントのインスタンスオブジェクトを作成できます。
2. `registerListener` 関数を呼び出して、コンポーネントのイベント通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	作用
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
useCDNFirst	視聴者の視聴方式の設定に使用します。trueは一般視聴者のCDN経由での視聴を表し、料金は安価ですがレイテンシーは高めです。falseは一般視聴者の低レイテンシーによる視聴を表し、料金はCDN とマイク接続の間ですが、遅延を1s以内に抑えることができます。
CDNPlayDomain	useCDNFirstの設定をtrueにした時に有効になり、CDN経由の視聴の再生ドメイン名の指定に利用します。CSSコンソール >【ドメイン名管理】ページからログインし、設定することができます。

手順5：キャスター側での配信開始

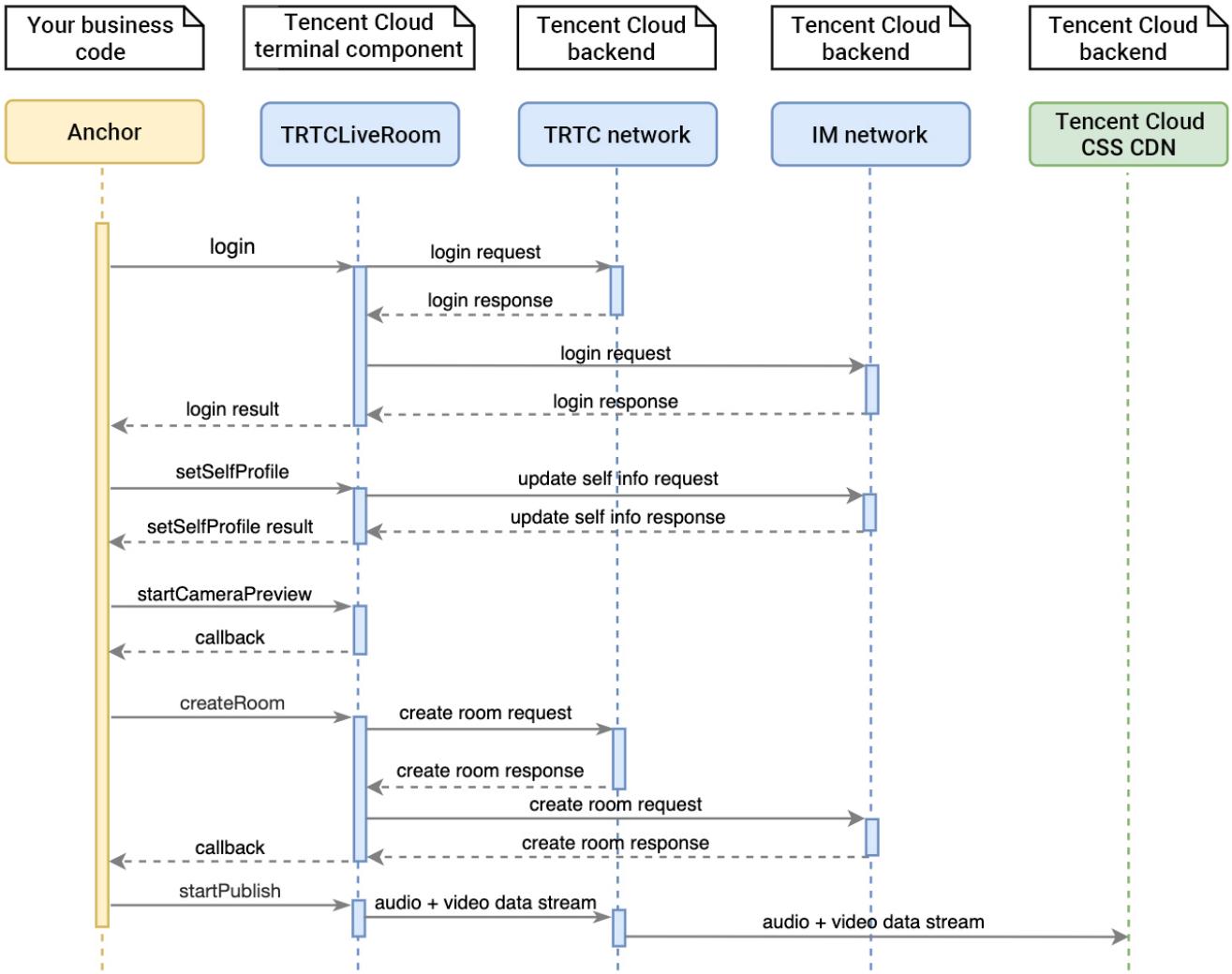
©2013-2019 Tencent Cloud. All rights reserved.

1. キャスターは、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. キャスターは、配信開始前に先に `startCameraPreview` を呼び出してカメラのプレビューを起動させ、インターフェース上に美顔調節ボタンを設置して呼び出し、 `getBeautyManager` で美顔設定を行うことができます。

> 説明：
> エンタープライズ版以外のSDKは美顔やスタンプの機能をサポートしていません。

3. 美顔効果の変更後、キャスターは `createRoom` を呼び出して新しいライブストリーミングルームを作成することができます。
4. キャスターが `startPublish` を呼び出し、プッシュを開始します。CDN視聴のサポートが必要な場合は、login時に渡される `TRTCLiveRoomConfig` パラメータ内で `useCDNFirst` と `CDNPlayDomain` を指定し、 `startPublish` 時にライブストリーミング・プルストリーム用のstreamIDを指定してください。

©2013-2019 Tencent Cloud. All rights reserved.



手順6：視聴者側での視聴

1. 視聴者側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 視聴者側が業務バックエンドから最新のライブストリーミングルームリストを取得します。

> 説明：
> Demo内のライブストリーミングルームリストはデモに使用するためだけのものです。ライブストリーミングルームリストの業務ロジックは千差万別です。現在、Tencent Cloudはライブストリーミングルームリスト管理のサービスを提供していません。各自でご自分のライブストリーミングルームリストを管理してください。

©2013-2019 Tencent Cloud. All rights reserved.

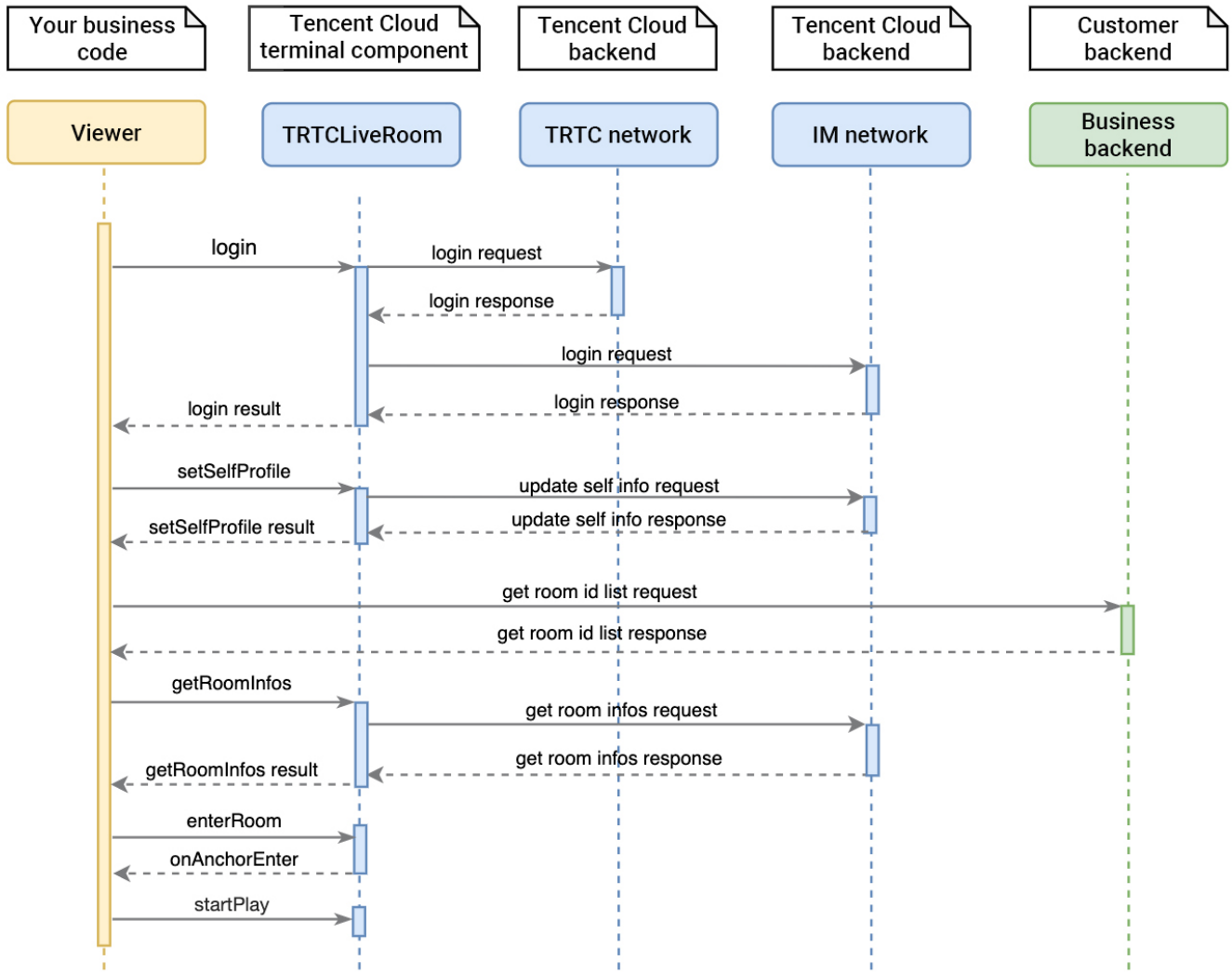
3. 視聴者側は、 `getRoomInfos` を呼び出してルームの詳細情報を取得します。この情報は、キャスター側が `createRoom` を呼び出してライブストリーミングルームを作成するときに設定する簡単な説明情報です。

> 注意：
> ライブストリーミングルームのリストに十分に包括的な情報がある場合は、 `getRoomInfos` の呼び出しに関する手順をスキップできます。

4. 視聴者は1つのライブストリーミングルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
5. `startPlay` を呼び出してキャスターのuserIdを渡し、再生を開始します。

- ライブストリーミングルームリストにキャスターのuserID情報がすでに含まれている場合、視聴者は直接 `startPlay` を呼び出してキャスターのuserIdを渡せば、再生を開始できます。
- ルーム参加前にキャスターのuserIDが一時的に取得できない場合、視聴者側はルーム参加後にキャスターの `onAnchorEnter` のイベントコールバックを受信します。このコールバックの中にキャスターのuserID情報が含まれていますので、 `startPlay` を呼び出せば再生できます。

©2013-2019 Tencent Cloud. All rights reserved.

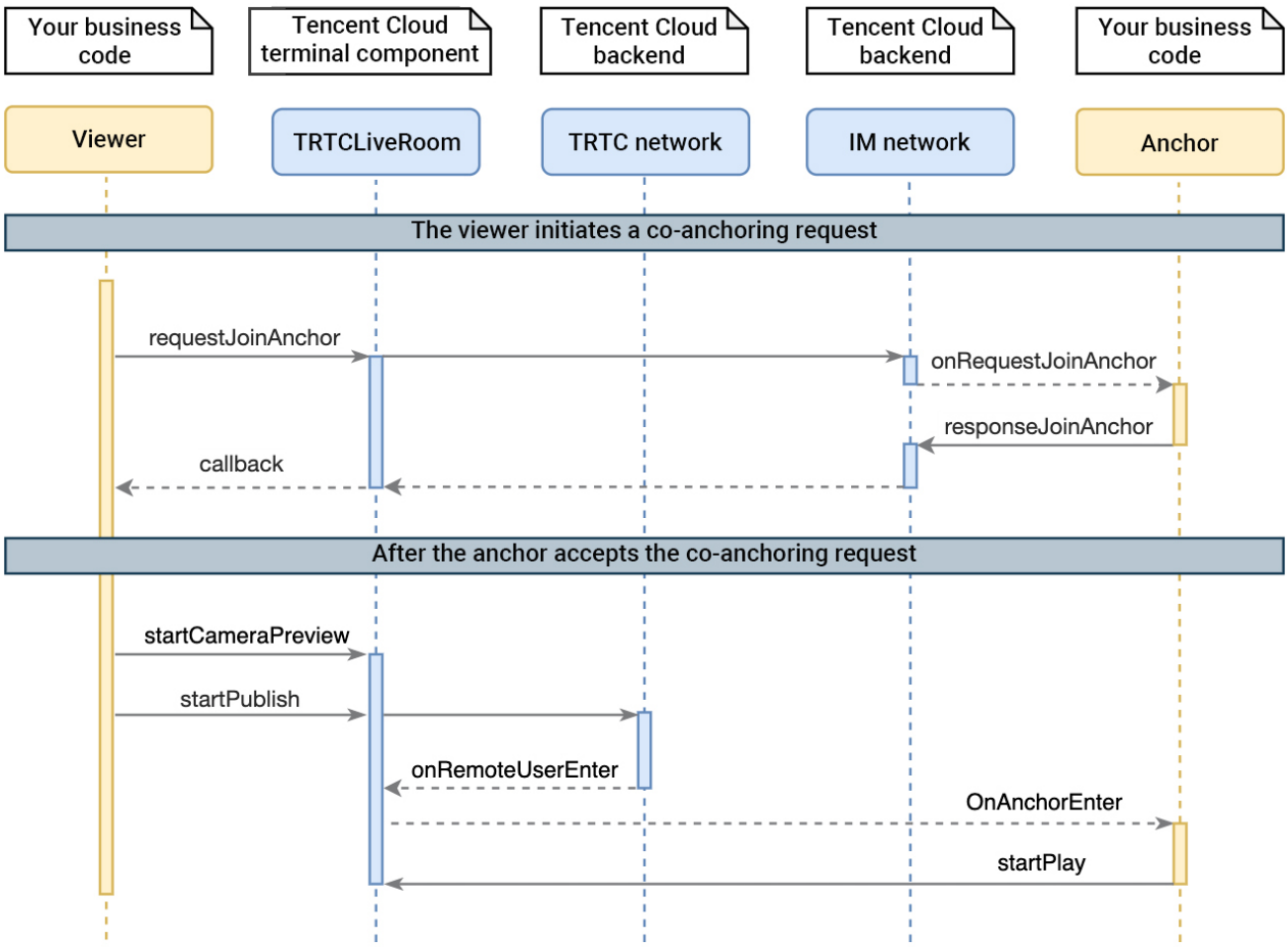


手順7：視聴者とキャスターとのマイク接続のオンとオフ

1. 視聴者側が `requestJoinAnchor` を呼び出し、キャスター側にマイク接続のリクエストを送信します。
2. キャスター側は `TRTCLiveRoomDelegate#onRequestJoinAnchor`（視聴者からのマイク接続のリクエストがあります）のイベント通知を受信します。
3. キャスター側は `responseJoinAnchor` を呼び出し、視聴者からのマイク接続リクエストを受け入れるかどうか決定できます。
4. 視聴者側は `onAnchorAccepted` イベント通知を受信します。この通知にはキャスター側の処理結果が含まれています。
5. キャスターがマイク接続リクエストに同意したら、視聴者側は `startCameraPreview` を呼び出し、ローカルカメラを起動させます。その後 `startPublish` を呼び出し、視聴者側のプッシュを開始します。
6. キャスター側は、視聴者側の起動の通知の後に、`TRTCLiveRoomDelegate#onAnchorEnter`（別のオーディオ・ビデオストリーミングが届いています）の通知を受信します。この通知には視聴者側のuserIdが含まれています。

©2013-2019 Tencent Cloud. All rights reserved.

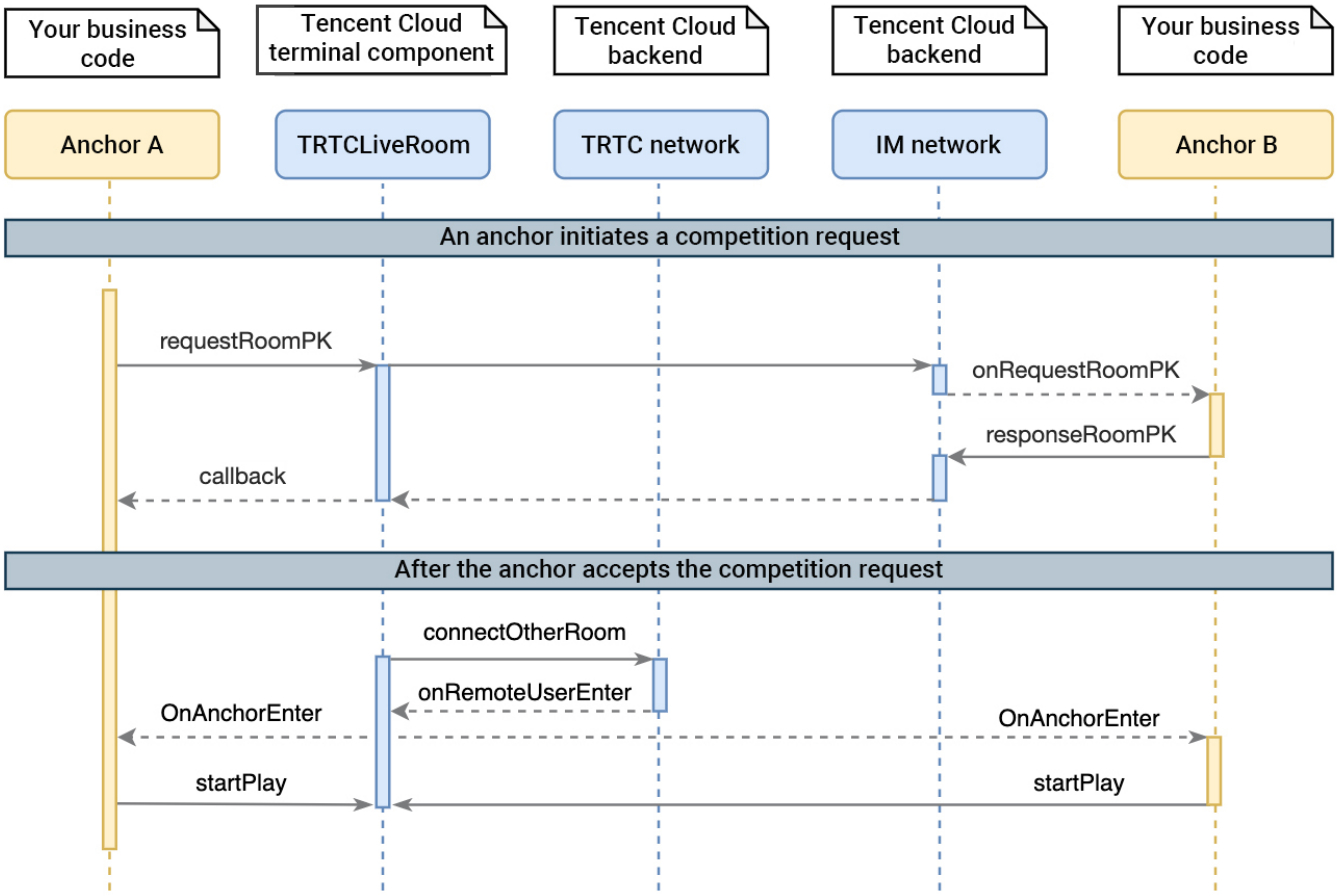
7. キャスター側が `startPlay` を呼び出すと、マイク接続の視聴者の画面を見ることができます。



手順8：キャスターとキャスターPK

1. キャスターAが `requestRoomPK` を呼び出し、キャスターBにPKのリクエストを送信します。
2. キャスターBが `TRTCLiveRoomDelegate onRequestRoomPK` のコールバック通知を受信します。
3. キャスターBが `responseRoomPK` を呼び出し、キャスターA のPKのリクエストを受け入れるか決定します。
4. キャスターBは、キャスターAのリクエストを受け入れたら、 `TRTCLiveRoomDelegate onAnchorEnter` の通知を待ってから、 `startPlay` を呼び出し、キャスターAを表示します。
5. PKのリクエストが同意されたかについて、キャスターAが `onRoomPKAccepted` のコールバック通知を受信します。
6. キャスターAのリクエストが同意されたら、 `TRTCLiveRoomDelegate onAnchorEnter` の通知を待ってから、 `startPlay` を呼び出し、キャスターBを表示します 。

©2013-2019 Tencent Cloud. All rights reserved.



手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、該当するルーム内の全てのキャスターおよび視聴者が `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

```
// 発信側：テキストメッセージの発信
trtcLiveCloud.sendRoomTextMsg("Hello Word!");


// 受信側：テキストメッセージのモニタリング
onListenerFunc(type, params) async {
switch (type) {
case TRTCLiveRoomDelegate.onRecvRoomTextMsg:
//グループテキストメッセージを受信すると、テキストチャットルームを使用できます
break;
}
}
```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターと視聴者が `onRecvRoomCustomMsg` コールバックを受信できます。
  カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCLiveRoom API (iOS)

最終更新日：：2021-10-14 17:51:03

TRTCLiveRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- キャスターが新しいライブストリーミングルームを作成して配信を開始し、視聴者がライブストリーミングルームに参加して視聴します。
- キャスターと視聴者がビデオでマイク接続によるコラボライブを行います。
- 2つの異なるルームの間でキャスター同士のPK（クロスルーム対戦）を行います。
- 各種のテキストメッセージやカスタムメッセージの送信をサポートします。カスタムメッセージは弾幕、「いいね」、ギフトを実装するために使用することができます。

TRTCLiveRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実現プロセスは、 ビデオマイク接続ライブストリーミング（iOS）をご参照ください。

- TRTC SDK： TRTC SDK を低遅延のライブストリーミングコンポーネントとして使用します。
- IM SDK： IM SDKのAVChatroomを使用して、ライブストリーミングチャットルーム機能を実装します。同時に、IMのメッセージでキャスター間のマイク接続のフローをつなげます。

TRTCLiveRoom API概要

SDK基本関数

API	説明
delegate	イベントコールバックを設定します。
login	ログイン。
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの新規作成（キャスターが呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動作成します。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
destroyRoom	ルームの廃棄（キャスターが呼び出し）。
enterRoom	入室（視聴者が呼び出し）。
exitRoom	退室（視聴者が呼び出し）。
getRoomInfos	ルームリストの詳細情報を取得します。
getAnchorList	ルーム内の全キャスターのリストを取得します。enterRoom() 成功後に呼び出しが有効となります。
getAudienceList	ルーム内の全視聴者の情報を取得します。enterRoom() 成功後に呼び出しが有効となります。

プッシュプルストリームに関連するインターフェース関数

API	説明
startCameraPreview	ローカルビデオのプレビュー画面を立ち上げます。
stopCameraPreview	ローカルのビデオキャプチャおよびプレビューを停止します。
startPublish	ライブストリーミング（プッシュ）を開始します。
stopPublish	ライブストリーミング（プッシュ）を停止します。
startPlay	リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。
stopPlay	リモートのビデオ画面のレンダリングを停止します。

キャスターと視聴者のマイク接続

API	説明
requestJoinAnchor	視聴者がマイク接続をリクエストします。
responseJoinAnchor	キャスターがマイク接続のリクエストを処理します。
kickoutJoinAnchor	キャスターがマイク接続視聴者をキックアウトします。

キャスターのルーム間PK

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
requestRoomPK	キャスターがルーム間PKをリクエストします。
responseRoomPK	キャスターがルーム間PKのリクエストに応答します。
quitRoomPK	ルーム間PKから退出します。

オーディオビデオ制御に関連するインターフェース関数

API	説明
switchCamera	フロント/リアカメラを切り替えます。
setMirror	ミラー表示のオン/オフを設定します。
muteLocalAudio	ローカルオーディオをミュートにします。
muteRemoteAudio	リモートオーディオをミュートにします。
muteAllRemoteAudio	全てのリモートオーディオをミュートにします。

BGMサウンドエフェクト関連インターフェース関数

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerを取得します。

美顔フィルタに関するインターフェース関数

API	説明
getBeautyManager	美顔管理オブジェクト TXBeautyManagerを取得します。

メッセージ送信関連インターフェース関数

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

©2013-2019 Tencent Cloud. All rights reserved.

デバックに関するインターフェース関数

API	説明
showVideoDebugLog	debug情報をインターフェースに表示するかどうか。

TRTCLiveRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが破棄された時のコールバック。
onRoomInfoChange	ライブストリーミングルーム情報変更のコールバック。

キャスターと視聴者の参加/退出のイベントコールバック

API	説明
onAnchorEnter	新しいキャスターの入室通知を受信します。
onAnchorExit	キャスターの退室通知を受信します。
onAudienceEnter	視聴者入室通知の受信。
onAudienceExit	視聴者退室通知の受信。

キャスターと視聴者のマイク接続のイベントコールバック

API	説明
onRequestJoinAnchor	キャスターが視聴者のマイク接続リクエストを受信した時のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onKickoutJoinAnchor	マイク接続の視聴者がマイク接続からキックアウトされた通知を受信します。

キャスター間PKのイベントコールバック

API	説明
onRequestRoomPK	ルーム間PKのリクエストの通知を受信。
onQuitRoomPK	ルーム間PK中止の通知を受信します。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

SDK基本関数

delegate

TRTCLiveRoom イベントコールバック。 TRTCLiveRoomDelegate を介してTRTCLiveRoomの各種ステータス通知を受け取ることができます。

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

> 説明：
> delegateは、TRTCLiveRoomのプロキシコールバックです。

login

ログイン。

```
/// コンポーネントシステムにログイン
/// - Parameters:
/// - sdkAppID: TRTCコンソール > 【[アプリケーション管理](https://console.cloud.tencent.com/trtc/app)】 > アプリケーション情報でSDKAppIDを確認できます。
```

©2013-2019 Tencent Cloud. All rights reserved.

```
/// - userID: 現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（¥_）のみ使用できます。
/// - userSig: Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については[UserSigの計算方法](https://intl.cloud.tencent.com/document/product/647/35166)をご参照ください。
/// - config: グローバルコンフィギュレーション情報。ログイン時に初期化してください。ログイン後は変更できなくなります。isAttachedTUIKit プロジェクトにTUIKitを導入し使用しているか。
/// - callback: ログインのコールバック。成功時にcodeは0になります。
/// - Note:
/// - userSigの設定は 7日を推奨します。usersign期限切れによるIMのメッセージ送受信の失敗、TRTCのマイク接続の失敗等を効果的に回避できます。
- (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:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppID	Int	TRTCコンソール > 【アプリケーション管理】> アプリケーション情報でSDKAppIDを確認できます。
userID	String	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法についてはUserSigの計算方法をご参照ください。
config	TRTCLiveRoomConfig	グローバルコンフィギュレーション情報。ログイン時に初期化してください。ログイン後は変更できなくなります。 • useCDNFirst 属性：視聴者の視聴方式の設定に使用します。trueは、普通の視聴者のCDN経由での視聴を表し、費用は安価ですがディレーは高めです。falseは、普通の視聴者の低遅延による視聴を表し、費用は CDN とマイク接続の間ですが、ディレーを1s以内に抑えることができます。 • CDNPlayDomain 属性： useCDNFirstの設定がtrueの時に有効となり、CDNでの視聴の再生ドメイン名の指定に使用します。CSSコンソール > 【ドメイン名管理】の画面からログインし、設定を行います。
callback	(_ code: Int, _ message: String?) -> Void	ログインのコールバック。成功時にcodeは0になります。

©2013-2019 Tencent Cloud. All rights reserved.

logout

ログアウト。

```
// ログイン状態から退出
/// - Parameter callback: ログアウトのコールバック。成功時にcodeは0になります。
- (void)logout:(Callback _Nullable)callback
NS_SWIFT_NAME(logout(_:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String?) -> Void	ログアウトのコールバック。成功時にcodeは0になります。

setSelfProfile

個人情報の修正。

```
/// ユーザー情報を設定します。設定したユーザー情報は、Tencent CloudのIMクラウドサーバーに保存されます。
/// - Parameters:
/// - name: ユーザーのニックネーム
/// - avatarURL: ユーザーのプロフィール画像のアドレス
/// - callback: 個人情報設定のコールバック。成功時にcodeは0になります。
- (void)setSelfProfileWithName:(NSString *)name
avatarURL:(NSString * _Nullable)avatarURL
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(setSelfProfile(name:avatarURL:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
name	String	ニックネーム。
avatarURL	String	プロフィール画像のアドレス。
callback	(_ code: Int, _ message: String?) -> Void	個人情報設定のコールバック。成功時にcodeは0になります。

ルーム関連インターフェース関数

createRoom

©2013-2019 Tencent Cloud. All rights reserved.

ルームの新規作成（キャスターが呼び出し）。

```
/// ルームを作成（キャスターが呼び出し）します。ルームが存在しない場合、システムが新しいルームを1つ自動作成します。
/// キャスターが配信を開始する際の通常の呼び出しプロセスは以下のとおりです。
/// 1.【キャスター】startCameraPreview()を呼び出し、カメラのプレビューを起動します。この時美顔パラメータを調整できます。
/// 2.【キャスター】 createRoom()を呼び出し、ライブストリーミングルームを作成します。ルーム作成の成功の有無がcallbackでキャスターに通知されます。
/// 3.【キャスター】startPublish() を呼び出し、プッシュを開始します。
/// - Parameters:
/// - roomID: ルームIDは、ご自身でアサインし、一元管理する必要があります。複数の roomID を、1つのライブストリーミングルームリストにまとめることができます。Tencent Cloudではライブストリーミングルームリストの管理サービスを一時的に行いませんので、ご自身でライブルームリストを管理してください。
/// - roomParam: TRTCCreateRoomParam | ルーム情報は、ルームの説明情報として利用します（例：ルーム名、カバー情報など）。ルームリストとルーム情報をご自分で管理する場合は、このパラメータは無視してかまいません。
/// - callback: 入室結果のコールバック。成功時にcodeは0になります。
/// - Note:
/// - キャスターがライブストリーミングを開始する時のコールでは、自分が作成したことがあるルームを再び作成することができます。
- (void)createRoomWithRoomID:(UInt32)roomID
roomParam:(TRTCCreateRoomParam *)roomParam
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(createRoom(roomID:roomParam:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomID	UInt32	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomIDを、1つのライブストリーミングリストにまとめることができます。Tencent Cloudではライブストリーミングリストの管理サービスを一時的に行いませんので、ご自身でライブストリーミングルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報は、ルームの説明情報として利用します（例：ルーム名、カバー情報など）。ルームリストとルーム情報をご自分で管理する場合は、このパラメータは無視してかまいません。
callback	(_ code: Int, _ message: String?) -> Void	ルームの新規作成結果のコールバック。成功時にcodeは0になります。

©2013-2019 Tencent Cloud. All rights reserved.

キャスターが配信を開始する際の通常の呼び出しプロセスは次のとおりです。

1. 【キャスター】 `startCameraPreview()` を呼び出し、カメラのプレビューを起動します。この時美顔パラメータを調整できます。
2. 【キャスター】 `createRoom()` を呼び出し、ライブストリーミングルームを作成します。ルーム作成の成功の有無がcallbackでキャスターに通知されます。
3. 【キャスター】 `starPublish()` を呼び出し、プッシュを開始します。

destroyRoom

ルームの破棄（キャスターが呼び出し）。キャスターは、ルーム作成後、この関数を呼び出してルームを破棄します。

```
/// ルームの破棄（キャスターが呼び出し）
/// キャスターはルーム作成後、この関数を呼び出してルームを破棄することができます。
/// - Parameter callback: ルーム破棄結果のコールバック。成功時にcodeは0になります。
/// - Note:
/// - キャスターは、ルーム作成後、この関数を呼び出してルームを破棄します。
- (void)destroyRoom:(Callback _Nullable)callback
NS_SWIFT_NAME(destroyRoom(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String?) -> Void	ルーム破棄結果のコールバック、成功時にcodeは0になります。

enterRoom

入室（視聴者が呼び出し）。

```
/// 入室（視聴者が呼び出し）
/// 視聴者のライブストリーミング視聴の通常の呼び出しフローは以下のとおりです。
/// 1.【視聴者】サーバーから取得する最新のライブストリーミングルームリストには、多くのライブストリーミングルームのroomId およびルーム情報が含まれます。
/// 2.【視聴者】視聴者は1つのライブストリーミングルームを選択し、enterRoom()を呼び出して、このルームに参加します。
/// 3.【視聴者】お客様のサーバーが管理するルームリストの中にそれぞれのルームのキャスターのuserIDが含まれていますので、enterRoom() 成功後、直接startPlay(userID)を呼び出せばキャスターの画面を再生できます。
/// お客様が管理するルームリストにroomidしかなくてもかまいません。視聴者は、enterRoom()成功後すぐに TRTCLiveRoomDelegateからonAnchorEnter(userID)のコールバックを受信します。
/// この時コールバックの中のuserIDを使用して startPlay(userID)を呼び出せば、キャスターの画面を再生できます。
```

©2013-2019 Tencent Cloud. All rights reserved.

```
/// - Parameters:
/// - roomID: ルームID。
/// - useCDNFirst: 優先的にCDNを使用して再生するかどうか。
/// - cdnDomain: CDNドメイン名
/// - callback: 入室結果のコールバック。成功時にcodeは0になります。
/// - Note:
/// - 視聴者がライブストリーミングルームに参加する時の呼び出し
/// - キャスターがこのインターフェースを呼び出して自分が作成したルームに参加することはできません。createRoomを使用する必要があります。
- (void)enterRoomWithRoomID:(UInt32)roomID
useCDNFirst:(BOOL)useCDNFirst
cdnDomain:(NSString * _Nullable)cdnDomain
callback:(Callback)callback
NS_SWIFT_NAME(enterRoom(roomID:useCDNFirst:cdnDomain:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomID	UInt32	ルームID。
callback	(_ code: Int, _ message: String?) -> Void	入室結果のコールバック。成功時にcodeは0になります。

視聴者のライブストリーミング視聴の通常の呼び出しフローは以下のとおりです。

1.【視聴者】サーバーから取得する最新のライブストリーミングルームリストには、多くのライブストリーミングルームのroomIdおよびルーム情報が含まれる場合があります。
2. 【視聴者】視聴者は1つのライブストリーミングルームを選択し、 `enterRoom()` を呼び出して、このルームに参加します。
3. 【視聴者】 `startPlay(userID)` を呼び出し、キャスターのuserIDを渡して再生を開始します。

- ライブストリーミングルームリストにキャスターのuserID情報がすでに含まれている場合、視聴者は、 `startPlay(userID)` を直接呼び出せば、再生を開始できます。
- ルーム参加前にキャスターのuserIDが一時的に取得できない場合、視聴者はルーム参加後に `TRTCLiveRoomDelegate` から `onAnchorEnter(userID)` のイベントコールバックを受信します。このコールバックの中にキャスターのuserID 情報が含まれていますので、 `startPlay(userID)` を再び呼び出せば、再生できます。

exitRoom

ルームを退出します。

```
/// 退室（視聴者が呼び出し）
/// - Parameter callback: 退室結果のコールバック。成功時にcodeは0になります。
```

©2013-2019 Tencent Cloud. All rights reserved.

```
/// - Note:
/// - 視聴者がライブストリーミングルームを退出する時の呼び出し
/// - キャスターはこのインターフェースを呼び出してルームを退出できません。
- (void)exitRoom:(Callback _Nullable)callback
NS_SWIFT_NAME(exitRoom(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String?) -> Void	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfos

ルームリストの詳細情報を取得します。ルーム情報は、キャスターが `createRoom()` 作成時にroomInfoによって設定します。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

```
/// ルームリストの詳細情報を取得します。
/// その中の情報は、キャスターがcreateRoom()作成時に roomInfoで設定します。ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。
/// - Parameter roomIDs: ルームナンバーリスト
/// - Parameter callback: ルーム詳細情報のコールバック
- (void)getRoomInfosWithRoomIDs:(NSArray<NSNumber *> *)roomIDs
callback:(RoomInfoCallback _Nullable)callback
NS_SWIFT_NAME(getRoomInfos(roomIDs:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIDs	[UInt32]	ルームナンバーリスト。
callback	(_ code: Int, _ message: String?, _ roomList: [TRTCLiveRoomInfo]) -> Void	ルーム詳細情報のコールバック。

getAnchorList

ルーム内の全キャスターのリストを取得します。 `enterRoom()` 成功後に呼び出しが有効となります。

©2013-2019 Tencent Cloud. All rights reserved.

```
/// ルーム内の全キャスターのリストを取得します。enterRoom() 成功後に呼び出しが有効となります。
/// - Parameter callback: ユーザーの詳細情報のコールバック
- (void)getAnchorList:(UserListCallback _Nullable)callback
NS_SWIFT_NAME(getAnchorList(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String, _ userList: [TRTCLiveUserInfo]) -> Void	ユーザーの詳細情報のコールバック。

getAudienceList

ルーム内の全視聴者の情報を取得します。 `enterRoom()` 成功後に呼び出しが有効となります。

```
/// ルーム内の全視聴者の情報を取得します。enterRoom()成功後に呼び出しが有効となります。
/// - Parameter callback: ユーザーの詳細情報のコールバック
- (void)getAudienceList:(UserListCallback _Nullable)callback
NS_SWIFT_NAME(getAudienceList(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String, _ userList: [TRTCLiveUserInfo]) -> Void	ユーザーの詳細情報のコールバック。

プッシュプルストリームに関連するインターフェース関数

startCameraPreview

ローカルビデオのプレビュー画面を立ち上げます。

```
/// ローカルビデオのプレビュー画面を立ち上げます
/// - Parameters:
/// - frontCamera: true：フロントカメラ；false：リアカメラ。
/// - view: ビデオ画像をロードするウィジェット。
/// - callback: 操作コールバック。
- (void)startCameraPreviewWithFrontCamera:(BOOL)frontCamera
view:(UIView *)view
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(startCameraPreview(frontCamera:view:callback:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
frontCamera	Bool	true：フロントカメラ；false：リアカメラ。
view	UIView	ビデオ画像をロードするウィジェット。
callback	(_ code: Int, _ message: String?) -> Void	操作コールバック。

stopCameraPreview

ローカルのビデオキャプチャおよびプレビューを停止します。

```
/// ローカルのビデオキャプチャおよびプレビューを停止します。
- (void)stopCameraPreview;
```

startPublish

ライブストリーミング（プッシュ）を開始します。次のシーンに適用します。

- キャスターの配信開始時のコール
- 視聴者のマイク接続開始時の呼び出し

```
/// ライブストリーミング（プッシュ）を開始します。次の2種類のシーンに適用します。
/// 1. キャスターの配信開始時の呼び出し
/// 2. 視聴者のマイク接続開始時の呼び出し
/// - Parameters:
/// - streamID: ライブCDNのstreamIdをバインドするために利用します。視聴者にライブCDN経由で視聴させたい場合は、現在のキャスターのライブストリーミングstreamIdを指定する必要があります。
/// - callback: 操作コールバック
- (void)startPublishWithStreamID:(NSString *)streamID
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(startPublish(streamID:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
streamID	String	ライブCDNのstreamIdをバインドするために利用します。視聴者にライブCDN経由で視聴させたい場合は、現在のキャスターのライブストリーミングstreamIdを指定する必要があります。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String?) -> Void	操作コールバック。

stopPublish

ライブストリーミング（プッシュ）を停止します。次のシーンに適用します。

- キャスターがライブストリーミングを終了する時の呼び出し。
- 視聴者がマイク接続を終了する時の呼び出し。

```
/// ライブストリーミング（プッシュ）を停止します。次の2つのシーンに適用します。
/// 1. キャスターがライブストリーミングを終了する時の呼び出し
/// 2. 視聴者がマイク接続を終了する時の呼び出し
/// - Parameter callback: 操作コールバック。
- (void)stopPublish:(Callback _Nullable)callback
NS_SWIFT_NAME(stopPublish(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String?) -> Void	操作コールバック。

startPlay

リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。

```
/// リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。
/// 【普通の視聴シーン】
/// 1. お客様のサーバーが管理するルームリストの中にそれぞれのルームのキャスターのuserIDが含まれていますので、enterRoom() 成功後、直接 startPlay(userID)を呼び出せばキャスターの画面を再生できます。
/// 2. お客様が管理するルームリストにroomidしかなくてもかまいません。視聴者はenterRoom()成功後すぐにTRTCLiveRoomDelegateからのonAnchorEnter(userID)のコールバックを受信します。
/// この時コールバックの中のuserIDを使用して startPlay(userID)を呼び出せば、キャスターの画面を再生できます。
/// 【ライブストリーミングのマイク接続シーン】
/// マイク接続の始動後、キャスターは TRTCLiveRoomDelegateから onAnchorEnter(userID)のコールバックを受信します。この時コールバックの中のuserIDを使用して startPlay(userID)を呼び出せば、マイク接続の画面を再生できます。
/// - Parameters:
```

©2013-2019 Tencent Cloud. All rights reserved.

```
/// - userID: 視聴が必要なユーザーのID。
/// - view: ビデオ画像をロードする view ウィジェット。
/// - callback: 操作コールバック。
- (void)startPlayWithUserID:(NSString *)userID
view:(UIView *)view
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(startPlay(userID:view:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	視聴が必要なユーザーのID。
view	UIView	ビデオ画像をロードするviewウィジェット。
callback	(_ code: Int, _ message: String?) -> Void	操作コールバック。

普通の視聴シーン

- ライブストリーミングルームリストにキャスターのuserID情報がすでに含まれている場合、視聴者は、 `enterRoom()` 成功後に `startPlay(userID)` を直接呼び出せば、キャスターの画面を再生できます。
- ルーム参加前にキャスターのuserIDが一時的に取得できない場合、視聴者はルーム参加後に `TRTCLiveRoomDelegate` から `onAnchorEnter(userID)` のイベントコールバックを受信します。このコールバックの中にキャスターのuserID 情報が含まれていますので、 `startPlay(userID)` を再び呼び出せば、キャスターの画面を再生できます。

ライブストリーミングのマイク接続シーン

マイク接続の始動後、キャスターは `TRTCLiveRoomDelegate` から `onAnchorEnter(userID)` のコールバックを受信します。この時コールバックの中のuserIDを使用して `startPlay(userID)` を呼び出せば、マイク接続の画面を再生できます。

stopPlay

リモートのビデオ画面のレンダリングを停止します。 `onAnchorExit()` のコールバック時、このインターフェースを呼び出す必要があります。

```
/// リモートのビデオ画面のレンダリングを停止。
/// - Parameters:
/// - userID: 相手側のユーザー情報。
/// - callback: 操作コールバック。
/// - Note:
```

©2013-2019 Tencent Cloud. All rights reserved.

```
/// - onAnchorExitのコールバック時、このインターフェースを呼び出します。
- (void)stopPlayWithUserID:(NSString *)userID
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(stopPlay(userID:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	相手側のユーザー情報。
callback	(_ code: Int, _ message: String?) -> Void	操作コールバック。

キャスターと視聴者のマイク接続

requestJoinAnchor

視聴者がマイク接続をリクエストします。

```
/// 視聴者側がマイク接続をリクエストします
/// - Parameters:
/// - reason: マイク接続リクエストの理由。
/// - responseCallback: マイク接続リクエストのコールバック。
/// - Note: 視聴者がリクエストを送信すると、キャスター側は`onRequestJoinAnchor`のコールバックを受信します。
- (void)requestJoinAnchor:(NSString *)reason
timeout:(double)timeout
responseCallback:(ResponseCallback _Nullable)responseCallback
NS_SWIFT_NAME(requestJoinAnchor(reason:timeout:responseCallback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
reason	String	マイク接続の理由。
timeout	long	キャスターの応答のコールバック。
responseCallback	(_ agreed: Bool, _ reason: String?) -> Void	キャスターの応答のコールバック。

キャスターと視聴者のマイク接続フローは次のとおりです。

1. 【視聴者】 `requestJoinAnchor()` を呼び出し、キャスターにマイク接続リクエストを送信します。

©2013-2019 Tencent Cloud. All rights reserved.

2. 【キャスター】 `TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバック通知を受信します。
3. 【キャスター】 `responseJoinAnchor()` を呼び出して、視聴者からのマイク接続リクエストを受け入れるか決定します。
4. 【視聴者】responseCallbackのコールバック通知を受信します。この通知にはキャスターの処理結果が含まれています。
5. 【視聴者】リクエストが同意されると、 `startCameraPreview()` が呼び出され、ローカルカメラが起動します。
6. 【視聴者】 `startPublish()` を呼び出し、正式にプッシュストリームの状態に入ります。
7. 【キャスター】一度視聴者がマイク接続に入ると、キャスターは、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を受信します。
8. 【キャスター】キャスターが `startPlay()` を呼び出すと、マイク接続の視聴者のビデオ画面を見ることができるようになります。
9. 【視聴者】ライブストリーミングルームの中で他の視聴者がすでにキャスターとマイク接続を行っている場合、新しく参加したマイク接続の視聴者は `onAnchorEnter()` の通知を受け取ります。この時に `startPlay()` を呼び出せば他のマイク接続者のビデオ画面を再生することができます。

responseJoinAnchor

キャスターがマイク接続のリクエストを処理します。キャスターは `TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバックを受け取った後、このインターフェースを呼び出して、視聴者のマイク接続リクエストを処理する必要があります。

```
/// キャスターが視聴者のマイク接続リクエストに回答します
/// - Parameters:
/// - user: 視聴者ID。
/// - agree: true：同意；false：拒否。
/// - reason: マイク接続を同意/拒否した理由の説明。
/// - Note: キャスターが回答すると、視聴者側は`requestJoinAnchor`で渡された`responseCallback`のコールバックを受信します。
- (void)responseJoinAnchor:(NSString *)userID
agree:(BOOL)agree
reason:(NSString *)reason
NS_SWIFT_NAME(responseJoinAnchor(userID:agree:reason:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	視聴者ID。
agree	Bool	true：同意；false：拒否。
reason	String?	マイク接続を同意/拒否した理由の説明。

©2013-2019 Tencent Cloud. All rights reserved.

kickoutJoinAnchor

キャスターがマイク接続の視聴者をキックアウトします。キャスターがこのインターフェースを呼び出し、マイク接続の視聴者をキックアウトすると、キックアウトされたマイク接続の視聴者は、`TRTCLiveRoomDelegate` の `onKickoutJoinAnchor()` のコールバック通知を受信します。

```
/// キャスターがマイク接続の視聴者を追放します
/// - Parameters:
/// - userID: マイク接続の視聴者ID。
/// - callback: 操作コールバック。
/// - Note: キャスターがこのインターフェースを呼び出し、マイク接続の視聴者を追放すると、追放されたマイク接続の視聴者は、trtcLiveRoomOnKickoutJoinAnchor()のコールバック通知を受信します。
- (void)kickoutJoinAnchor:(NSString *)userID
callback:(Callback _Nullable)callback
NS_SWIFT_NAME(kickoutJoinAnchor(userID:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	マイク接続の視聴者ID。
callback	(_ code: Int, _ message: String?) -> Void	操作コールバック。

キャスターのルーム間PK

requestRoomPK

キャスターがルーム間PKをリクエストします。

```
/// キャスターがルーム間PKをリクエストします。
/// - Parameters:
/// - roomID: 招待された側のルームID。
/// - userID: 招待された側のキャスターID。
/// - responseCallback: ルーム間PKのリクエスト結果のコールバック。
/// - Note: リクエストを送信すると、相手側のキャスターが`onRequestRoomPK`のコールバックを受信します。
- (void)requestRoomPKWithRoomID:(UInt32)roomID
userID:(NSString *)userID
responseCallback:(ResponseCallback _Nullable)responseCallback
NS_SWIFT_NAME(requestRoomPK(roomID:userID:responseCallback:));
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomID	UInt32	招待された側のルームID。
userID	String	招待された側のキャスターID。
responseCallback	(_ agreed: Bool, _ reason: String?) -> Void	ルーム間PKのリクエスト結果のコールバック。

キャスターとキャスターの間でルーム間PKを行うことができます。ライブストリーミング中の2名のキャスターAとBのルーム間PKのフローは次のとおりです。

1. 【キャスター A】 `requestRoomPK()` を呼び出してキャスターBに向けてマイク接続リクエストを送信します。
2. 【キャスター B】 `TRTCLiveRoomDelegate` の `onRequestRoomPK()` のコールバック通知を受信します。
3. 【キャスター B】 `responseRoomPK()` を呼び出して、キャスター AのPKのリクエストを受け入れるかどうか決定します。
4. 【キャスターB】キャスターAのリクエストを受け入れる場合は、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出してキャスターAのビデオ画面を表示させます。
5. 【キャスター A】 `responseCallback` のコールバック通知を受信します。この通知にはキャスター B の処理結果が含まれます。
6. 【キャスターA】リクエストが同意された場合は、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出し、キャスターBのビデオ画面を表示します。

responseRoomPK

キャスターがルーム間PKのリクエストに応答します。キャスターが応答した後、相手側キャスターは `requestRoomPK` で渡された `responseCallback` のコールバックを受信します。

```
/// ルーム間PKのリクエストに応答します。
/// キャスターが他のルームのキャスターのPKのリクエストに応答します。
/// - Parameters:
/// - user: PKをリクエストしたキャスターID
/// - agree: true：同意；false：拒否
/// - reason: PKに同意/拒否した理由の説明
/// - Note: キャスターの回答後、相手側キャスターは`requestRoomPK`で渡された`responseCallback`のコールバックを受信します。
- (void)responseRoomPKWithUserID:(NSString *)userID
agree:(BOOL)agree
reason:(NSString *)reason
NS_SWIFT_NAME(responseRoomPK(userID:agree:reason:));
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userID	String	PKをリクエストしたキャスターID。
agree	Bool	true：同意；false：拒否。
reason	String?	PKに同意/拒否した理由の説明。

quitRoomPK

ルーム間PKから退出します。PK中のいずれかのキャスターがルーム間PK 状態から退出すると、もう一方のキャスターは `TRTCLiveRoomDelegate` の `trtcLiveRoomOnQuitRoomPK()` のコールバック通知を受信します。

```
/// キャスターがルーム間PKから退出します。
/// - Parameter callback: ルーム間PK退出の結果のコールバック
/// - Note: 2名のキャスターのうちのいずれかがルーム間PKの状態から退出した後、もう一方のキャスターは `trtcLiveRoomOnQuitRoomPK`のコールバック通知を受信します。
- (void)quitRoomPK:(Callback _Nullable)callback
NS_SWIFT_NAME(quitRoomPK(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String?) -> Void	操作コールバック。

オーディオビデオ制御に関連するインターフェース関数

switchCamera

フロント/リアカメラを切り替えます。

```
/// フロント/リアカメラを切り替えます
- (void)switchCamera;
```

setMirror

ミラー表示のオン/オフを設定します。

```
/// ミラー表示のオン/オフを設定します
/// - Parameter isMirror: ミラーオン／オフ。
- (void)setMirror:(BOOL)isMirror
NS_SWIFT_NAME(setMirror(isMirror:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMirror	Bool	ミラーオン/オフ。

muteLocalAudio

ローカルオーディオをミュートにします。

```
/// ローカルオーディオをミュートにします。
/// - Parameter isMuted: true：ミュートオン；false：ミュートオフ
- (void)muteLocalAudio:(BOOL)isMuted
NS_SWIFT_NAME(muteLocalAudio(isMuted:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMuted	Bool	true：ミュート起動、false：ミュート停止。

muteRemoteAudio

リモートオーディオをミュートにします。

```
/// リモートオーディオをミュートにします
/// - Parameters:
/// - userID: リモートのユーザーID。
/// - isMuted: true：ミュートオン；false：ミュートオフ
- (void)muteRemoteAudioWithUserID:(NSString *)userID isMuted:(BOOL)isMuted
NS_SWIFT_NAME(muteRemoteAudio(userID:isMuted:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	リモートのユーザーのID。
isMuted	Bool	true：ミュート起動、false：ミュート停止。

muteAllRemoteAudio

すべてのリモート側のオーディオをミュートにします。

©2013-2019 Tencent Cloud. All rights reserved.

```
/// すべてのリモート側のオーディオをミュートにします
/// - Parameter isMuted: true：ミュートオン；false：ミュートオフ
- (void)muteAllRemoteAudio:(BOOL)isMuted
NS_SWIFT_NAME(muteAllRemoteAudio(_:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMuted	Bool	true：ミュート起動、false：ミュート停止。

setAudioQuality

音質の設定

```
/// 音質を設定します。サポートする値は1 2 3で、低中高を表します
/// - Parameter quality音質
- (void)setAudioQuality:(NSInteger)quality
NS_SWIFT_NAME(setAudioiQuality(quality:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	NSInteger	1：音声、2：標準、3：音楽

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

BGMサウンドエフェクト管理オブジェクト TXAudioEffectManagerを取得します。

```
/// サウンドエフェクト管理オブジェクトを取得します。
- (TXAudioEffectManager *)getAudioEffectManager;
```

美顔フィルタに関するインターフェース関数

getBeautyManager

美顔管理オブジェクトTXBeautyManagerを取得します。 |

©2013-2019 Tencent Cloud. All rights reserved.

```
/* 美顔管理オブジェクト TXBeautyManagerを取得します。
*
* 美顔管理では、次の機能を利用できます。
* - 「美顔のスタイル」、「美白」、「肌色補正（血色・つや感）」、「デカ眼」、「顔痩せ」、「V顔」、「下あご」、「面長補正」、「小鼻」、「キラキラ目」、「白い歯」、「目の弛み除去」、「シワ除去」、「ほうれい線除去」などの美容効果を設定します。
* - 「髪の生え際」、「眼と眼の距離」、「眼の角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
* - 人の顔のスタンプ（素材）等のダイナミック効果を設定します
* - メイクアップを追加します
* - ジェスチャー認識を行います
*/
- (TXBeautyManager *)getBeautyManager;
```

美顔管理では、次の機能を使用できます。

- 「美顔のスタイル」、「美白」、「肌色補正（血色・つや感）」、「デカ眼」、「顔痩せ」、「V顔」、「下あご」、「面長補正」、「小鼻」、「キラキラ目」、「白い歯」、「目の弛み除去」、「シワ除去」、「ほうれい線除去」などの美容効果を設定します。
- 「髪の生え際」、「眼と眼の距離」、「眼の角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
- 人の顔のスタンプ（素材）等のダイナミック効果を設定します。
- メイクアップを追加します。
- ジェスチャー認識を行います。

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

```
/// 送信したテキストメッセージは、ルーム内の全参加者が見ることができます。
/// - Parameters:
/// - message: テキストメッセージ。
/// - callback: 送信コールバック。
- (void)sendRoomTextMsg:(NSString *)message callback:(Callback _Nullable)callback
NS_SWIFT_NAME(sendRoomTextMsg(message:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	(_ code: Int, _ message: String?) -> Void	結果のコールバックを送信

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

```
/// カスタムメッセージを送信します
/// - Parameters:
/// - command: コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます
/// - message: テキストメッセージ。
/// - callback: 送信コールバック。
- (void)sendRoomCustomMsgWithCommand:(NSString *)command message:(NSString *)message callback:(Callback _Nullable)callback
NS_SWIFT_NAME(sendRoomCustomMsg(command:message:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
command	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	(_ code: Int, _ message: String?) -> Void	結果のコールバックを送信。

デバックに関するインターフェース関数

showVideoDebugLog

インターフェースにdebug 情報を表示するかどうか。

```
/// インターフェースにdebug情報を表示するかどうか
/// - Parameter isShow: Debug情報表示のオン/オフ。
- (void)showVideoDebugLog:(BOOL)isShow
NS_SWIFT_NAME(showVideoDebugLog(_:));
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
isShow	Bool	Debug情報表示のオン/オフ。

TRTCLiveRoomDelegateイベントのコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
code	Int	エラーコード。
message	String?	エラー情報。

onWarning

警告のコールバック。

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

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
code	Int	エラーコード TRTCWarningCode。
message	String?	警告メッセージ。

onDebugLog

Logコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
log	String	ログ情報。

ルームイベントのコールバック

onRoomDestroy

ルームが破棄された時のコールバック。キャスターが退室する時、ルーム内の全ユーザーがこの通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
roomID	String	ルームID。

©2013-2019 Tencent Cloud. All rights reserved.

onRoomInfoChange

ライブストリーミングルーム情報変更のコールバック。ライブストリーミングのマイク接続やPKで、ルーム状態の変化を通知するシーンに多用されます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
info	TRTCLiveRoomInfo	ルーム情報。

キャスターと視聴者の参加/退出のイベントコールバック

onAnchorEnter

新しいキャスターの入室の通知を受信します。マイク接続の視聴者とルーム間PKのキャスターが入室すると、視聴者は、新しいキャスターが入室したというイベントを受信します。 `TRTCLiveRoom` の `startPlay()` を呼び出して、そのキャスターのビデオ画面を表示することができます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
userID	String	新しくルームに参加したユーザーID。

onAnchorExit

キャスターの退室の通知を受信します。ルーム内のキャスター（およびマイク接続中の視聴者）が、新しいキャスターの退室のイベントを受信します。 `TRTCLiveRoom` の `stopPlay()` を呼び出して、そのキャスターのビデオ画面を閉じることができます。

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
userID	String	ルームを退出したユーザーID。

onAudienceEnter

視聴者入室通知の受信。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
user	TRTCLiveUserInfo	入室した視聴者の情報。

onAudienceExit

視聴者退室通知の受信。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
user	TRTCLiveUserInfo	退室した視聴者の情報。

©2013-2019 Tencent Cloud. All rights reserved.

キャスターと視聴者のマイク接続のイベントコールバック

onRequestJoinAnchor

キャスターが視聴者のマイク接続リクエストを受信した時のコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
user	TRTCLiveUserInfo	マイク接続をリクエストした視聴者の情報。
reason	String?	マイク接続の理由の説明。
timeout	Double	リクエスト処理のタイムアウトの時間。

onKickoutJoinAnchor

マイク接続の視聴者がマイク接続からキックアウトされた通知を受信します。マイク接続の視聴者は、キャスターからマイク接続キックアウトのメッセージを受信した場合、 `TRTCLiveRoom` の `stopPublish()` を呼び出してマイク接続から退出する必要があります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。

キャスター間PKのイベントコールバック

onRequestRoomPK

©2013-2019 Tencent Cloud. All rights reserved.

ルーム間PKのリクエストの通知を受信します。キャスターは、他のルームのキャスターからPKのリクエストを受け取り、PKに同意する場合、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出して招待側キャスターのストリームを再生する必要があります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
user	TRTCLiveUserInfo	ルーム間マイク接続を始動したキャスターの情報。
timeout	Double	リクエスト処理のタイムアウトの時間。

onQuitRoomPK

ルーム間PK中止の通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

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

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
message	String	テキストメッセージ。
user	TRTCLiveUserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージの受信。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
trtcLiveRoom	TRTCLiveRoomImpl	現在のTRTCLiveRoomコンポーネントインスタンス。
command	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
user	TRTCLiveUserInfo	送信者のユーザー情報。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCLiveRoom API (Android)

最終更新日：：2021-12-24 17:12:16

TRTCLiveRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- キャスターが新しいライブストリーミングルームを作成して配信を開始し、視聴者がライブストリーミングルームに参加して視聴します。
- キャスターと視聴者がビデオでマイク接続によるコラボライブを行います。
- 2つの異なるルームの間でキャスター同士のPK（クロスルーム対戦）を行います。
- 各種のテキストメッセージやカスタムメッセージの送信をサポートします。カスタムメッセージは弾幕、「いいね」、ギフトを実装するために使用することができます。

TRTCLiveRoom はオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実現プロセスは、ビデオマイク接続ライブストリーミング（Android）をご参照ください。

- TRTC SDK： TRTC SDK を低遅延のライブストリーミングコンポーネントとして使用します。
- IM SDK： IM SDKのAVChatroomを使用して、ライブストリーミングチャットルーム機能を実装します。同時に、IMのメッセージでキャスター間のマイク接続のフローをつなげます。

TRTCLiveRoom API概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを破棄します。
setDelegate	イベントコールバックを設定します。
setDelegateHandler	イベントコールバックが設定されているスレッドです。
login	ログイン。
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
createRoom	ルームの作成（キャスターが呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動作成します。
destroyRoom	ルームの破棄（キャスターが呼び出し）。
enterRoom	入室（視聴者が呼び出し）。
exitRoom	退室（視聴者が呼び出し）。
getRoomInfos	ルームリストの詳細情報を取得します。
getAnchorList	ルーム内の全キャスターのリストを取得します。enterRoom()成功後に呼び出しが有効になります。
getAudienceList	ルーム内の全視聴者の情報を取得します。enterRoom()成功後に呼び出しが有効になります。

プッシュプルストリームに関連するインターフェース関数

API	説明
startCameraPreview	ローカルビデオのプレビュー画面を立ち上げます。
stopCameraPreview	ローカルのビデオキャプチャおよびプレビューを停止します。
startPublish	ライブストリーミング（プッシュ）を開始します。
stopPublish	ライブストリーミング（プッシュ）を停止します。
startPlay	リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。
stopPlay	リモートのビデオ画面のレンダリングを停止します。

キャスターと視聴者のマイク接続

API	説明
requestJoinAnchor	視聴者がマイク接続をリクエストします。
responseJoinAnchor	キャスターがマイク接続のリクエストを処理します。
kickoutJoinAnchor	キャスターがマイク接続の視聴者をキックアウトします。

©2013-2019 Tencent Cloud. All rights reserved.

キャスターのルーム間PK

API	説明
requestRoomPK	キャスターがルーム間のPKをリクエストします。
responseRoomPK	キャスターがルーム間PKのリクエストに応答します。
quitRoomPK	ルーム間PKから退出します。

オーディオビデオ制御に関連するインターフェース関数

API	説明
switchCamera	フロント/リアカメラを切り替えます。
setMirror	ミラー表示のオン/オフを設定します。
muteLocalAudio	ローカルオーディオをミュートにします。
muteRemoteAudio	リモートオーディオをミュートにします。
muteAllRemoteAudio	全てのリモート側のオーディオをミュートにします。

BGMサウンドエフェクト関連インターフェース関数

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerを取得します。

美顔フィルタに関するインターフェース関数

API	説明
getBeautyManager	美顔管理オブジェクトTXBeautyManagerを取得します。

メッセージ送信関連インターフェース関数

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

©2013-2019 Tencent Cloud. All rights reserved.

デバックに関するインターフェース関数

API	説明
showVideoDebugLog	インターフェースにdebug情報を表示するか。

TRTCLiveRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントコールバック

API	説明
onRoomDestroy	ルームが破棄された時のコールバック。
onRoomInfoChange	ライブストリーミングルーム情報変更のコールバック。

キャスターと視聴者の参加/退出のイベントコールバック

API	説明
onAnchorEnter	新しいキャスターの入室通知を受信します。
onAnchorExit	キャスターの退室通知を受信します。
onAudienceEnter	視聴者入室通知の受信。
onAudienceExit	視聴者退室通知の受信。

キャスターと視聴者のマイク接続のイベントコールバック

API	説明
onRequestJoinAnchor	キャスターが視聴者のマイク接続リクエストを受信した時のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onKickoutJoinAnchor	マイク接続の視聴者がマイク接続からキックアウトされた通知を受信します。

キャスター間PKのイベントコールバック

API	説明
onRequestRoomPK	ルーム間PKのリクエストの通知を受信。
onQuitRoomPK	ルーム間PK中止の通知を受信します。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

SDK基本関数

sharedInstance

TRTCLiveRoom シングルトンオブジェクトを取得します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
context	Context	Androidコンテキスト。内部ではApplicationContextに変換してシステムAPIの呼び出しに使用します

destroySharedInstance

TRTCLiveRoom シングルトンオブジェクトを破棄します。

> 説明：
> インスタンスの破棄後は、外部にキャッシュされたTRTCLiveRoomインスタンスの再使用ができません。
> 改めて sharedInstance をコールして、インスタンスを新規取得する必要があります。

©2013-2019 Tencent Cloud. All rights reserved.

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

setDelegate

TRTCLiveRoom イベントコールバック。 TRTCLiveRoomDelegateを介してTRTCLiveRoomの各種ステータス通知を受け取ることができます。

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

説明：
setDelegateはTRTCLiveRoomのプロキシコールバックです。

setDelegateHandler

イベントコールバックが配置されているスレッドを設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
handler	Handler	TRTCLiveRoomの中の各種状態の通知のコールバックは、このhandlerによって通知されます。setDelegateと混用しないでください。

login

ログイン。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
sdkAppId	int	TRTCコンソール >【アプリケーション管理】> アプリケーション情報の中でSDKAppIDを確認できます。
user	String	現在のユーザーID、文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、 UserSigの計算方法をご参照ください。
config	TRTCLiveRoomConfig	グローバルコンフィギュレーション情報。ログイン時に初期化してください。ログイン後は変更できなくなります。 • useCDNFirst属性：視聴者の視聴方式の設定に使用します。trueは、普通の視聴者のCDN経由での視聴を表し、費用は安価ですがディレーは高めです。falseは、普通の視聴者の低遅延による視聴を表し、費用はCDNとマイク接続の間ですが、ディレーを1s以内に抑えることができます。 • CDNPlayDomain属性：useCDNFirstの設定をtrueにすると有効になり、CDN経由で視聴する再生ドメイン名の指定に利用します。CSSコンソール>【ドメイン名管理】の画面からログインし、設定を行います。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

logout

ログアウト。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ログアウトのコールバック。成功時にcodeは0になります。

setSelfProfile

個人情報の修正。

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

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像のアドレス。
callback	ActionCallback	個人情報設定のコールバック。成功時にcodeは0になります。

ルーム関連インターフェース関数

createRoom

ルームの新規作成（キャスターが呼び出し）。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomID を、1つのライブストリーミングルームリストにまとめることができます。Tencent Cloudではライブストリーミングルームリストの管理サービスを一時的に行いませんので、ご自身でライブストリーミングルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報は、ルーム名、マイク情報、カバー情報などのルーム説明の情報に使用します。ルームリストおよびルーム情報をご自身のサーバーで管理する場合は、このパラメータは無視してもかまいません。
callback	ActionCallback	ルームの作成結果のコールバック。成功時にcodeは0になります。

キャスターが配信を開始する際の通常の呼び出しプロセスは次のとおりです。

1. 【キャスター】 `startCameraPreview()` を呼び出し、カメラのプレビューを起動します。この時美顔パラメータを調整できます。
2. 【キャスター】 `createRoom()` を呼び出し、ライブストリーミングルームを作成します。ルーム作成の成功の有無がActionCallbackでキャスターに通知されます。

©2013-2019 Tencent Cloud. All rights reserved.

3. 【キャスター】 `starPublish()` を呼び出し、プッシュを開始します。

destroyRoom

ルームの破棄（キャスターが呼び出し）。キャスターは、ルーム作成後、この関数を呼び出してルームを破棄します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

enterRoom

入室（視聴者が呼び出し）。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

視聴者のライブストリーミング視聴の通常の呼び出しフローは以下のとおりです。
1.【視聴者】サーバーから取得する最新のライブストリーミングルームリストには、多くのライブストリーミングルームのroomIdおよびルーム情報が含まれる場合があります。
2. 【視聴者】視聴者は1つのライブストリーミングルームを選択し、 `enterRoom()` を呼び出して、このルームに参加します。
3. 【視聴者】 `startPlay(userId)` を呼び出し、キャスターのuserIdを渡して再生を開始します。

- ライブストリーミングルームリストにキャスターのuserId情報がすでに含まれている場合、視聴者は、 `startPlay(userId)` を直接呼び出せば、再生を開始できます。
- ルーム参加前にキャスターのuserIDが一時的に取得できない場合、視聴者はルーム参加後に `TRTCLiveRoomDelegate` の中の `onAnchorEnter(userID)` のイベントコールバックを受信します。このコールバックの中にキャスターのuserID 情報が含まれていますので、 `startPlay(userID)` 再び呼び出しせば、再生できます。

©2013-2019 Tencent Cloud. All rights reserved.

exitRoom

ルームを退出します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfos

ルームリストの詳細情報を取得します。ルーム情報は、キャスターが `createRoom()` 作成時にroomInfoによって設定します。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIdList	List<Integer>	ルームナンバーリスト。
callback	RoomInfoCallback	ルーム詳細情報のコールバック。

getAnchorList

ルーム内の全キャスターのリストを取得します。 `enterRoom()` 成功後に呼び出しが有効となります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	UserListCallback	ユーザーの詳細情報のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

getAudienceList

ルーム内の全視聴者の情報を取得します。 `enterRoom()` 成功後に呼び出しが有効となります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	UserListCallback	ユーザーの詳細情報のコールバック。

プッシュプルストリームに関連するインターフェース関数

startCameraPreview

ローカルビデオのプレビュー画面を立ち上げます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFront	boolean	true：フロントカメラ；false：リアカメラ。
view	TXCloudVideoView	ビデオ画像をロードするウィジェット。
callback	ActionCallback	操作コールバック。

stopCameraPreview

ローカルのビデオキャプチャおよびプレビューを停止します。

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

startPublish

ライブストリーミング（プッシュ）を開始します。次のシーンに適用します。

- キャスターの配信開始時のコール
- 視聴者のマイク接続開始時の呼び出し

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
streamId	String	ライブCDNのstreamIdをバインドするために利用します。視聴者にライブCDN経由で視聴させたい場合は、現在のキャスターのライブストリーミングstreamIdを指定する必要があります。
callback	ActionCallback	操作コールバック。

stopPublish

ライブストリーミング（プッシュ）を停止します。次のシーンに適用します。

- キャスターがライブストリーミングを終了する時の呼び出し
- 視聴者がマイク接続を終了する時の呼び出し

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

startPlay

リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	視聴が必要なユーザーid。
view	TXCloudVideoView	ビデオ画像をロードするviewウィジェット。
callback	ActionCallback	操作コールバック。

©2013-2019 Tencent Cloud. All rights reserved.

普通の視聴シーン

- ライブストリーミングルームリストにキャスターのuserId情報がすでに含まれている場合、視聴者は、 `enterRoom()` 成功後に `startPlay(userId)` を直接呼び出せばキャスターの画面を再生できます。
- ルーム参加前にキャスターのuserIDが一時的に取得できない場合、視聴者はルーム参加後に `TRTCLiveRoomDelegate` の中の `onAnchorEnter(userId)` のイベントコールバックを受信します。このコールバックの中にキャスターのuserID情報が含まれていますので、 `startPlay(userId)` を再び呼び出せば、キャスターの画面を再生できます。

ライブストリーミングのマイク接続シーン

マイク接続の始動後、キャスターは `TRTCLiveRoomDelegate` から `onAnchorEnter(userId)` のコールバックを受信します。この時コールバックの中のuserIDを使用してstartPlay(userID)を呼び出せば、マイク接続の画面を再生できます。

stopPlay

リモートのビデオ画面のレンダリングを停止します。 `onAnchorExit()` のコールバック時、このインターフェースを呼び出す必要があります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	相手側のユーザー情報。
callback	ActionCallback	操作コールバック。

キャスターと視聴者のマイク接続

requestJoinAnchor

視聴者がマイク接続をリクエストします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
reason	String	マイク接続の理由。
timeout	int	タイムアウトの時間。
callback	ActionCallback	キャスターの応答のコールバック。

キャスターと視聴者のマイク接続フローは次のとおりです。

1. 【視聴者】 `requestJoinAnchor()` を呼び出し、キャスターにマイク接続リクエストを送信します。
2. 【キャスター】 `TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバック通知を受信します。
3. 【キャスター】 `responseJoinAnchor()` を呼び出して、視聴者からのマイク接続リクエストを受け入れるか決定します。
4. 【視聴者】responseCallbackのコールバック通知を受信します。この通知にキャスターの処理結果が含まれています。
5. 【視聴者】リクエストが同意されると、 `startCameraPreview()` が呼び出され、ローカルカメラが起動します。
6. 【視聴者】その後 `startPublish()` を呼び出し、正式にプッシュストリームの状態に入ります。
7. 【キャスター】一度視聴者がマイク接続に入ると、キャスターは、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を受信します。
8. 【キャスター】キャスターが `startPlay()` を呼び出すと、マイク接続の視聴者のビデオ画面を見ることができるようになります。
9. 【視聴者】ライブストリーミングルームの中で他の視聴者がすでにキャスターとマイク接続を行っている場合、新しく参加したマイク接続の視聴者は `onAnchorEnter()` の通知を受け取ります。この時に `startPlay()` を呼び出せば他のマイク接続者のビデオ画面を再生することができます。

responseJoinAnchor

キャスターがマイク接続のリクエストを処理します。キャスターは、 `TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバックを受け取った後、このインターフェースを呼び出して、視聴者のマイク接続リクエストを処理する必要があります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	視聴者ID。
agree	boolean	true：同意；false：拒否。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
reason	String	マイク接続に同意/拒否した理由の説明。

kickoutJoinAnchor

キャスターがマイク接続の視聴者をキックアウトします。キャスターがこのインターフェースを呼び出し、マイク接続の視聴者をキックアウトすると、キックアウトされたマイク接続の視聴者は、`TRTCLiveRoomDelegate` の `onKickoutJoinAnchor()` のコールバック通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	マイク接続の視聴者ID。
callback	ActionCallback	操作コールバック。

キャスターのルーム間PK

requestRoomPK

キャスターがルーム間PKをリクエストします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	招待された側のルームID。
userId	String	招待された側のキャスターID。
callback	ActionCallback	ルーム間PKのリクエスト結果のコールバック。

キャスターとキャスターの間でルーム間PKを行うことができます。ライブストリーミング中の2名のキャスターAとBのルーム間PKのフローは次のとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

1. 【キャスター A】 `requestRoomPK()` を呼び出してキャスターBに向けてマイク接続リクエストを送信します。
2. 【キャスター B】 `TRTCLiveRoomDelegate` の `onRequestRoomPK()` のコールバック通知を受信します。
3. 【キャスター B】 `responseRoomPK()` を呼び出して、キャスター AのPKのリクエストを受け入れるかどうか決定します。
4. 【キャスターB】キャスターAのリクエストを受け入れる場合は、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出してキャスターAのビデオ画面を表示させます。
5. 【キャスター A】 `responseCallback` のコールバック通知を受信します。この通知にはキャスター B の処理結果が含まれます。
6. 【キャスターA】リクエストが同意された場合は、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出し、キャスターBのビデオ画面を表示します。

responseRoomPK

キャスターがルーム間PKのリクエストに応答します。キャスターが応答した後、相手側キャスターは `requestRoomPK` で渡された `responseCallback` のコールバックを受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	PKをリクエストしたキャスターID。
agree	boolean	true：同意；false：拒否。
reason	String	PKを同意/拒否した理由の説明。

quitRoomPK

ルーム間PKから退出します。PK中のいずれかのキャスターがルーム間PKの状態から退出すると、もう一方のキャスターは `TRTCLiveRoomDelegate` の `onQuitRoomPk()` のコールバック通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

©2013-2019 Tencent Cloud. All rights reserved.

オーディオビデオ制御に関連するインターフェース関数

switchCamera

フロント/リアカメラを切り替えます。

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

setMirror

ミラー表示のオン/オフを設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMirror	boolean	ミラーオン/オフ。

muteLocalAudio

ローカルオーディオをミュートにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動、false：ミュート停止。

muteRemoteAudio

リモートオーディオをミュートにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	リモートユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動、false：ミュート停止。

muteAllRemoteAudio

すべてのリモート側のオーディオをミュートにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動、false：ミュート停止。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

BGMサウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

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

美顔フィルタに関するインターフェース関数

getBeautyManager

美顔管理オブジェクト TXBeautyManagerを取得します。

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

美顔管理では、次の機能を使用できます。

- 「美顔のスタイル」、「美白」、「肌色補正（血色・つや感）」、「デカ眼」、「顔痩せ」、「V顔」、「下あご」、「面長補正」、「小鼻」、「キラキラ目」、「白い歯」、「目の弛み除去」、「シワ除去」、「ほうれい線除去」などの美容効果を設定します。
- 「髪の生え際」、「眼と眼の距離」、「眼の角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
- 人の顔のスタンプ（素材）等のダイナミック効果を設定します。

©2013-2019 Tencent Cloud. All rights reserved.

- メイクアップを追加します。
- ジェスチャー認識を行います。

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

デバックに関するインターフェース関数

showVideoDebugLog

インターフェースの中にdebug情報を表示するかどうか。

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isShow	boolean	Debug情報表示のオン/オフ。

TRTCLiveRoomDelegateイベントのコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラー情報。

onWarning

警告のコールバック。

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

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	警告メッセージ。

onDebugLog

Logコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	ログ情報。

ルームイベントのコールバック

onRoomDestroy

ルームが破棄された時のコールバック。キャスターが退室する時、ルーム内の全ユーザーがこの通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	String	ルームID。

onRoomInfoChange

ライブストリーミングルーム情報変更のコールバック。ライブストリーミングのマイク接続やPKで、ルーム状態の変化を通知するシーンに多用されます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomInfo	TRTCLiveRoomInfo	ルーム情報。

キャスターと視聴者の参加/退出のイベントコールバック

onAnchorEnter

新しいキャスターの入室の通知を受信します。マイク接続の視聴者とルーム間PKのキャスターが入室すると、視聴者は、新しいキャスターが入室したというイベントを受信します。 `TRTCLiveRoom` の `startPlay()` を呼び出して、そのキャスターのビデオ画面を表示することができます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	新しくルームに参加したキャスターのID。

onAnchorExit

キャスターの退室の通知を受信します。ルーム内のキャスター（およびマイク接続中の視聴者）が、新しいキャスターの退室のイベントを受信します。 `TRTCLiveRoom` の `stopPlay()` を呼び出して、そのキャスターのビデオ画面を閉じることができます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ルームを退出したユーザーID。

onAudienceEnter

視聴者入室通知の受信。

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

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userInfo	TRTCLiveUserInfo	入室した視聴者の情報。

onAudienceExit

視聴者退室通知の受信。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	TRTCLiveUserInfo	退室した視聴者の情報。

キャスターと視聴者のマイク接続のイベントコールバック

onRequestJoinAnchor

キャスターが視聴者のマイク接続リクエストを受信した時のコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	TRTCLiveUserInfo	マイク接続をリクエストした視聴者の情報。
reason	String	マイク接続の理由の説明。
timeout	int	リクエスト処理のタイムアウトの時間。上の階層がこの時間をオーバーしても処理しない場合、このリクエストは自動的に破棄されます。

onKickoutJoinAnchor

マイク接続の視聴者がマイク接続からキックアウトされた通知を受信します。マイク接続の視聴者は、キャスターからマイク接続キックアウトのメッセージを受信した場合、 `TRTCLiveRoom` の `stopPublish()` を呼び出してマイク接続から退出する必要があります。

```
void onKickoutJoinAnchor();
```

©2013-2019 Tencent Cloud. All rights reserved.

キャスター間PKのイベントコールバック

onRequestRoomPK

ルーム間PKのリクエストの通知を受信します。キャスターは、他のルームのキャスターからPKのリクエストを受け取り、PKに同意する場合、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出して招待側キャスターのストリームを再生する必要があります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	TRTCLiveUserInfo	ルーム間マイク接続を始動したキャスターの情報。
timeout	int	リクエスト処理のタイムアウトの時間。

onQuitRoomPK

ルーム間PK中止の通知を受信します。

```
void onQuitRoomPK();
```

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
userInfo	TRTCLiveUserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージの受信。

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
command	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
userInfo	TRTCLiveUserInfo	送信者のユーザー情報。

TRTCAudioEffectManager

playBGM

BGMを再生します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
url	String	BGMファイルパス。
loopTimes	int	ループ回数
bgmVol	int	BGM音量
micVol	int	キャプチャ音量
notify	TRTCCloud.BGMNotify	再生の通知

stopBGM

BGMの再生を停止します。

```
void stopBGM();
```

pauseBGM

BGMの再生を一時停止します。

©2013-2019 Tencent Cloud. All rights reserved.

```
void pauseBGM();
```

resumeBGM

BGMの再生を続けます

```
void resumeBGM();
```

setBGMVolume

BGMの音量の大きさを設定します。BGMをミキシングして再生する時に使用し、BGMの音量の大きさの制御に利用します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	音量の大きさ。100は通常の音量を表し、値の範囲は0 - 100です。デフォルト値は100です。

setBGMPosition

BGM再生の進度を設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
position	int	BGM再生の進度。単位はミリ秒（ms）。

戻る

0：成功。

setMicVolume

マイク音量の大きさを設定します。BGMをミキシングして再生する時に使用し、マイク音量の大きさの制御に利用します。

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	Int	音量の大きさ。値の範囲は0 - 100、デフォルト値は100です。

setReverbType

リバーブ効果を設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
reverbType	int	リバーブタイプ。詳細は、 `TRTCCloudDef` の中のTRTC_REVERB_TYPEの定義をご参照ください。

setVoiceChangerType

ボイスチェンジのタイプを設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
type	int	リバーブタイプ。詳細は、 `TRTCCloudDef` の中のTRTC_VOICE_CHANGER_TYPEの定義をご参照ください。

playAudioEffect

サウンドエフェクトを再生します。サウンドエフェクトごとに具体的なIDを指定する必要があります。このIDでサウンドエフェクトのスタート、ストップ、音量等を設定することができます。aac、mp3、m4a形式をサポートしています。

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

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
effectId	int	サウンドエフェクト ID。
path	String	サウンドエフェクトパス。
count	int	ループ回数。
publish	boolean	プッシュかどうか / trueは視聴者に対するプッシュ、 falseはローカルプレビューです。
volume	int	音量の大きさ。数値の範囲は0 - 100、デフォルト値は100です。

pauseAudioEffect

サウンドエフェクトの再生を一時停止します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
effectId	int	サウンドエフェクト ID。

resumeAudioEffect

サウンドエフェクトの再生を再開します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
effectId	int	サウンドエフェクト ID。

stopAudioEffect

サウンドエフェクトの再生を停止します。

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

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
effectId	int	サウンドエフェクト ID。

stopAllAudioEffects

すべてのサウンドエフェクトの再生を停止します。

```
void stopAllAudioEffects();
```

setAudioEffectVolume

サウンドエフェクトの音量を設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
effectId	int	サウンドエフェクト ID。
volume	int	音量の大きさ。数値の範囲は0 - 100、デフォルト値は100です。

setAllAudioEffectsVolume

全てのサウンドエフェクトの音量を設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	音量の大きさ。数値の範囲は0 - 100、デフォルト値は100です。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCLiveRoom API (Flutter)

最終更新日：：2021-10-12 09:40:09

TRTCLiveRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- キャスターが新しいライブストリーミングルームを作成して配信を開始し、視聴者がライブストリーミングルームに参加して視聴します。
- キャスターと視聴者がビデオでマイク接続によるコラボライブを行います。
- 2つの異なるルームの間でキャスター同士のPK（クロスルーム対戦）を行います。
- 各種のテキストメッセージやカスタムメッセージの送信をサポートします。カスタムメッセージは弾幕、「いいね」、ギフトを実装するために使用することができます。

TRTCLiveRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実現プロセスは、 ビデオマイク接続ライブストリーミング（Flutter）をご参照ください。

- TRTC SDK： TRTC SDK を低遅延のライブストリーミングコンポーネントとして使用します。
- IM SDK： IM SDKのAVChatroomを使用して、ライブストリーミングチャットルーム機能を実装します。同時に、IMのメッセージでキャスター間のマイク接続のフローをつなげます。

TRTCLiveRoom API 概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを破棄します。
registerListener	イベントコールバックを設定します。
unRegisterListener	イベントコールバックのあるスレッドを設定します。
login	ログイン。
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
createRoom	ルームの新規作成（キャスターが呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの廃棄（キャスターが呼び出し）。
enterRoom	入室（視聴者が呼び出し）。
exitRoom	退室（視聴者が呼び出し）。
getRoomInfos	ルームリストの詳細情報を取得します。
getAnchorList	ルーム内の全キャスターのリストを取得します。enterRoom() 成功後に呼び出しが有効となります。
getRoomMemberList	ルーム内のすべてのメンバー情報を取得し、enterRoom() 成功後に呼び出しが有効となります。

プッシュプルストリームに関連するインターフェース関数

API	説明
startCameraPreview	ローカルビデオのプレビュー画面を立ち上げます。
stopCameraPreview	ローカルのビデオキャプチャおよびプレビューを停止します。
startPublish	ライブストリーミング（プッシュ）を開始します。
stopPublish	ライブストリーミング（プッシュ）を停止します。
startPlay	リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。
stopPlay	リモートのビデオ画面のレンダリングを停止します。

キャスターと視聴者のマイク接続

API	説明
requestJoinAnchor	視聴者がマイク接続をリクエストします。
responseJoinAnchor	キャスターがマイク接続のリクエストを処理します。
kickoutJoinAnchor	キャスターがマイク接続視聴者をキックアウトします。

©2013-2019 Tencent Cloud. All rights reserved.

キャスターのルーム間PK

API	説明
requestRoomPK	キャスターがルーム間PKをリクエストします。
responseRoomPK	キャスターがルーム間PKのリクエストに応答します。
quitRoomPK	ルーム間PKから退出します。

オーディオビデオ制御に関連するインターフェース関数

API	説明
switchCamera	フロント/リアカメラを切り替えます。
setMirror	ミラー表示のオン／オフを設定します。
muteLocalAudio	ローカルオーディオをミュートにします。
muteRemoteAudio	リモートオーディオをミュートにします。
muteAllRemoteAudio	全てのリモートオーディオをミュートにします。

BGMサウンドエフェクト関連インターフェース関数

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

美顔フィルタに関するインターフェース関数

API	説明
getBeautyManager	美顔管理オブジェクト TXBeautyManagerを取得します。

メッセージ送信関連インターフェース関数

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCLiveRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onKickedOffline	他のユーザーが同じアカウントでログインすると、キックアウトされてオフラインになります。

ルームイベントコールバック

API	説明
onEnterRoom	ローカルの入室のコールバック。
onUserVideoAvailable	リモートユーザーが再生できるビッグストリーム画面が存在するかどうか（一般的にはカメラに用いられる）。
onRoomDestroy	ルームが破棄されたときのコールバック。

キャスターと視聴者の参加／退出のイベントコールバック

API	説明
onAnchorEnter	新しいキャスターの入室通知を受信します。
onAnchorExit	キャスターの退室通知を受信します。
onAudienceEnter	視聴者入室通知の受信。
onAudienceExit	視聴者退室通知の受信。

キャスターと視聴者のマイク接続のイベントコールバック

API	説明
onRequestJoinAnchor	キャスターが視聴者のマイク接続リクエストを受信した時のコールバック。
onAnchorAccepted	キャスターが視聴者のマイク接続リクエストに同意します。
onAnchorRejected	キャスターが視聴者のマイク接続リクエストを拒否します。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onKickoutJoinAnchor	マイク接続の視聴者がマイク接続からキックアウトされた通知を受信します。

キャスター間PKのイベントコールバック

API	説明
onRequestRoomPK	ルーム間PKのリクエストの通知を受信。
onRoomPKAccepted	キャスターがルーム間PKのリクエストに同意します。
onRoomPKRejected	キャスターがルーム間PKのリクエストを拒否します。
onQuitRoomPK	ルーム間PK中止の通知を受信します。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

SDK基本関数

sharedInstance

TRTCLiveRoom シングルトンオブジェクトを取得します。

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

destroySharedInstance

TRTCLiveRoom シングルトンオブジェクトを破棄します。

> 説明：
> インスタンスを破棄後は、外部にキャッシュされたTRTCLiveRoomインスタンスの再使用ができません。
> 改めて sharedInstance を呼び出して、インスタンスを新規取得する必要があります。

©2013-2019 Tencent Cloud. All rights reserved.

```
static void destroySharedInstance()
```

registerListener

TRTCLiveRoom イベントコールバック。 TRTCLiveRoomDelegate を介してTRTCLiveRoomの各種ステータス通知を受け取ることができます。

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

> 説明：
> registerListenerは、TRTCLiveRoomのプロキシコールバックです。

unRegisterListener

コンポーネントイベントの監視インターフェースを削除します。

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

login

ログイン。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	TRTCコンソール > 【アプリケーション管理】 > アプリケーション情報でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法についてはUserSigの計算方法をご参照ください。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
config	TRTCLiveRoomConfig	グローバルコンフィギュレーション情報。ログイン時に初期化してください。ログイン後は変更できなくなります。 • useCDNFirst 属性：視聴者の視聴方式の設定に使用します。trueは、普通の視聴者のCDN経由での視聴を表し、費用は安価ですがレイテンシーは高めです。falseは、普通の視聴者の低レイテンシーによる視聴を表し、費用は CDN とマイク接続の間ですが、レイテンシーを1s以内に抑えることができます。 • CDNPlayDomain 属性： useCDNFirstの設定がtrueの時に有効となり、CDNでの視聴の再生ドメイン名の指定に使用します。CSSコンソール >【ドメイン名管理】の画面からログインし、設定を行います。

logout

ログアウト。

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

setSelfProfile

個人情報の修正。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像URL。

ルーム関連インターフェース関数

createRoom

ルームの新規作成（キャスターが呼び出し）。

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

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomId	int	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomID を、1つのライブストリーミングルームリストにまとめることができます。Tencent Cloudでは現在、ライブストリーミングルームリストの管理サービスを行っていませんので、ご自身でライブストリーミングルームリストを管理してください。
roomParam	RoomParam	ルーム情報は、ルーム名、カバー情報など、ルームを説明するための情報です。ルームリストおよびルーム情報をご自身のサーバーで管理する場合は、このパラメータは無視してもかまいません。

キャスターがブロードキャストを開始する際の通常の呼び出しプロセスは次のとおりです。

1. 【キャスター】 `startCameraPreview()` を呼び出し、カメラのプレビューを起動します。この時美顔パラメータを調整できます。
2. 【キャスター】 `createRoom()` を呼び出し、ライブストリーミングルームを作成します。ルーム作成が成功したかどうかがActionCallbackでキャスターに通知されます。
3. 【キャスター】 `starPublish()` を呼び出し、プッシュを開始します。

destroyRoom

ルームの破棄（キャスターが呼び出し）。キャスターは、ルーム作成後、この関数を呼び出してルームを破棄します。

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

enterRoom

入室（視聴者が呼び出し）。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームID。

視聴者のライブストリーミング視聴の通常の呼び出しフローは以下のとおりです。

1.【視聴者】サーバーから取得する最新のライブストリーミングルームリストには、多くのライブストリーミングルームのroomIdおよびルーム情報が含まれる場合があります。

©2013-2019 Tencent Cloud. All rights reserved.

2. 【視聴者】視聴者は1つのライブストリーミングルームを選択し、 `enterRoom()` を呼び出して、このルームに参加します。
3. 【視聴者】 `startPlay(userId)` を呼び出し、キャスターのuserIdを渡して再生を開始します。

- ライブストリーミングルームリストにキャスターのuserId情報がすでに含まれている場合、視聴者は、 `startPlay(userID)` を直接呼び出せば、再生を開始できます。
- 入室前にキャスターのuserIdが一時的に取得できない場合、視聴者は入室後に `TRTCLiveRoomDelegate` から `onAnchorEnter(userId)` のイベントコールバックを受信します。このコールバックの中にキャスターのuserId情報が含まれていますので、 `startPlay(userId)` を再び呼び出せば、再生できます。

exitRoom

ルームを退出します。

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

getRoomInfos

ルームリストの詳細情報を取得します。ルーム情報は、キャスターが `createRoom()` 作成時にroomInfoによって設定します。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIdList	List&dxlt;String&dxgt;	ルームナンバーリスト。

getAnchorList

ルーム内の全キャスターのリストを取得します。 `enterRoom()` 成功後に呼び出しが有効となります。

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

getRoomMemberList

©2013-2019 Tencent Cloud. All rights reserved.

ルーム内の全視聴者の情報を取得します。 `enterRoom()` 成功後に呼び出しが有効となります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
nextSeq	int	ページネーションプルフラグ。最初のプルに0と入力すると、コールバックが成功します。nextSeqがゼロでない場合は、ページネーションが必要であり、0になるまで渡して再度プルします。

プッシュプルストリームに関連するインターフェース関数

startCameraPreview

ローカルビデオのプレビュー画面を立ち上げます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFrontCamera	bool	true：フロントカメラ；false：リアカメラ。
viewId	int	ビデオviewのコールバック ID。

stopCameraPreview

ローカルのビデオキャプチャおよびプレビューを停止します。

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

startPublish

ライブストリーミング（プッシュ）を開始します。次のシーンに適用します。

- キャスターの放送開始時の呼び出し。
- 視聴者のマイク接続開始時の呼び出し。

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

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
streamId	String?	ライブストリーミングのCDNのstreamIdをバインドするために利用します。視聴者にライブCDN経由でライブストリーミングを視聴させたい場合は、現在のキャスターのライブストリーミングstreamIdを指定する必要があります。

stopPublish

ライブストリーミング（プッシュ）を停止します。次のシーンに適用します。

- キャスターがライブストリーミングを終了する時の呼び出し。
- 視聴者がマイク接続を終了する時の呼び出し。

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

startPlay

リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	視聴が必要なユーザーid。
viewId	int	ビデオviewのコールバックid。

- **普通の視聴シーン：**
  - ライブストリーミングルームリストにキャスターのuserId情報がすでに含まれている場合、視聴者は、 `enterRoom()` 成功後に `startPlay(userId)` を直接呼び出せばキャスターの画面を再生できます。
  - 入室前にキャスターのuserIdが一時的に取得できない場合、視聴者は入室後に `TRTCLiveRoomDelegate` の `onAnchorEnter(userId)` のイベントコールバックを受信します。このコールバックの中にキャスターのuserId 情報が含まれていますので、 `startPlay(userId)` を再び呼び出せば、キャスターの画面を再生できます。

- **ライブストリーミングのマイク接続シーン：**
  マイク接続の始動後、キャスターは `TRTCLiveRoomDelegate` から `onAnchorEnter(userId)` のコールバックを受

©2013-2019 Tencent Cloud. All rights reserved.

信します。この時コールバックの中のuserIdを使用して `startPlay(userId)` を呼び出せば、マイク接続の画面を再生できます。

stopPlay

リモートのビデオ画面のレンダリングを停止します。 `onAnchorExit()` のコールバック時、このインターフェースを呼び出す必要があります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	相手側のユーザー情報。

キャスターと視聴者のマイク接続

requestJoinAnchor

視聴者がマイク接続をリクエストします。

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

キャスターと視聴者のマイク接続フローは次のとおりです。

1. 【視聴者】 `requestJoinAnchor()` を呼び出し、キャスターにマイク接続リクエストを送信します。
2. 【キャスター】 `TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバック通知を受信します。
3. 【キャスター】 `responseJoinAnchor()` を呼び出して、視聴者からのマイク接続リクエストを受け入れるか決定します。
4. 【視聴者】responseCallbackのコールバック通知を受信します。この通知にキャスターの処理結果が含まれています。
5. 【視聴者】リクエストが同意されると、 `startCameraPreview()` が呼び出され、ローカルカメラが起動します。
6. 【視聴者】その後 `startPublish()` を呼び出し、正式にプッシュストリームの状態に入ります。
7. 【キャスター】一度視聴者がマイク接続に入ると、キャスターは、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を受信します。
8. 【キャスター】キャスターが `startPlay()` を呼び出すと、マイク接続の視聴者のビデオ画面を見ることができるようになります。
9. 【視聴者】ライブストリーミングルームの中で他の視聴者がすでにキャスターとマイク接続を行っている場合、新しく参加したマイク接続の視聴者は `onAnchorEnter()` の通知を受け取ります。この時

©2013-2019 Tencent Cloud. All rights reserved.

に `startPlay()` を呼び出せば他のマイク接続者のビデオ画面を再生することができます。

responseJoinAnchor

キャスターがマイク接続のリクエストを処理します。キャスターは、`TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバックを受け取った後、このインターフェースを呼び出して、視聴者のマイク接続リクエストを処理する必要があります。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	視聴者ID。
agree	bool	true：同意；false：拒否。

kickoutJoinAnchor

キャスターがマイク接続の視聴者をキックアウトします。キャスターがこのインターフェースを呼び出し、マイク接続の視聴者をキックアウトすると、キックアウトされたマイク接続の視聴者は、`TRTCLiveRoomDelegate` の `onKickoutJoinAnchor()` のコールバック通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	マイク接続の視聴者 ID。

キャスターのルーム間PK

requestRoomPK

キャスターがルーム間PKをリクエストします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomId	int	招待された側のルーム ID。
userId	String	招待された側のキャスター ID。

キャスターとキャスターの間でルーム間PKを行うことができます。ライブストリーミング中の2名のキャスターAとBのルーム間PKのフローは次のとおりです。

1. 【キャスター A】 `requestRoomPK()` を呼び出してキャスターBに向けてマイク接続リクエストを送信します。
2. 【キャスター B】 `TRTCLiveRoomDelegate` の `onRequestRoomPK()` のコールバック通知を受信します。
3. 【キャスター B】 `responseRoomPK()` を呼び出して、キャスター AのPKのリクエストを受け入れるかどうか決定します。
4. 【キャスター B】キャスター Aのリクエストを受け入れる場合は、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出してキャスターAのビデオ画面を表示させます 。
5. 【キャスター A】 `onRoomPKAccepted` または `onRoomPKRejected` のコールバック通知を受信します。
6. 【キャスターA】リクエストが同意された場合は、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出し、キャスターBのビデオ画面を表示します。

responseRoomPK

キャスターがルーム間PKのリクエストに応答します。キャスターが応答した後、相手側キャスターは `requestRoomPK` で渡された `responseCallback` のコールバックを受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	PKをリクエストしたキャスターID。
agree	bool	true：同意；false：拒否。

quitRoomPK

ルーム間PKから退出します。PK中のいずれかのキャスターがルーム間PKの状態から退出すると、もう一方のキャスターは `TRTCLiveRoomDelegate` の `onQuitRoomPk()` のコールバック通知を受信します。

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

©2013-2019 Tencent Cloud. All rights reserved.

オーディオビデオ制御に関連するインターフェース関数

switchCamera

フロント/リアカメラを切り替えます。

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

setMirror

ミラー表示のオン／オフを設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMirror	bool	ミラーオン/オフ。

muteLocalAudio

ローカルオーディオをミュートにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動；false：ミュート停止。

muteRemoteAudio

リモートオーディオをミュートにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	リモートユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動；false：ミュート停止。

muteAllRemoteAudio

すべてのリモート側のオーディオをミュートにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動；false：ミュート停止。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

BGMサウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

```
getAudioEffectManager();
```

美顔フィルタに関するインターフェース関数

getBeautyManager

美顔管理オブジェクトTXBeautyManagerを取得します。

```
getBeautyManager();
```

美顔管理では、次の機能を使用できます。

- 「美顔のスタイル」、「美白」、「肌色補正（血色・つや感）」、「デカ眼」、「顔痩せ」、「V顔」、「下あご」、「面長補正」、「小鼻」、「キラキラ目」、「白い歯」、「目の弛み除去」、「シワ除去」、「ほうれい線除去」などの美容効果を設定します。
- 「髪の生え際」、「眼と眼の距離」、「眼角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
- 人の顔のスタンプ（素材）等のダイナミック効果を設定します。

©2013-2019 Tencent Cloud. All rights reserved.

- メイクアップを追加します。
- ジェスチャー認識を行います。

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。

TRTCLiveRoomDelegateイベントのコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

説明：
SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
errCode	int	エラーコード。
errMsg	String	エラー情報。

onWarning

警告のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
warningCode	int	エラーコード。
warningMsg	String	警告情報。

onKickedOffline

他のユーザーが同じアカウントでログインすると、キックアウトされてオフラインになります。

ルームイベントのコールバック

onRoomDestroy

ルームが破棄された時のコールバック。キャスターが退室する時、ルーム内の全ユーザーがこの通知を受信します。

onEnterRoom

ローカル入室

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
result	int	result &dxgt; 0 の場合は入室に費やした時間（ms）、result &dxlt; 0 の場合は入室エラーコード。

onUserVideoAvailable

リモートユーザーが再生できるビッグストリーム画面が存在するかどうか（一般的にはカメラに用いられる）。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
available	boolean	画面の起動の有無。

キャスターと視聴者の参加／退出のイベントコールバック

onAnchorEnter

新しいキャスターの入室の通知を受信します。マイク接続の視聴者とルーム間PKのキャスターが入室すると、視聴者は、新しいキャスターが入室したというイベントを受信します。 `TRTCLiveRoom` の `startPlay()` を呼び出して、そのキャスターのビデオ画面を表示することができます。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	新しく入室したキャスターの ID。
userName	String	ユーザーのニックネーム。
userAvatar	String	ユーザーのプロフィール画像のアドレス。

onAnchorExit

キャスターの退室の通知を受信します。ルーム内のキャスター（およびマイク接続中の視聴者）が、新しいキャスターの退室のイベントを受信します。 `TRTCLiveRoom` の `stopPlay()` を呼び出して、そのキャスターのビデオ画面を閉じることができます。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	String	退出するキャスターの ID。
userName	String	ユーザーのニックネーム。
userAvatar	String	ユーザーのプロフィール画像のアドレス。

onAudienceEnter

視聴者入室通知の受信。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	TRTCLiveRoomDef.TRTCLiveUserInfo	入室する視聴者のユーザーID、ニックネーム、プロフィール画像などの情報。

onAudienceExit

視聴者退室通知の受信。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	退室した視聴者の情報。
userName	String	ユーザーのニックネーム。
userAvatar	String	ユーザーのプロフィール画像のアドレス。

onRequestJoinAnchor

キャスターが視聴者のマイク接続リクエストを受信した時のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	マイク接続をリクエストするユーザーID。
userName	String	ユーザーのニックネーム。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userAvatar	String	ユーザーのプロフィール画像のアドレス。

onAnchorAccepted

キャスターが視聴者のマイク接続リクエストに同意します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	キャスターのユーザーID。

onAnchorRejected

キャスターが視聴者のマイク接続リクエストを拒否します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	キャスターのユーザーID。

onKickoutJoinAnchor

マイク接続の視聴者がマイク接続からキックアウトされた通知を受信します。マイク接続の視聴者は、キャスターからマイク接続キックアウトのメッセージを受信した場合、 `TRTCLiveRoom` の `stopPublish()` を呼び出してマイク接続から退出する必要があります。

キャスター間PKのイベントコールバック

onRequestRoomPK

ルーム間PKのリクエストの通知を受信します。キャスターは、他のルームのキャスターからPKのリクエストを受け取り、PKに同意する場合、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を待ってから、 `startPlay()` を呼び出して招待側キャスターのストリームを再生する必要があります。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ルーム間をリクエストするユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userName	String	ユーザーのニックネーム。
userAvatar	String	ユーザーのプロフィール画像のアドレス。

onRoomPKAccepted

キャスターはルーム間PKのリクエストに同意します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ルーム間PKに同意するユーザーID。

onRoomPKRejected

キャスターはルーム間PKのリクエストに同意します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ルーム間PKを拒否するユーザーID。

onQuitRoomPK

ルーム間PK中止の通知を受信します。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。

onRecvRoomCustomMsg

カスタムメッセージの受信。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
command	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。

©2013-2019 Tencent Cloud. All rights reserved.

音声チャットルーム
ボイスチャットルーム(iOS)

最終更新日：：2022-03-09 17:59:26

デモンストレーション

ダウンロードからAppをインストールして、マイク管理、低レイテンシー音声インタラクション、文字チャットなどのボイスチャットのユースケースにおけるTRTCの関連機能を含む、ボイスチャットルームの機能を体験できます。

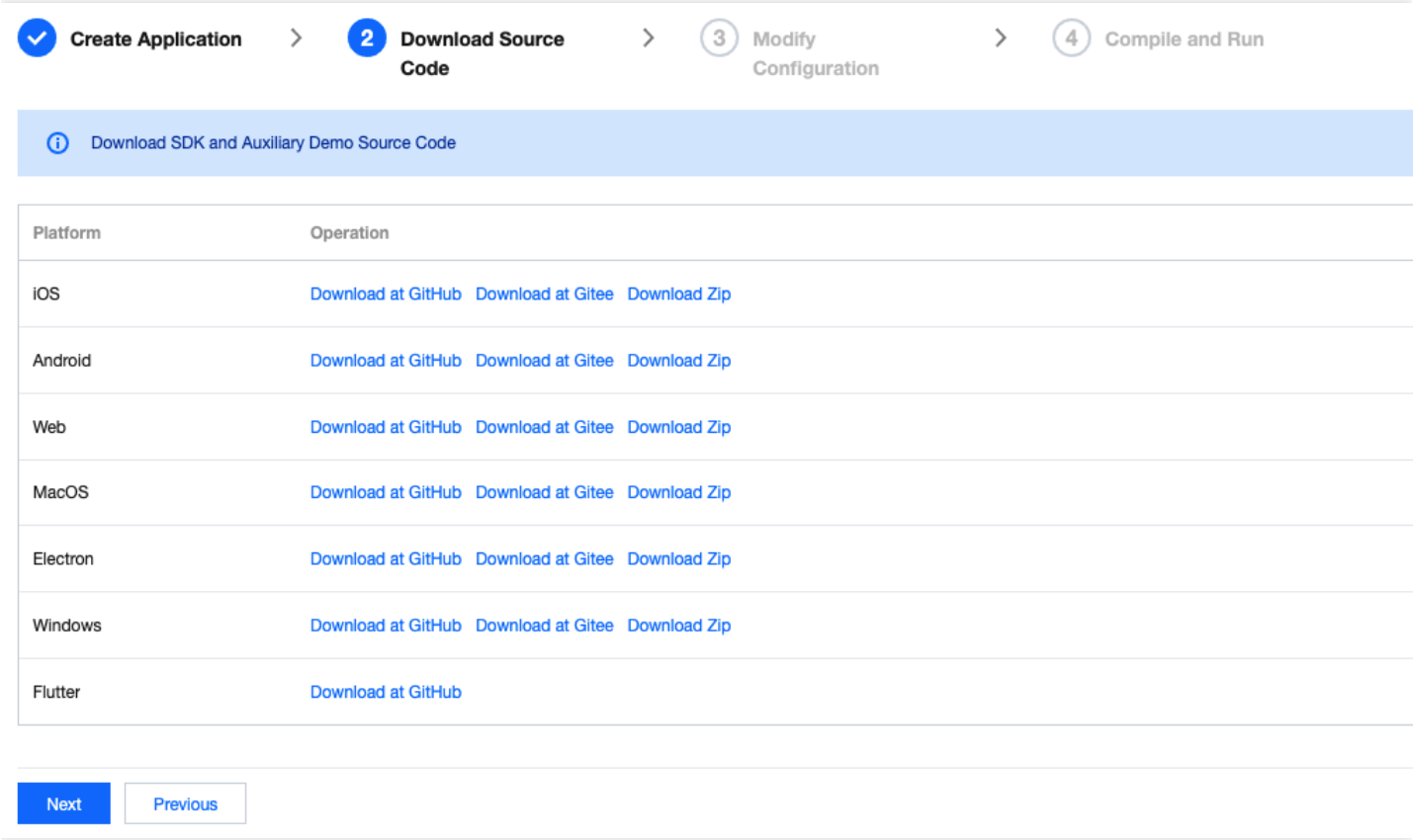
ボイスチャットルーム機能をすばやく実装する必要がある場合は、当社が提供するAppを基に直接変更を加えて適合させることも、当社が提供するTUIVoiceRoomコンポーネントでカスタマイズしたUIを実装することも可能です。

AppのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、 **開発支援** > **Demoクイックスタート**を選択します。
2. `TestLiveRoom` などのアプリケーション名を入力して、 **作成 **をクリックします。
3. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。

©2013-2019 Tencent Cloud. All rights reserved.



説明：
本機能はTencent Cloud TRTC と IM という2つの基本的な PaaS サービスを同時に使用し、TRTCをアクティブにした後、IM サービスを同期的にアクティブにすることができます。 IM は付加価値サービスであり、課金ルールの詳細については Instant Messagingの価格説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUIVoiceRoomに進み、ソースコードをCloneまたはダウンロードします。

手順3：Appプロジェクトファイルの設定

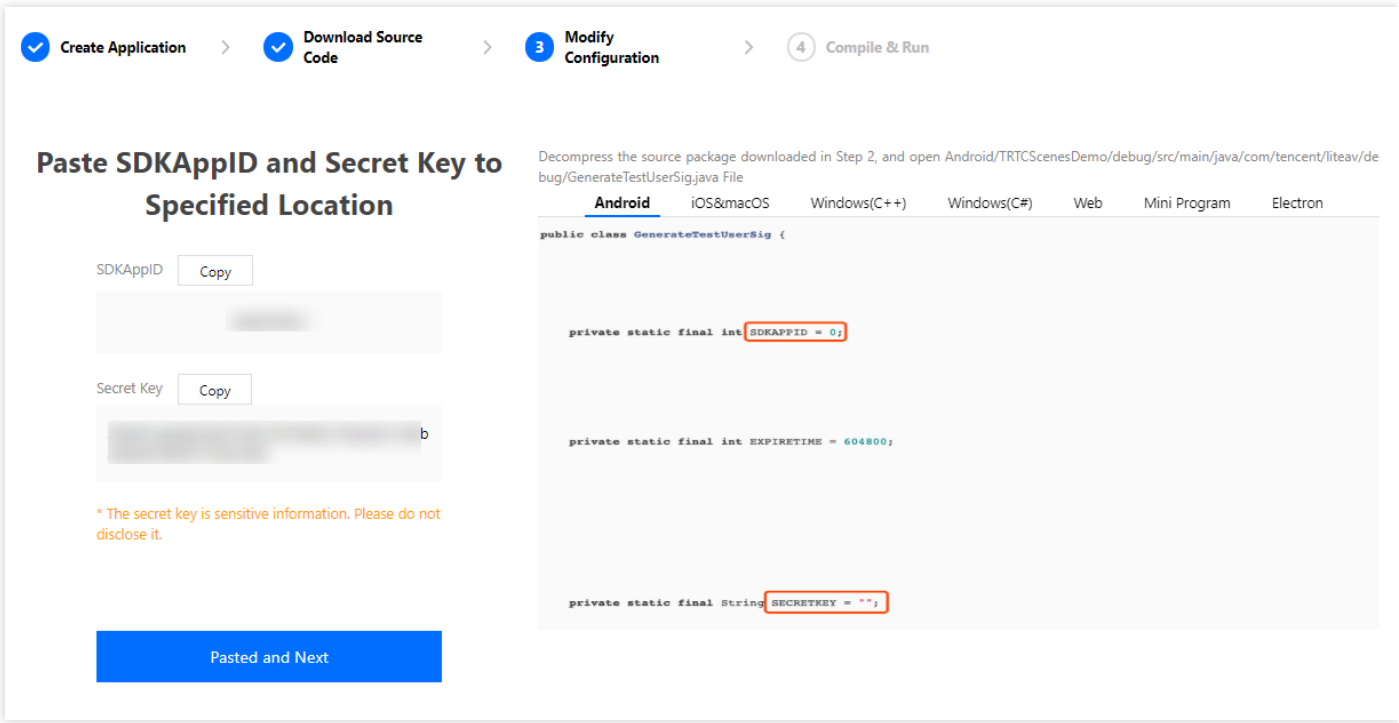
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `TUIVoiceRoom/Debug/GenerateTestUserSig.swift` ファイルを見つけて開きます。

3. `GenerateTestUserSig.swift` ファイル内の関連パラメータを設定します。

   - SDKAPPID：デフォルトは0。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトは空文字列。実際のキー情報を設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Xcode（11.0以上のバージョン）を使用してソースコードプロセス `TUIVoiceRoom/TUIVoiceRoomApp.xcworkspace` を開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

手順5：Appソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードの `Source` フォルダは、uiとmodelの2つのサブフォルダを含んでいます。uiフォルダには、インターフェースコードとインターフェース関連のロジックが含まれています。以下の表にはそれぞれのswiftファイルまたはフォルダおよびそれらが対応するUIをリストアップして、二次調整を行いやすくしています。

ファイルまたはフォルダ	機能の説明
TRTCVoiceRoomEnteryControl.swift	このフォルダにはすべてのViewControllerの初期化取得方法が含まれています。このインスタンスを介してViewControllerオブジェクトをすばやく取得することができます。
TRTCCreateVoiceRoomViewController	ボイスチャットルームページのロジック作成。
TRTCVoiceRoomViewController	メインルームページ。管理者と視聴者という2種類のインターフェースがあります。

各 `TRTC'XXXX'ViewController` フォルダには、 `ViewController` 、 `RootView` および `ViewModel` が含まれ、各ファイルの役割は下表に示すとおりです。

ファイル	機能説明
ViewController.swift	ページコントローラ。画面のルーティング作業、RootViewおよびViweModelのバインディング作業を担当します。
RootView.swift	ビュー。すべてのビューのレイアウト。
ViewModel.swift	ビューコントローラ。ビューのインタラクションに応答し、ビューの応答ステータスを返す作業を担当します。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. **ルームの作成**をクリックします。
3. ルームトピックを入力し、**チャット開始**をクリックします。

©2013-2019 Tencent Cloud. All rights reserved.

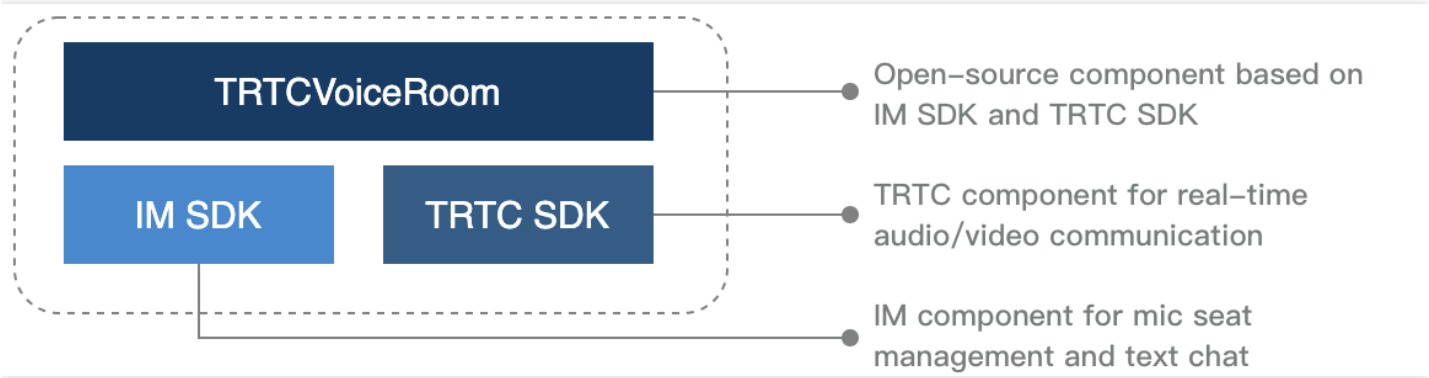
ユーザーB

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ユーザーAが作成したルーム番号を入力し、**入室する**をクリックします。

> 注意：
> ルームナンバーはユーザーAのルーム上部に表示されます。

UIカスタマイズの実装

ソースコードの `Source` フォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTRTCChorusRoomが含まれています。 `TRTCChorusRoom.h` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタマイズしたUIを実装することができます。



手順1：SDKの統合

ボイスチャットコンポーネントTRTCVoiceRoom は、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

- **方法1：cocoapodsリポジトリを介する依存**

```
pod 'TXIMSDK_iOS'
pod 'TXLiteAVSDK_TRTC'
```

- **方法2：ローカルを介する依存**
  開発環境でのcocoapodsリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合

©2013-2019 Tencent Cloud. All rights reserved.

ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定

info.plistファイルに `Privacy > Camera Usage Description` 、 `Privacy > Microphone Usage Description` を追加し、マイクの権限を申請する必要があります。

手順3：TUIVoiceRoomコンポーネントをインポート

cocoapodsを介してコンポーネントをインポートします。具体的な操作は次のとおりです。

1. プロジェクトディレクトリ下の `Source` 、 `Resources` 、 `TXAppBasic` フォルダ、 `TUIVoiceRoom.podspec` ファイルをプロジェクトディレクトリ下へコピーします。
2. `Podfile` ファイル内に以下の依存関係を追加します。その後、 `pod install` コマンドを実行すると、インポートが完了します。

```
pod 'TXAppBasic', :path =&gt; "TXAppBasic/"
pod 'TXLiteAVSDK_TRTC'
pod 'TUIVoiceRoom', :path =&gt; "./", :subspecs =&gt; ["TRTC"]
```

手順4：コンポーネントの作成およびログイン

1. TRTCVoiceRoomの `sharedInstance` クラスメソッドを呼び出せば、TRTCVoiceRoomプロトコルを遵守するインスタンスオブジェクトを作成できます。 `shared` クラスメソッドを呼び出しTRTCVoiceRoomImpインスタンスオブジェクトを取得して直接使用することもできます。両者は、TRTCVoiceRoomのインターフェースでの使い方に違いはありません。
2. `setDelegate` 関数を呼び出して、コンポーネントのイベントコールバック通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	機能
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、

©2013-2019 Tencent Cloud. All rights reserved.

	UserSigの計算方法をご参照ください。
callback	ログインのコールバック。成功時にcodeは0になります。

```
// Swift例
// コード内でビジネスロジックを担当するクラス
class YourController {
// 属性を計算してシングルトンオブジェクトを取得
var voiceRoom: TRTCVoiceRoomImp {
return TRTCVoiceRoomImp.shared()
}

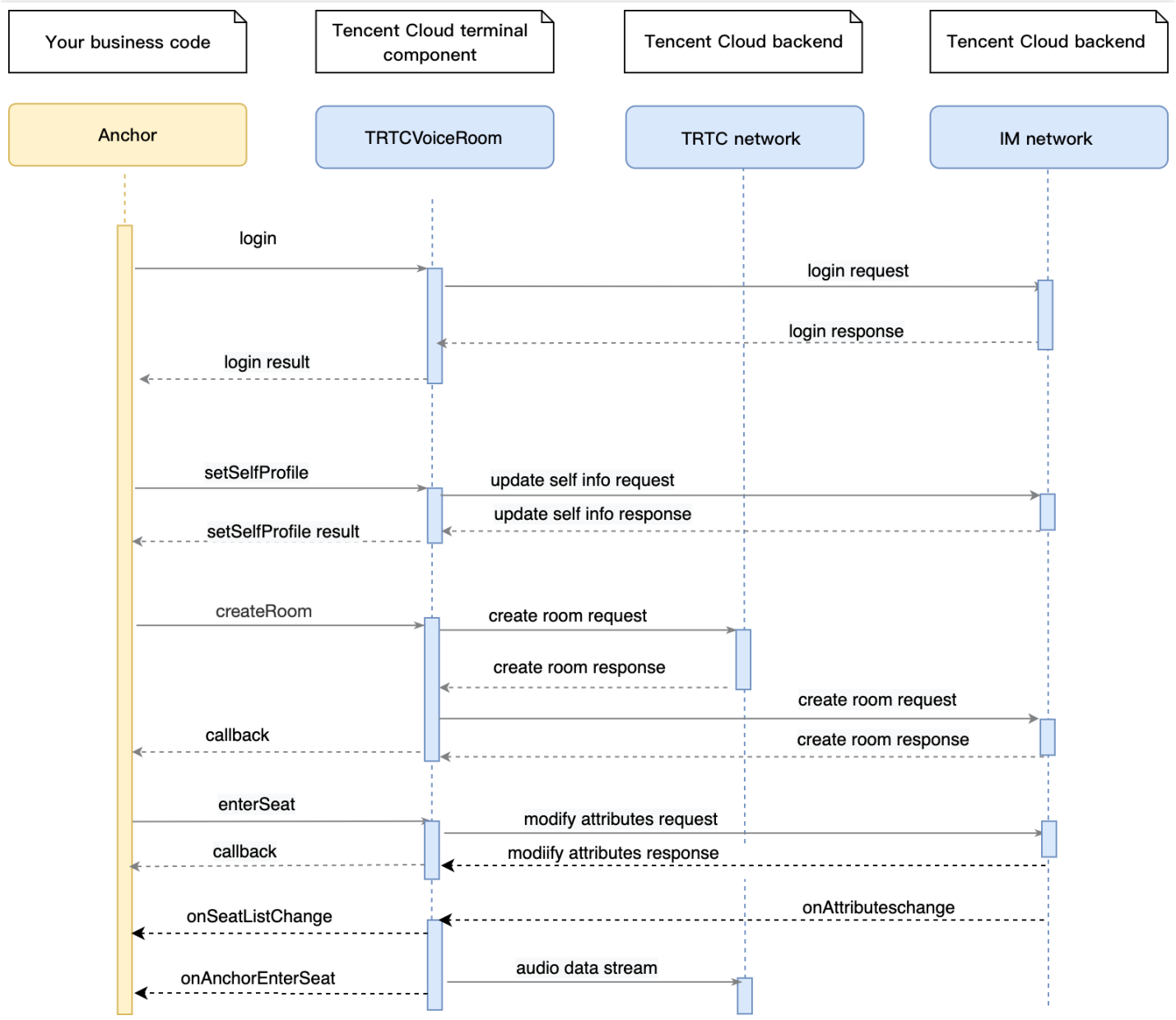
// その他のコードロジック
......
}
// voiceroomプロキシの設定
self.vocieRoom. setDelegate(delegate: voiceRoomDelegate)

// コール方法は以下のとおり。クロージャーの中には weak self を使用して循環引用にしないことをお勧めします（以下の例示コードでは、 weak self 例を省略）
self.vocieRoom.login(sdkAppId: sdkAppID, userId: userId, userSig: userSig) { [weak self] (code, message) in
guard let `self` = self else { return }
// コールバックビジネスロジック
}
```

手順5：管理者側での配信開始

1. 管理者は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 管理者は、 `createRoom` を呼び出して新しいボイスチャットルームを作成します。この時にルームID、マイク・オンに管理者の確認の要否、マイク数などルームのプロパティ情報を渡します。
3. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
4. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
5. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

©2013-2019 Tencent Cloud. All rights reserved.



サンプルコード：

```
// 1.管理者は、ニックネームおよびプロフィール画像を設定します
self.voiceRoom.setSelfProfile(userName: userName, avatarUrl: avatarURL) { (code, message) in
// 結果のコールバック
}

// 2.管理者側でルームを作成します
let param = VoiceRoomParam.init()
Param.roomName = "ルーム名"
param.needRequest = true // リスナーのマイク・オンに対する管理者の同意の要否
param.coverUrl = 「カバーURL」
param.seatCount = 7 // ルームの座席数。ここでは計7席あり、管理者が1席を占め、残り6席がリスナーと
```

©2013-2019 Tencent Cloud. All rights reserved.

```
なります
param.seatInfoList = []

for _ in 0..&lt; param.seatCount {
let seatInfo = VoiceRoomSeatInfo.init()
param.seatInfoList.append(seatInfo)
}

self.voiceRoom.createRoom(roomID: yourRoomID, roomParam: param) { (code, message) in
guard code == 0 else { reutrn }
// ルーム作成成功後、席の占有を開始
self.voiceRoom.enterSeat(seatIndex: 0) { [weak self] (code, message) in
guard let `self` = self else { return }
if code == 0 {
// 管理者による席の占有成功
}else{
// 管理者による席の占有失敗
}
}
}

// 3.席の占有に成功した後、onSeatListChangeイベント通知を受信します
func onSeatListChange(seatInfoList: [VoiceRoomSeatInfo]) {
// マイクリストの更新
}

// 4. onAnchorEnterSeatイベント通知を受信します
func onAnchorEnterSeat(index: Int, user: VoiceRoomUserInfo) {
// 管理者のマイク・オンイベントの処理
}
```

手順6：リスナー側の視聴

1. リスナー側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. リスナー側は、業務バックエンドに最新のボイスチャットルームのルームリストを取得します。

> 説明：
> Appのボイスチャットルームリストは、デモンストレーション用のためだけのものです。ボイスチャットルームリストのビジネスロジックは様々です。Tencent Cloudは現在、ボイスチャットルームリストの管理サービスを提供していませんので、ご自身でボイスチャットのルームリストを管理してください。

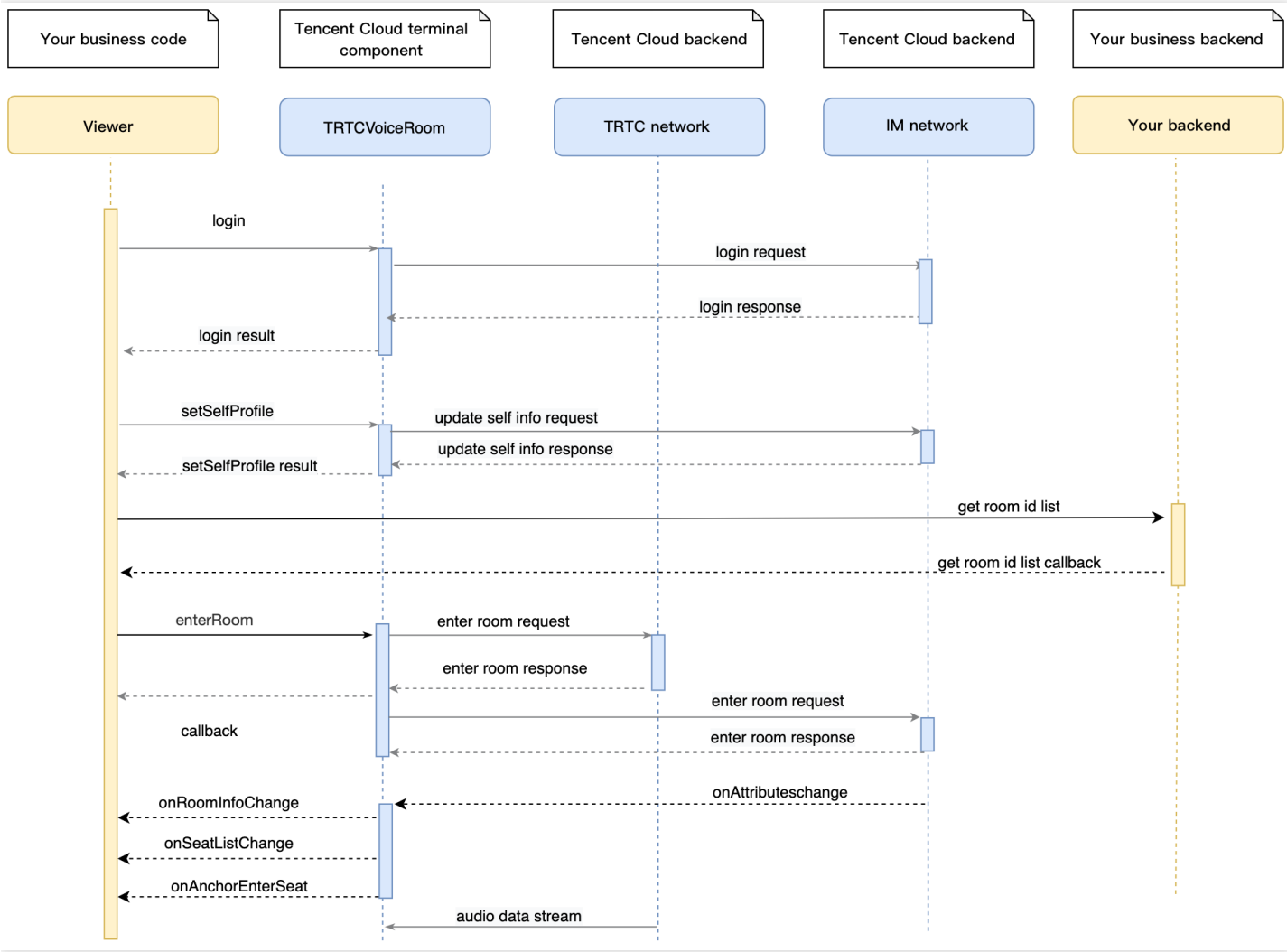
©2013-2019 Tencent Cloud. All rights reserved.

3. リスナー側は、 `getRoomInfoList` を呼び出してルームの詳細情報を取得します。この情報は、管理者側が `createRoom` を呼び出してボイスチャットルームを作成するときに設定する簡単な説明情報です。

> 注意：
> ボイスチャットルームのリストに十分に包括的な情報がある場合は、 `getRoomInfoList` の呼び出しに関する手順をスキップできます。
> 4. リスナーは1つのボイスチャットルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。

4. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
5. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
6. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` というイベント通知も受信します。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.リスナーは、ニックネームおよびプロフィール画像を設定します
self.voiceRoom.setSelfProfile(userName: userName, avatarUrl: avatarURL) { (code, message) in
// 結果のコールバック
}

// 2.業務バックエンドから取得したルームリストをroomListと仮定します
let roomList: [Int] = getRoomIDList() // ルームIDリストを取得する関数

// 3.getRoomInfoListを呼び出すことによって、ルームの詳細情報を取得します
self.voiceRoom.getRoomInfoList(roomIdList: roomIdsInt) { (code, message, roomInfos: [VoiceRoomInf
o]) in
//結果の取得。この時、UIの更新が可能
}

// 4.ボイスチャットルームを選択後、roomIdを渡して入室します
self.voiceRoom.enterRoom(roomID: roomInfo.roomID) { (code, message) in
// 入室結果コールバック
if code == 0 {
```

```
// 入室に成功
}
}

// 5.入室に成功後、onRoomInfoChangeイベント通知を受信します
func onRoomInfoChange(roomInfo: VoiceRoomInfo) {
// ルーム名などの情報の更新可
}

// 6.入室に成功後、onSeatListChangeイベント通知を受信します
func onSeatListChange(seatInfoList: [VoiceRoomSeatInfo]) {
// マイクリストを更新
}

// 7. onAnchorEnterSeatイベント通知を受信します
func onAnchorEnterSeat(index: Int, user: VoiceRoomUserInfo) {
// マイク・オンイベントの処理
}
```

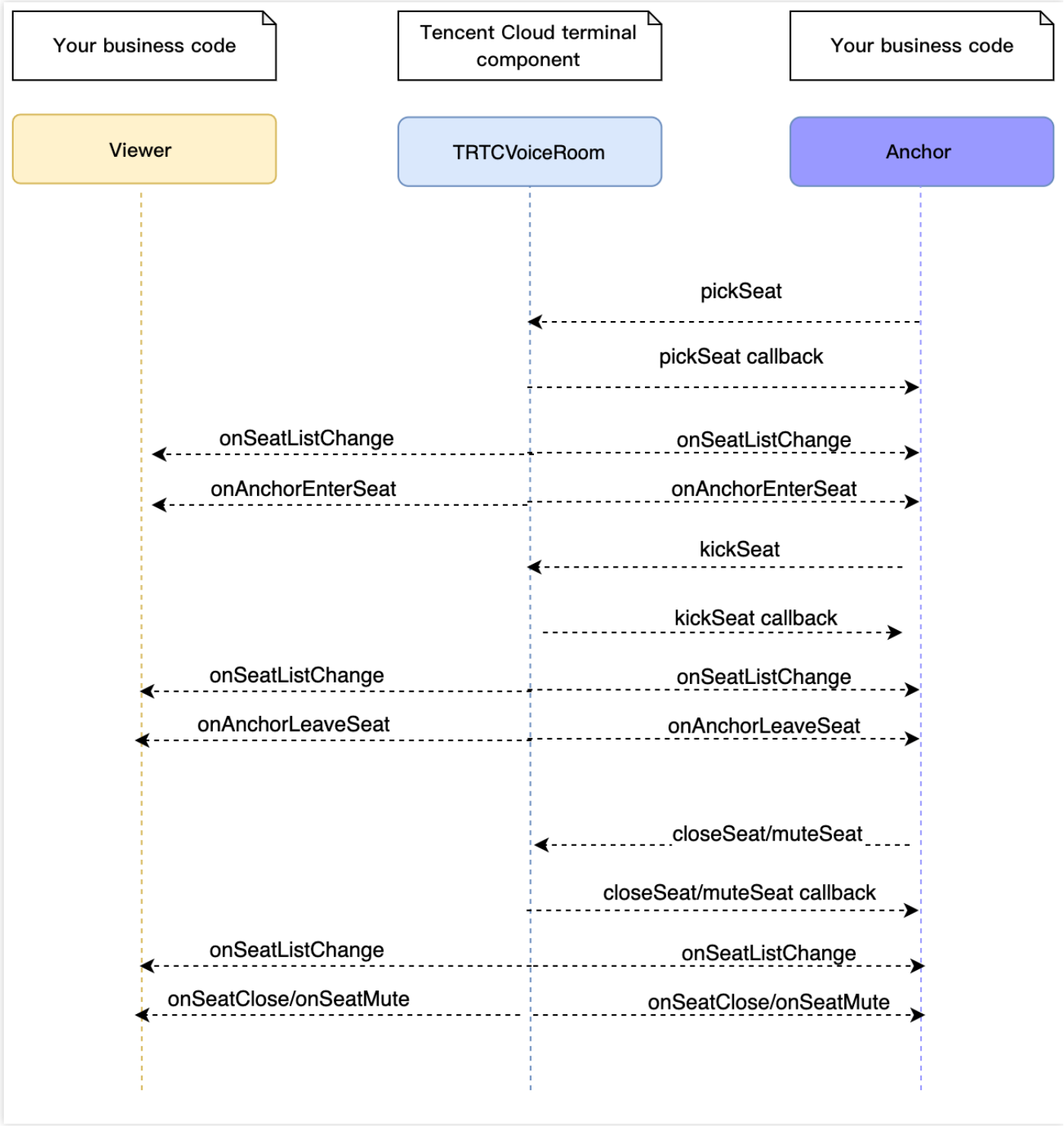
手順7：マイク管理

- 管理者側
- リスナー側

1. `pickSeat` は、対応するマイクおよびリスナーのuserIdを渡し、視聴者を発言できるように招待できます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。
2. `kickSeat` は、対象となるマイクを渡した後、キックアウトしてマイク・オフにすることができます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。
3. `muteSeat` は、対応するマイクを渡した後、ミュート/ミュート解除をすることができます。ルーム内の全参加者は `onSeatListChange` および `onSeatMute` というイベント通知を受信します。
4. `closeSeat` は、対応するマイクを送信後、任意のマイクのクローズ/解除をすることができます。クローズ後は、リスナー側はこれ以上マイクを使用することはできません。ルーム内の全参加者

©2013-2019 Tencent Cloud. All rights reserved.

は `onSeatListChange` および `onSeatClose` というイベント通知を受信します。



手順8：招待シグナリングの使用

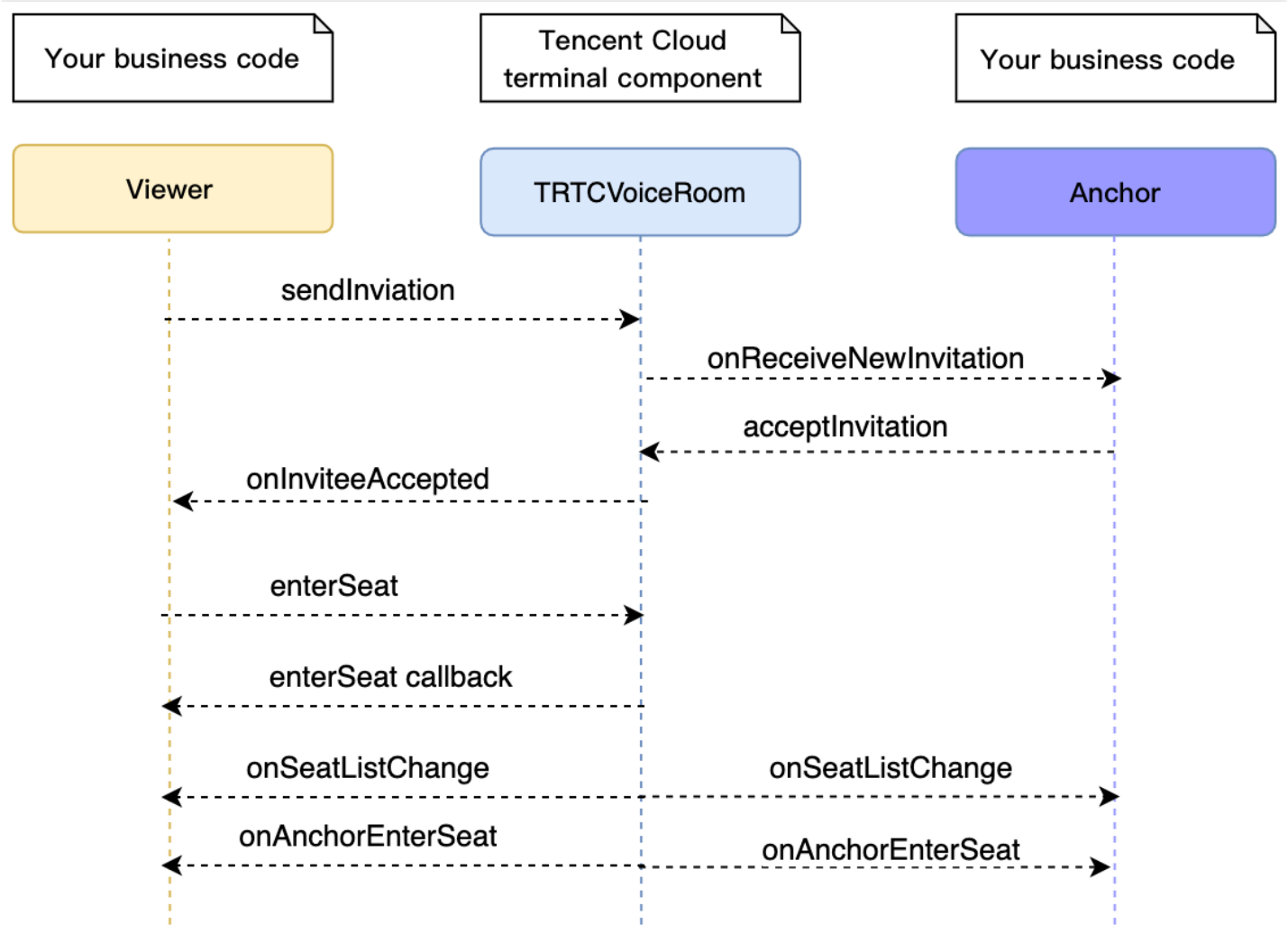
マイク管理では、リスナーがマイク・オン/オフにする場合や、管理者が視聴者を発言できるように招待する場合は、相手の同意を必要とせずに直接操作することができます。

©2013-2019 Tencent Cloud. All rights reserved.

Appが次の操作の業務フローを実施するために、相手の同意を必要とする場合、招待シグナリングによって対応するサポートを提供することができます。

- リスナーからの自主的なマイク・オン申請
- 管理者からの招待によるリスナーのマイク・オン

1. リスナー側が `sendInvitation` を呼び出し、管理者のuserIdおよび業務のカスタムコマンドワードなどを渡します。この時、関数が1つのinviteIdを返しますので、このinviteIdを記録します。
2. 管理者側は、 `onReceiveNewInvitation` というイベント通知を受信します。この時UIでウィンドウをポップアップさせ、管理者に同意の有無をたずねることができます。
3. 管理者が同意を選択後、 `acceptInvitation` を呼び出してinviteIdを渡します。
4. リスナー側は、 `onInviteeAccepted` というイベント通知を受信し、 `enterSeat` を呼び出してマイク・オンにします。



©2013-2019 Tencent Cloud. All rights reserved.

```
// リスナー側の視点
// 1. sendInvitationを呼び出し、1号マイクの使用をリクエストします
let inviteId = self.voiceRoom.sendInvitation(cmd: "ENTER_SEAT", userId: ownerUserId, content: "1"
) { (code, message) in
// 送信結果のコールバック
}
// 2.招待のリクエスト同意を受信し、正式にマイク・オンになります
func onInviteeAccepted(identifier: String, invitee: String) {
if identifier == selfID {
self.voiceRoom.enterSeat(seatIndex: ) { (code, message) in
// マイク・オン結果のコールバック
}
}
}

// 管理者側の視点
// 1.管理者がリクエストを受信します
func onReceiveNewInvitation(identifier: String, inviter: String, cmd: String, content: String) {
if cmd == "ENTER_SEAT" {
// 2.管理者がリスナーのリクエストに同意します
self.voiceRoom.acceptInvitation(identifier: identifier, callback: nil)
}
}
```

手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、このルーム内のすべてのキャスターおよびリスナーが `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

  ```
  // 送信側：テキストメッセージの送信
  self.voiceRoom.sendRoomTextMsg(message: message) { (code, message) in


  }
  // 受信側：テキストメッセージのモニタリング
  func onRecvRoomTextMsg(message: String, userInfo: VoiceRoomUserInfo) {
  // 受信したmessage情報の処理方法
  }
  ```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターとリスナーが `onRecvRoomCustomMsg` コールバックを受信できます。
  カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

©2013-2019 Tencent Cloud. All rights reserved.

```
// 例：送信側：カスタマイズCmdによって、弾幕と「いいね」情報を区分することができます
// eg:「CMD_DANMU」は弾幕コメントを表し、「CMD_LIKE」は「いいね」情報を表します
self.vocieRoom.sendRoomCustomMsg(cmd: "CMD_DANMU", message: "hello world", callback: nil)
self.voiceRoom.sendRoomCustomMsg(cmd: "CMD_LIKE", message: "", callback: nil)
// 受信側：カスタムメッセージのモニタリング
func onRecvRoomCustomMsg(cmd: String, message: String, userInfo: VoiceRoomUserInfo) {
if cmd == "CMD_DANMU" {
// 弾幕コメントの受信
}
if cmd == "CMD_LIKE" {
// 「いいね」情報の受信
}
}
```

©2013-2019 Tencent Cloud. All rights reserved.

ボイスチャットルーム(Android)

最終更新日：：2022-03-09 18:10:40

デモンストレーション

ダウンロードからAppをインストールして、マイク管理、低レイテンシー音声インタラクション、文字チャットなどのボイスチャットのユースケースにおけるTRTCの関連機能を含む、ボイスチャットルームの機能を体験できます。

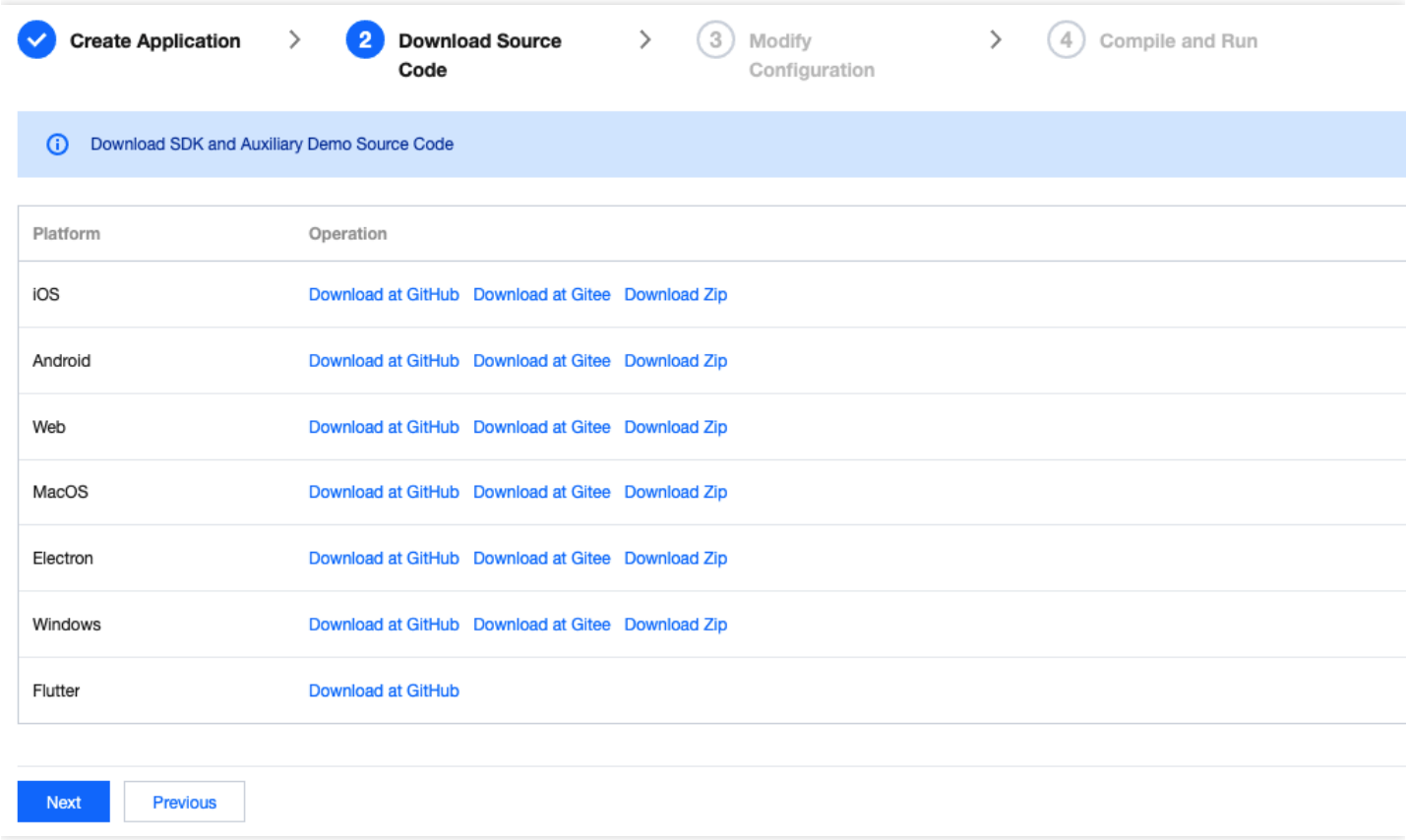
ボイスチャットルーム機能をすばやく実装する必要がある場合は、当社が提供するAppを基に直接変更を加えて適合させることも、当社が提供するTRTCVoiceRoomコンポーネントでカスタマイズしたUIを実装することも可能です。

AppのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援**>**Demoクイックスタート**を選択します。
2. `TestLiveRoom` などのアプリケーション名を入力して、 **作成 **をクリックします。
3. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。

©2013-2019 Tencent Cloud. All rights reserved.



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUIVoiceRoomに進み、ソースコードをCloneまたはダウンロードします。

手順3：Appプロジェクトファイルの設定

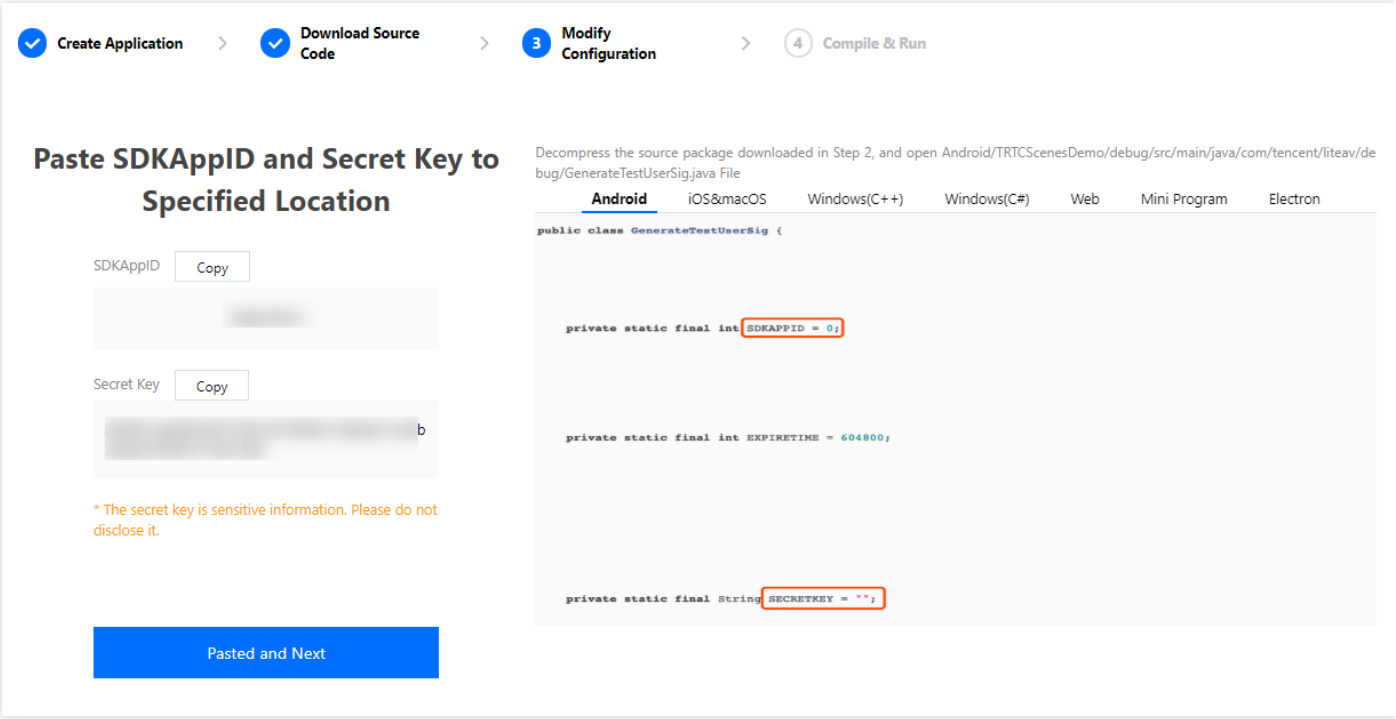
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `Android/Debug/src/main/java/com/tencent/liteav/debug/GenerateTestUserSig.java` のファイルを見つけて開きます。

3. `GenerateTestUserSig.java` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはプレースホルダー(PLACEHOLDER)。実際のSDKAppIDを設定してください。

©2013-2019 Tencent Cloud. All rights reserved.

- SECRETKEY：デフォルトはプレースホルダー(PLACEHOLDER)。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Android Studio（バージョン3.5以上）を使用してソースコード `TUIVoiceRoom` プロジェクトを開き、 **実行 ** をクリックすれば、このAppのデバッグが開始されます。

手順5：Appソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードのSourceフォルダは、uiとmodelという2つのサブフォルダを含んでいます。uiフォルダにはインターフェースコードが含まれています。以下の表にファイルまたはフォルダおよび対応するUIをリストアップして、二次調整を行いやすくしています。

ファイルまたはフォルダ	機能の説明
base	UIに使用される基礎となるクラス。
room	メインルームページは、管理者とリスナーという2種類のインターフェースがあります。
widget	汎用ウィジェット。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. **ルームの作成**をクリックします。
3. ルームトピックを入力し、**チャット開始**をクリックします。

ユーザーB

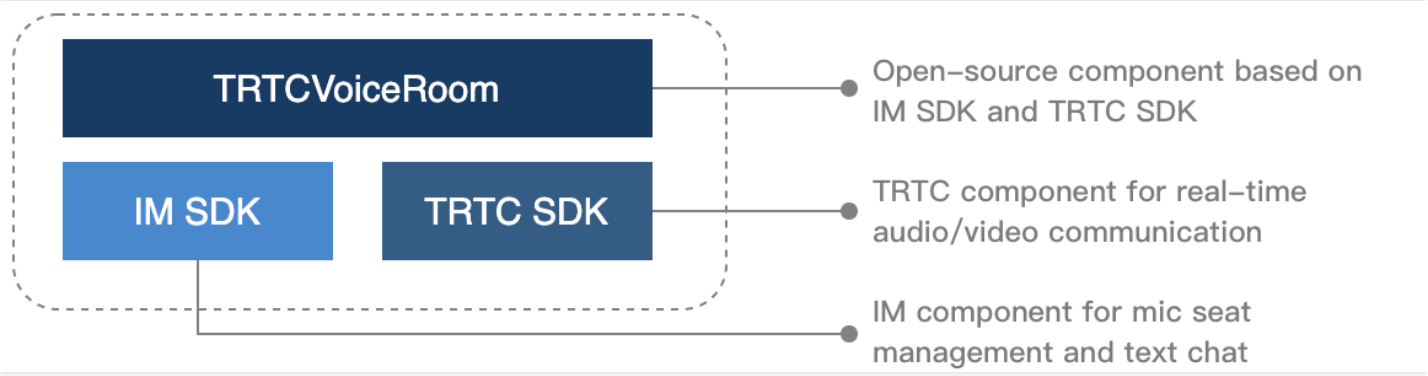
1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ユーザーAが作成したルームナンバーを入力し、**入室する**をクリックします。

> 注意：
> ルームナンバーはユーザーAのルーム上部に表示されます。

UIカスタマイズの実装

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードのSourceフォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTRTCVoiceRoomがあります。 `TRTCVoiceRoom.java` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタマイズしたUIを実装することができます。



手順1：SDKの統合

ボイスチャットコンポーネントTRTCVoiceRoom は、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

方法1：Mavenリポジトリを介する依存

1. dependenciesにTRTCSDKとIMSDKの依存を追加します。

```
dependencies {
complie ”com.tencent.liteav:LiteAVSDK_TRTC:latest.release”
complie ’com.tencent.imsdk:imsdk:latest.release’
compile ’com.google.code.gson:gson:2.3.1’
}
```

2. defaultConfigでAppが使用するCPUアーキテクチャを指定します。

```
defaultConfig {
ndk {
abiFilters ”armeabi-v7a”
}
}
```

3. **Sync Now**をクリックし、自動でSDKをダウンロードし、プロジェクトに統合します。

©2013-2019 Tencent Cloud. All rights reserved.

方法2：ローカルAARを介する依存

開発環境でのMavenリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定および難読化ルール

AndroidManifest.xmlにAppの権限を設定します。SDKには次の権限が必要です（6.0以降のAndroidシステムではストレージ読み取りの権限を動的に申請する必要があります）。

```
<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.READ_PHONE_STATE">
```

proguard-rules.proファイルでは、SDK関連を非難読化リストに追加します。

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

手順3： TRTCVoiceRoom コンポーネントをインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
Source/src/main/java/com/tencent/liteav/trtcvoiceroom/model
```

手順4：コンポーネントの作成およびログイン

1. `sharedInstance` インターフェースを呼び出すと、TRTCVoiceRoomコンポーネントのインスタンスオブジェクトを作成できます。
2. `setDelegate` 関数を呼び出してコンポーネントのイベント通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	機能

©2013-2019 Tencent Cloud. All rights reserved.

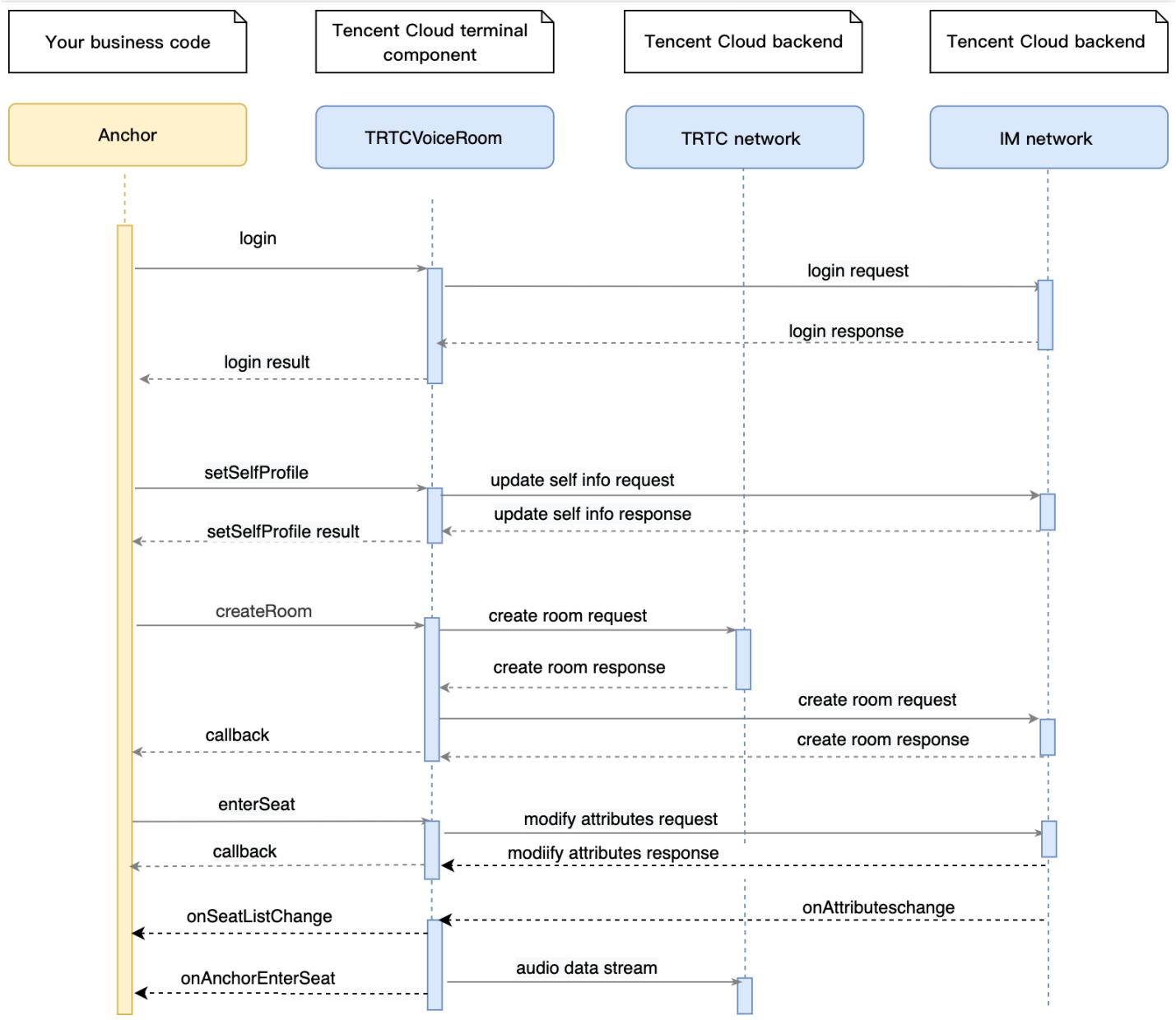
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
callback	ログインのコールバック。成功時にcodeは0になります。

```
TRTCVoiceRoom mTRTCVoiceRoom = TRTCVoiceRoom.sharedInstance(this);
mTRTCVoiceRoom.setDelegate(this);
mTRTCVoiceRoom.login(SDKAPPID, userId, userSig, new TRTCVoiceRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {![](https://main.qcloudimg.com/raw/3151fc0df16c063db7a75f6c1facf562.png)
//ログイン成功
}
}
});
```

手順5：管理者側での配信開始

1. 管理者は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 管理者は、 `createRoom` を呼び出して新しいボイスチャットルームを作成します。この時にルームID、マイク・オンに管理者の確認の要否、マイク数などルームのプロパティ情報を渡します。
3. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
4. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
5. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.管理者は、ニックネームおよびプロフィール画像を設定します
mTRTCVoiceRoom.setSelfProfile("my_name", "my_face_url", null);

// 2.管理者は、createRoomを呼び出してルームを作成します
final TRTCVoiceRoomDef.RoomParam roomParam = new TRTCVoiceRoomDef.RoomParam();
roomParam.roomName = 「ルーム名」;
roomParam.needRequest = false; // マイク・オンに対する管理者の確認の要否
roomParam.seatCount = 7; // ルームの座席数。ここでは計7席あり、管理者が1席を占め、残り6席がリスナーとなります
roomParam.coverUrl = 「ルームカバー図のURL」;
mTRTCVoiceRoom.createRoom(mRoomId, roomParam, new TRTCVoiceRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
if (code == 0) {
//3. 0番の席を占有します
mTRTCVoiceRoom.enterSeat(0, new TRTCVoiceRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
}
}
});
}
}
});

// 4.席の占有に成功した後、onSeatListChangeイベント通知を受信します
@Override
public void onSeatListChange(final List<trtcvoiceroomdef.seatinfo> seatInfoList) {
// マイクリストの表示
}

// 5. onAnchorEnterSeatイベント通知を受信
@Override
public void onAnchorEnterSeat(TRTCVoiceRoomDef.UserInfo userInfo) {
}
```

手順6：リスナー側の視聴

1. リスナー側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. リスナー側は、業務バックエンドに最新のボイスチャットルームのルームリストを取得します。

> 説明：
> Appのボイスチャットルームリストは、デモンストレーション用のためだけのものです。ボイスチャットルームリストのビジネスロジックは様々です。Tencent Cloudは現在、ボイスチャットルームリストの管理サービスを提供していませんので、ご自身でボイスチャットのルームリストを管理してください。

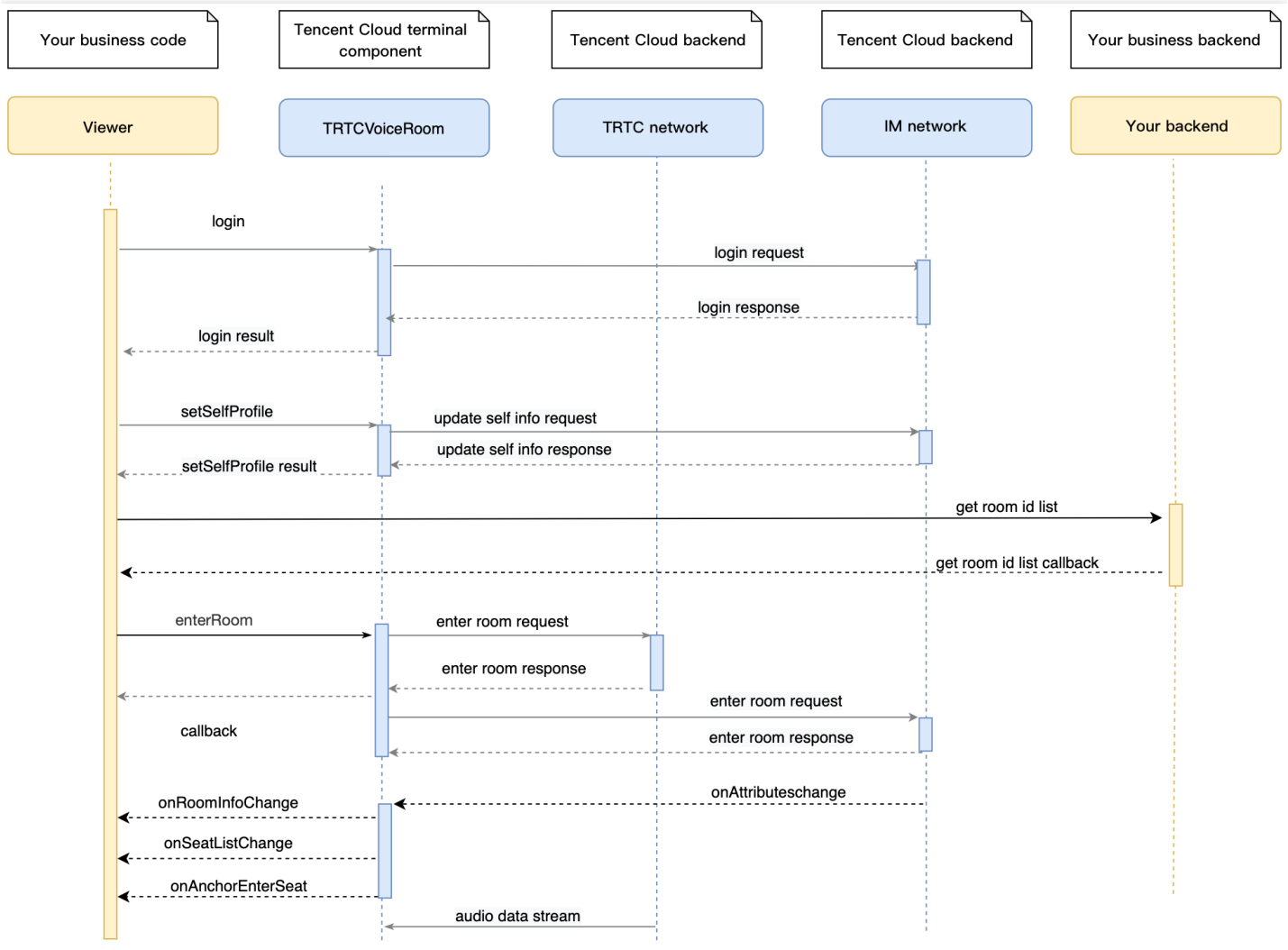
3. リスナー側は、 `getRoomInfoList` を呼び出してルームの詳細情報を取得します。この情報は、管理者側が `createRoom` を呼び出してボイスチャットルームを作成するときに設定する簡単な説明情報です。

> 注意：

©2013-2019 Tencent Cloud. All rights reserved.

ボイスチャットルームのリストに十分に包括的な情報がある場合は、 `getRoomInfoList` の呼び出しに関する手順をスキップできます。
4. リスナーは1つのボイスチャットルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。

4. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
5. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
6. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。




```
// 1. リスナーは、ニックネームおよびプロフィール画像を設定します
mTRTCVoiceRoom.setSelfProfile("my_name", "my_face_url", null);
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// 2.業務バックエンドから取得したルームリストをroomListと仮定します
List<integer> roomList = GetRoomList();

// 3.getRoomInfoListを呼び出すことによって、ルームの詳細情報を取得します
mTRTCVoiceRoom.getRoomInfoList(roomList, new TRTCVoiceRoomCallback.RoomInfoCallback() {
@Override
public void onCallback(int code, String msg, List<trtcvoiceroomdef.roominfo> list) {
if (code == 0) {
// この時、自分のUIルームリストを更新することができます
}
}
});

// 4.ボイスチャットルームを選択後、roomIdを渡して入室します
mTRTCVoiceRoom.enterRoom(roomId, new TRTCVoiceRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//入室に成功
}
}
});

// 5.入室に成功後、onRoomInfoChangeイベント通知を受信します
@Override
public void onRoomInfoChange(TRTCVoiceRoomDef.RoomInfo roomInfo) {
mNeedRequest = roomInfo.needRequest;
mRoomName = roomInfo.roomName;
// UIはタイトルなどを表示することができます
}

// 6.入室に成功後、onSeatListChangeイベント通知を受信します
@Override
public void onSeatListChange(final List<trtcvoiceroomdef.seatinfo> seatInfoList) {
// マイクリストの表示
}

// 7. onAnchorEnterSeatイベント通知を受信します
@Override
public void onAnchorEnterSeat(TRTCVoiceRoomDef.UserInfo userInfo) {
}
```

手順7：マイク管理

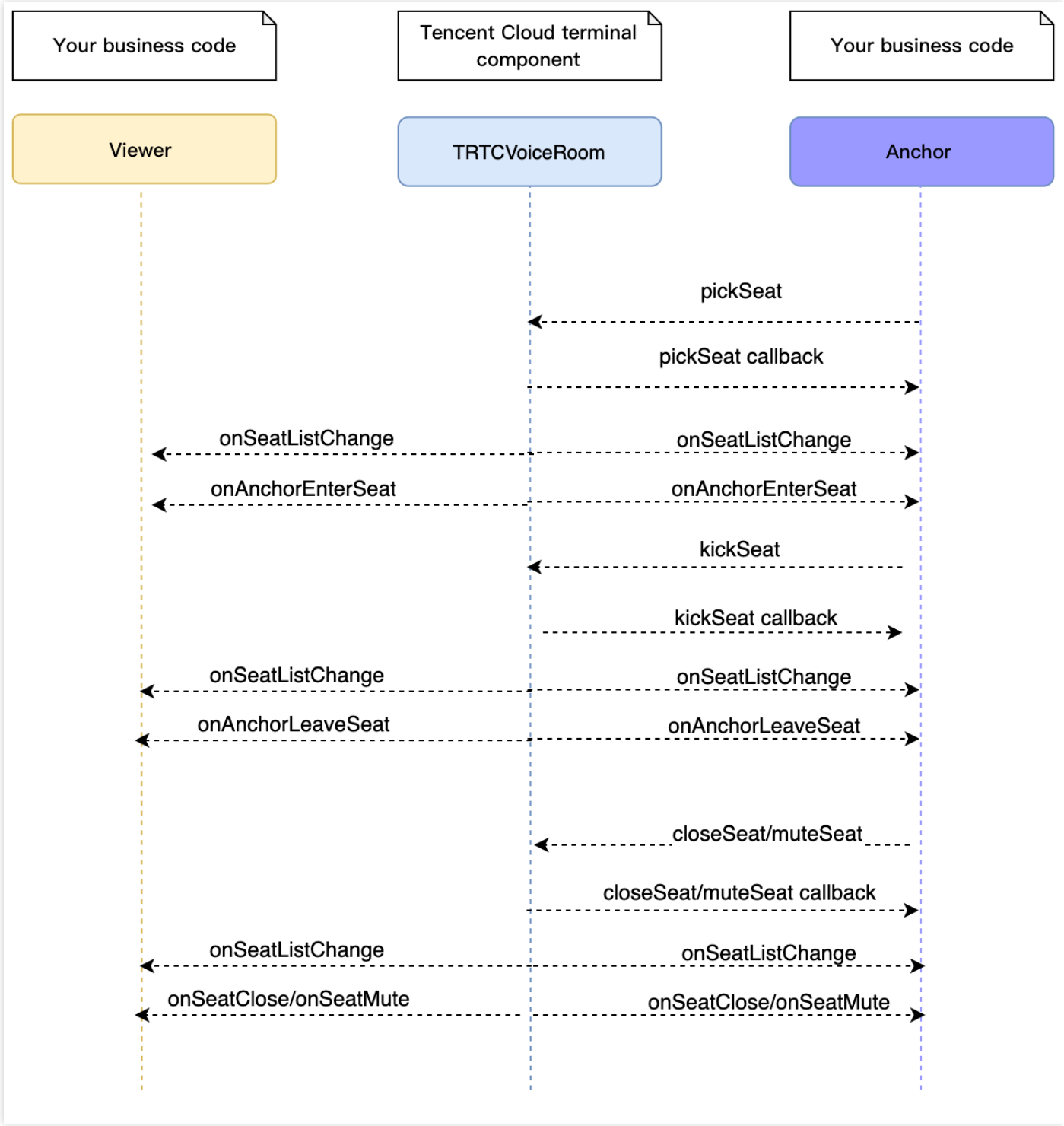
- 管理者側
- リスナー側

©2013-2019 Tencent Cloud. All rights reserved.

1. `pickSeat` は、対応するマイクおよびリスナーのuserIdを渡し、視聴者を発言できるように招待できます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。
2. `kickSeat` は、対象となるマイクを渡した後、キックアウトしてマイク・オフにすることができます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。
3. `muteSeat` は、対応するマイクを渡した後、ミュート/ミュート解除をすることができます。ルーム内の全参加者は `onSeatListChange` および `onSeatMute` というイベント通知を受信します。
4. `closeSeat` は、対応するマイクを送信後、任意のマイクのクローズ/解除をすることができます。クローズ後は、リスナー側はこれ以上マイクを使用することはできません。ルーム内の全参加者

©2013-2019 Tencent Cloud. All rights reserved.

は `onSeatListChange` および `onSeatClose` というイベント通知を受信します。



手順8：招待シグナリングの使用

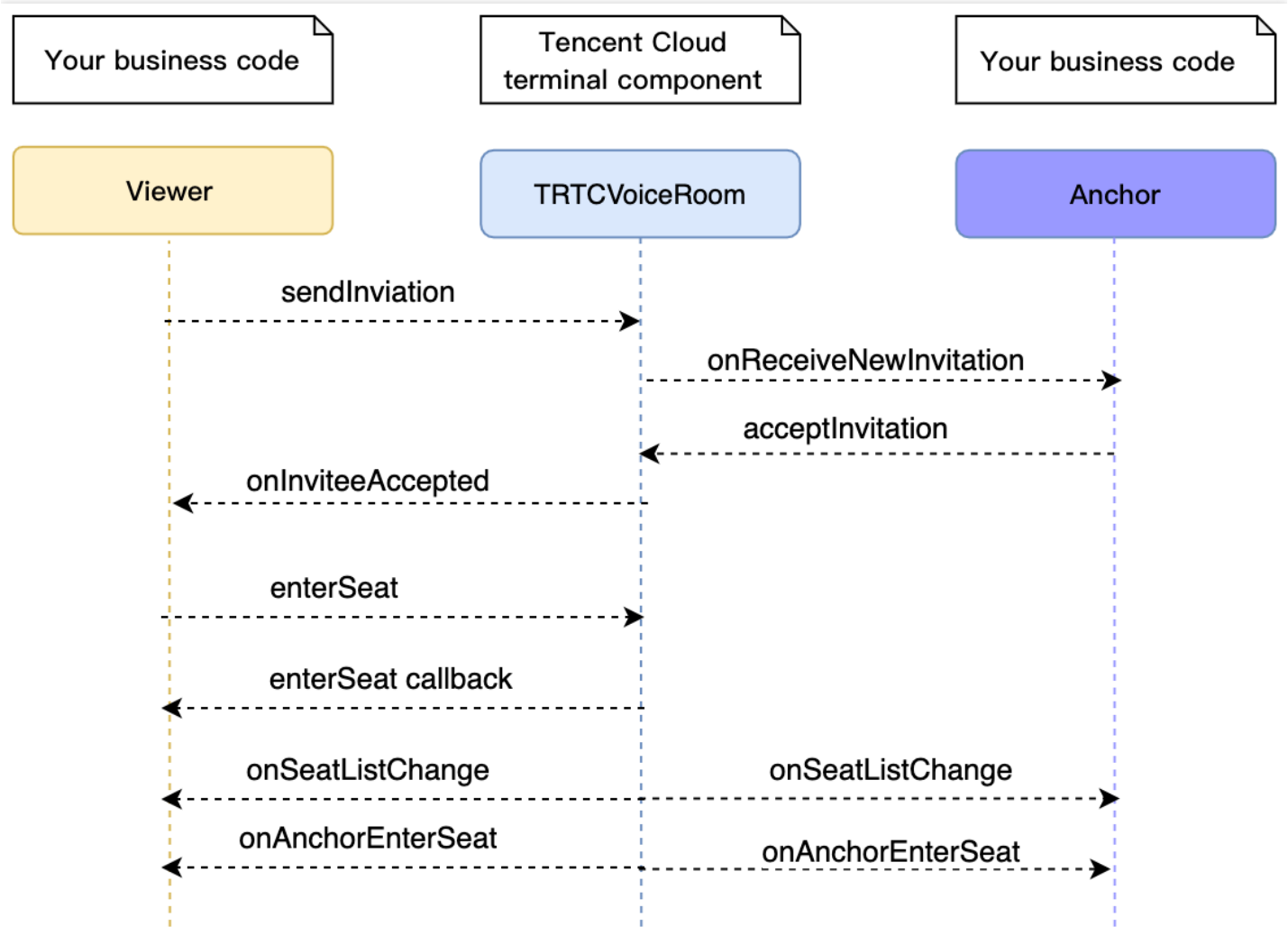
マイク管理では、リスナーがマイク・オン/オフにする場合や、管理者が視聴者を発言できるように招待する場合は、相手の同意を必要とせずに直接操作することができます。

©2013-2019 Tencent Cloud. All rights reserved.

Appが次の操作の業務フローを実施するために、相手の同意を必要とする場合、招待シグナリングによって対応するサポートを提供することができます。

- リスナーからの自主的なマイク・オン申請
- 管理者からの招待によるリスナーのマイク・オン

1. リスナー側が `sendInvitation` を呼び出し、管理者のuserIdおよび業務のカスタムコマンドワードなどを渡します。この時、関数が1つのinviteIdを返しますので、このinviteIdを記録します。
2. 管理者側は、 `onReceiveNewInvitation` というイベント通知を受信します。この時UIでウィンドウをポップアップさせ、管理者に同意の有無をたずねることができます。
3. 管理者が同意を選択後、 `acceptInvitation` を呼び出してinviteIdを渡します。
4. リスナー側は、 `onInviteeAccepted` というイベント通知を受信し、 `enterSeat` を呼び出してマイク・オンにします。



©2013-2019 Tencent Cloud. All rights reserved.

```
// リスナー側の視点
// 1. sendInvitationを呼び出し、1号マイクの使用をリクエストします
String inviteId = mTRTCVoiceRoom.sendInvitation("ENTER_SEAT", ownerUserId, "1", null);

// 2.招待のリクエスト同意を受信し、正式にマイク・オンになります
@Override
public void onInviteeAccepted(String id, String invitee) {
if(id.equals(inviteId)) {
mTRTCVoiceRoom.enterSeat(1, null);
}
}

// 管理者側の視点
// 1.管理者がリクエストを受信します
@Override
public void onReceiveNewInvitation(final String id, String inviter, String cmd, final String cont
ent) {
if (cmd.equals("ENTER_SEAT")) {
// 2.管理者がリスナーのリクエストに同意します
mTRTCVoiceRoom.acceptInvitation(id, null);
}
}
```

手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、このルーム内のすべてのキャスターおよびリスナーが `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

```
// 送信側：テキストメッセージの送信
mTRTCVoiceRoom.sendRoomTextMsg("Hello Word!", null);
// 受信側：テキストメッセージのモニタリング
mTRTCVoiceRoom.setDelegate(new TRTCVoiceRoomDelegate() {
@Override
public void onRecvRoomTextMsg(String message, TRTCVoiceRoomDef.UserInfo userInfo) {
Log.d(TAG、 + userInfo.userName + 「からのメッセージを受信:」 + message);
}
});
```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターとリスナーが `onRecvRoomCustomMsg` コールバックを受信できます。
  カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

©2013-2019 Tencent Cloud. All rights reserved.

```
// 送信側：カスタマイズCmdによって弾幕と「いいね」情報を区別することができます
// eg:「CMD_DANMU」は弾幕コメントを表し、「CMD_LIKE」は「いいね」情報を表します
mTRTCVoiceRoom.sendRoomCustomMsg("CMD_DANMU", "Hello world", null);
mTRTCVoiceRoom.sendRoomCustomMsg("CMD_LIKE", "", null);
// 受信側：カスタムメッセージのモニタリング
mTRTCVoiceRoom.setDelegate(new TRTCVoiceRoomDelegate() {
@Override
public void onRecvRoomCustomMsg(String cmd, String message, TRTCVoiceRoomDef.UserInfo userInf
o) {
if ("CMD_DANMU".equals(cmd)) {
// 弾幕コメントの受信
Log.d(TAG、 + userInfo.userName + 「からの弾幕コメントを受信:」 + message);
} else if ("CMD_LIKE".equals(cmd)) {
// 「いいね」情報の受信
Log.d(TAG、 userInfo.userName + 「いいねを付けました！」);
}
}
});
```

©2013-2019 Tencent Cloud. All rights reserved.

TRTCVoiceRoom (iOS)

最終更新日：：2021-12-14 16:32:30

TRTCVoiceRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- 管理者が新しいボイスチャットルームを作成して配信を開始し、リスナーがボイスチャットルームに参加して視聴/インタラクティブなコミュニケーションを行います。
- 管理者は、リスナーを招待してマイク・オンにしたり、マイク・オンのキャスターをキックアウトしたりすることもできます。
- 管理者はまた、座席をクローズすることができ、その他のリスナーはマイク・オンを申請することができなくなります。
- リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、他者と音声インタラクションを行うことができます。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
- 各種のテキストメッセージやカスタマイズメッセージの発信をサポートし、カスタムメッセージは弾幕、「いいね」、ギフトなどに使用することきます。

TRTCVoiceRoomは、オープンソースのClassであり、Tencent Cloudの2つのクローズソースであるSDKに依存しています。具体的な実装プロセスは ボイスチャットルーム（iOS）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用しています。
- IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCVoiceRoom API 概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。
setDelegate	イベントコールバックを設定します。
setDelegateHandler	イベントのコールバックが配置されているスレッドを設定します。
login	ログイン。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。nilの場合は、ルーム内全員の情報を取得します。

マイク管理インターフェース

API	説明
enterSeat	ユーザーが発言者になる（リスナー側/管理者ともに呼び出し可）。
moveSeat	マイクの移動 (マイク・オンするキャスター側で呼び出せます) 。
leaveSeat	自主的にマイク・オフ（キャスターが呼び出し）。
pickSeat	視聴者が発言できるように招待（管理者が呼び出し）。
kickSeat	キックアウトしてマイク・オフ（管理者が呼び出し）。
muteSeat	任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
closeSeat	任意のマイクのクローズ/解除（管理者が呼び出し）。

ローカルの音声操作インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
setAudioQuality	音質の設定。
muteLocalAudio	ローカルオーディオミュートの開始/停止。
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。
setVoiceEarMonitorEnable	インイヤーモニタリングのオン/オフ。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerの取得。

メッセージ送信関連インターフェース

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

招待シグナリング関連インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
sendInvitation	ユーザーに招待を送信。
acceptInvitation	招待の同意。
rejectInvitation	招待の辞退。
cancelInvitation	招待の取り消し。

TRTCVoiceRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが廃棄された時のコールバック。
onRoomInfoChange	ボイスチャットルーム情報変更のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

API	説明
onSeatListChange	全量のマイクリストの変更。
onAnchorEnterSeat	発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。
onAnchorLeaveSeat	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onSeatMute	管理者のマイクミュート。
onUserMicrophoneMute	ユーザーのマイクがミュートされているかどうか。
onSeatClose	管理者のマイククローズ。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

シグナリングイベントのコールバック

API	説明
onReceiveNewInvitation	新規招待リクエストの受信。
onInviteeAccepted	被招待者が招待に同意。
onInviteeRejected	被招待者が招待を拒否。
onInvitationCancelled	招待者が招待を取り消し。

SDK基本関数

sharedInstance

TRTCVoiceRoom シングルトンオブジェクトを取得します。

```
/**
* TRTCVoiceRoom シングルトンオブジェクトの取得
```

©2013-2019 Tencent Cloud. All rights reserved.

```
*
* - returns: TRTCVoiceRoom インスタンス
* - note: {@link TRTCVoiceRoom#destroySharedInstance()}を呼び出してシングルトンオブジェクトを破棄することができます
*/
+ (instancetype)sharedInstance NS_SWIFT_NAME(shared());
```

destroySharedInstance

TRTCVoiceRoom シングルトンオブジェクトを破棄します。

> 説明：
> インスタンスの破棄後は、外部にキャッシュされた TRTCVoiceRoomインスタンスの再使用ができません。改めて sharedInstance を呼び出し、インスタンスを新規取得する必要があります。

```
/**
* TRTCVoiceRoom のシングルトンオブジェクトの破棄
*
* - note: インスタンスの破棄後は、外部にキャッシュされたTRTCVoiceRoomインスタンスの再使用ができません。改めて {@link TRTCVoiceRoom#sharedInstance()} を呼び出し、インスタンスを新規取得する必要があります
*/
+ (void)destroySharedInstance NS_SWIFT_NAME(destroyShared());
```

setDelegate

TRTCVoiceRoom イベントコールバック、 TRTCVoiceRoomDelegate を介して TRTCVoiceRoom の各種ステータス通知を受け取ることができます。

```
/**
* コンポーネントコールバックインターフェースの設定
*
* TRTCVoiceRoomDelegate によって TRTCVoiceRoom の各種ステータス通知を取得することができます
*
* - parameter delegateコールバックインターフェース
* - note: TRTCVoiceRoomのコールバックイベントは、デフォルトではMain Queueでコールバックされます。イベントのコールバックが配置されているキューを指定する必要がある場合は、 {@link TRTCVoiceRoom#setDelegateQueue(queue)}を使用することができます
*/
- (void)setDelegate:(id<TRTCVoiceRoomDelegate>)delegate NS_SWIFT_NAME(setDelegate(delegate:));
```

©2013-2019 Tencent Cloud. All rights reserved.

説明：
setDelegate は、 TRTCVoiceRoom のプロキシコールバックです。

setDelegateQueue

イベントコールバックが所在するスレッドキューを設定し、デフォルトはメインスレッド MainQueue に発送します。

```
/**
* イベントコールバックが配置されているキューの設定
*
* - parameter queueキュー, TRTCVoiceRoomの各種ステータス通知コールバックは、指定したqueueに発信されます。
*/
- (void)setDelegateQueue:(dispatch_queue_t)queue NS_SWIFT_NAME(setDelegateQueue(queue:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
パラメータ	タイプ	意味
queue	dispatch_queue_t	TRTCVoiceRoomの各種ステータス通知は、指定したスレッドキューに発信されます。

login

ログイン

```
- (void)login:(int)sdkAppID
userId:(NSString *)userId
userSig:(NSString *)userSig
callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(login(sdkAppID:userId:userSig:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	**TRTCコンソール >アプリケーション管理>** アプリケーション情報でSDKAppIDを確認できます。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	NSString	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	NSString	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

logout

ログアウト。

```
- (void)logout:(ActionCallback _Nullable)callback NS_SWIFT_NAME(logout(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ログアウトのコールバック。成功時にcodeは0になります。

setSelfProfile

個人情報の修正。

```
- (void)setSelfProfile:(NSString *)userName avatarURL:(NSString *)avatarURL callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(setSelfProfile(userName:avatarURL:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	NSString	ニックネーム。
avatarURL	NSString	プロフィール画像のアドレス。
callback	ActionCallback	個人情報設定のコールバック。成功時にcodeは0になります。

ルーム関連インターフェース関数

createRoom

©2013-2019 Tencent Cloud. All rights reserved.

ルームの作成（管理者が呼び出し）。

```
- (void)createRoom:(int)roomID roomParam:(VoiceRoomParam *)roomParam callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(createRoom(roomID:roomParam:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomIDを、1つのボイスチャットルームリストにまとめることができます。Tencent Cloudでは現在、ボイスチャットルームリストの管理サービスを行っていませんので、ご自身でボイスチャットルームリストを管理してください。
roomParam	VoiceRoomParam	ルーム情報は、ルーム名、マイク情報、カバー情報など、ルームを説明するために用いる情報に使用します。マイク管理が必要な場合は、ルームのマイク数を入力する必要があります。
callback	ActionCallback	ルームの新規作成結果のコールバック。成功時にcodeは0になります。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです。

1. 管理者は、 `createRoom` を呼び出して新しいボイスチャットルームを作成します。この時にルームID、マイク・オンに管理者の確認の要否、マイク数などルームのプロパティ情報を渡します。
2. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
3. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリスト変更をUI上に更新することができます。
4. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。

```
- (void)destroyRoom:(ActionCallback _Nullable)callback NS_SWIFT_NAME(destroyRoom(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

©2013-2019 Tencent Cloud. All rights reserved.

enterRoom

入室（リスナーが呼び出し）。

```
- (void)enterRoom:(NSInteger)roomID callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(enterRoom(roomID:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	NSInteger	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

リスナーが入室し聴取する際の通常の呼び出しプロセスは次のとおりです。

1. リスナーがサーバーから取得する最新のボイスチャットルームリストには、多くのボイスチャットルームのroomIdおよびルーム情報が含まれる場合があります。
   2. リスナーが1つのボイスチャットルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
2. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
3. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時にマイクリストの変更をUI上に更新することができます。
4. 入室後にマイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。

exitRoom

ルームから退出します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

©2013-2019 Tencent Cloud. All rights reserved.

説明：
ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

```
- (void)getRoomInfoList:(NSArray<NSNumber *> *)roomIdList callback:(VoiceRoomInfoCallback _Nullable)callback NS_SWIFT_NAME(getRoomInfoList(roomIdList:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIdList	NSArray&dxlt;NSNumber&dxgt;	ルームナンバーリスト。
callback	RoomInfoCallback	ルーム詳細情報のコールバック。

getUserInfoList

指定されたuserIdのユーザー情報を取得します。

```
- (void)getUserInfoList:(NSArray<NSString *> * _Nullable)userIDList callback:(VoiceRoomUserListCallback _Nullable)callback NS_SWIFT_NAME(getUserInfoList(userIDList:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIdList	NSArray&dxlt;NSString&dxgt;	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。
userlistcallback	UserListCallback	ユーザーの詳細情報のコールバック。

マイク管理インターフェース

enterSeat

ユーザーが発言者になります（リスナー側/管理者ともに呼び出し可）。

説明：
マイク・オンの成功後、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)enterSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(enterSeat(seatIndex:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	NSInteger	マイク・オンの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、 `onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

moveSeat

マイクの移動 (マイク・オンするキャスター側で呼び出せます)。

> 説明：
> マイクの移動に成功したら、ルーム内のすべてのメンバーが `onSeatListChange` 、 `onAnchorLeaveSeat` 、 `onAnchorEnterSeat` のイベント通知を受け取ります。(キャスターが呼び出した後に変更できるのは、マイクのシート番号情報のみで、ユーザーのキャスターロールを切り替えることはできません。)

```
- (NSInteger)moveSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(moveSeat(seatIndex:callback:))
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	NSInteger	移動の必要があるマイク番号。
callback	ActionCallback	操作コールバック。

戻り値：

戻り値	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

戻り値	タイプ	意味
code	NSInteger	マイク移動操作の結果（0は成功、その他は失敗、10001はインターフェース呼び出し頻度制限）。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、`onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

leaveSeat

ユーザーが視聴者になる（キャスターが呼び出し）。

> 説明：
> マイク・オフの成功後、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

```
- (void)leaveSeat:(ActionCallback _Nullable)callback NS_SWIFT_NAME(leaveSeat(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

pickSeat

視聴者が発言できるように招待（管理者が呼び出し）。

> 説明：
> 管理者が視聴者を発言できるように招待すると、ルーム内の全メンバーは、`onSeatListChange` と `onAnchorEnterSeat` というイベント通知を受信します。

```
- (void)pickSeat:(NSInteger)seatIndex userId:(NSString *)userId callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(pickSeat(seatIndex:userId:callback:));
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
seatIndex	NSInteger	マイク・オンの必要があるマイク番号。
userId	NSString	ユーザーID。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。管理者がリスナーの同意がなければマイク・オンできないケースの場合は、まず `sendInvitation` を呼び出してからリスナーに申請し、`onInvitationAccept` を受信すると、この関数を呼び出せるようになります。

kickSeat

キックアウトしてマイク・オフ（管理者が呼び出し）。

> 説明：
> 管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

```
- (void)kickSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(kickSeat(seatIndex:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	NSInteger	キックアウトしてマイク・オフの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

muteSeat

任意のマイクのミュート/ミュート解除（管理者が呼び出し）。

> 説明：
> 任意のマイクのミュート/ミュート解除は、ルーム内の全メンバーが `onSeatListChange` および `onSeatMute` というイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)muteSeat:(NSInteger)seatIndex isMute:(BOOL)isMute callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(muteSeat(seatIndex:isMute:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	NSInteger	操作が必要なマイク番号。
isMute	BOOL	YES：該当するマイクのミュート；NO：該当するマイクのミュート解除。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。seatIndexの座席に該当するキャスターは、muteAudioを自動的に呼び出してミュート/ミュートオフにします。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。

> 説明：
> 管理者は、該当するマイクをクローズ/解除し、ルーム内の全メンバーは `onSeatListChange` および `onSeatClose` というイベント通知を受信します。

```
- (void)closeSeat:(NSInteger)seatIndex isClose:(BOOL)isClose callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(closeSeat(seatIndex:isClose:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	NSInteger	操作が必要なマイク番号。
isClose	BOOL	YES：該当するマイクのクローズ； NO：該当するマイクのクローズ解除。
callback	ActionCallback	操作コールバック。

©2013-2019 Tencent Cloud. All rights reserved.

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。該当するseatIndexの座席上のキャスターはクローズされ、自動的にマイク・オフになります。

ローカル音声操作のインターフェース

startMicrophone

マイクの集音開始。

```
- (void)startMicrophone;
```

stopMicrophone

マイクの集音停止。

```
- (void)stopMicrophone;
```

setAudioQuality

音質の設定。

```
- (void)setAuidoQuality:(NSInteger)quality NS_SWIFT_NAME(setAuidoQuality(quality:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	NSInteger	オーディオ品質。詳細については、TRTC SDKをご参照ください。

muteLocalAudio

ローカルの音声のミュート/ミュート取り消し。

```
- (void)muteLocalAudio:(BOOL)mute NS_SWIFT_NAME(muteLocalAudio(mute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	BOOL	ミュート/ミュートの取り消し。詳細については、TRTC SDKをご参照ください。

setSpeaker

©2013-2019 Tencent Cloud. All rights reserved.

スピーカーの起動設定。

```
- (void)setSpeaker:(BOOL)userSpeaker NS_SWIFT_NAME(setSpeaker(userSpeaker:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
useSpeaker	BOOL	YES：スピーカー；NO：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。

```
- (void)setAudioCaptureVolume:(NSInteger)voluem NS_SWIFT_NAME(setAudioCaptureVolume(volume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	NSInteger	集音音量、0 - 100、デフォルト100。

setAudioPlayoutVolume

再生音量の設定。

```
- (void)setAudioPlayoutVolume:(NSInteger)volume NS_SWIFT_NAME(setAudioPlayoutVolume(volume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	NSInteger	再生音量、0 - 100、デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。

```
- (void)muteRemoteAudio:(NSString *)userId mute:(BOOL)mute NS_SWIFT_NAME(muteRemoteAudio(userId:mute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	NSString	指定のユーザーID。
mute	BOOL	YES：ミュート起動；NO：ミュート停止。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。

```
- (void)muteAllRemoteAudio:(BOOL)isMute NS_SWIFT_NAME(muteAllRemoteAudio(isMute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	BOOL	YES：ミュート起動；NO：ミュート停止。

setVoiceEarMonitorEnable

インイヤーモニタリングのオン/オフ。

```
- (void)setVoiceEarMonitorEnable:(BOOL)enable NS_SWIFT_NAME(setVoiceEarMonitor(enable:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	BOOL	YES：インイヤーモニタリング起動；NO：インイヤーモニタリング停止。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

バックグラウンド・サウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

```
- (TXAudioEffectManager * _Nullable)getAudioEffectManager;
```

メッセージ送信関連インターフェース関数

sendRoomTextMsg

©2013-2019 Tencent Cloud. All rights reserved.

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

```
- (void)sendRoomTextMsg:(NSString *)message callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendRoomTextMsg(message:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	NSString	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

```
- (void)sendRoomCustomMsg:(NSString *)cmd message:(NSString *)message callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendRoomCustomMsg(cmd:message:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	NSString	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	NSString	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

招待シグナリング関連インターフェース

sendInvitation

ユーザーに招待を送信。

```
- (NSString *)sendInvitation:(NSString *)cmd
userId:(NSString *)userId
content:(NSString *)content
callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendInvitation(cmd:userId:content:callback:));
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
cmd	NSString	業務カスタマイズコマンド。
userId	NSString	招待するユーザーID。
content	NSString	招待するコンテンツ。
callback	ActionCallback	送信結果のコールバック。

戻り値：

戻り値	タイプ	意味
inviteId	NSString	今回の招待IDの識別に使用。

acceptInvitation

招待の同意。

```
- (void)acceptInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(acceptInvitation(id:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	NSString	招待ID。
callback	ActionCallback	送信結果のコールバック。

rejectInvitation

招待の拒否。

```
- (void)rejectInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(rejectInvitation(id:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	NSString	招待ID。
callback	ActionCallback	送信結果のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

cancelInvitation

招待の取り消し。

```
- (void)cancelInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(cancelInvitation(id:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	NSString	招待ID。
callback	ActionCallback	送信結果のコールバック。

TRTCVoiceRoomDelegateイベントのコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

```
- (void)onError:(int)code
message:(NSString*)message
NS_SWIFT_NAME(onError(code:message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	NSString	エラー情報。

onWarning

©2013-2019 Tencent Cloud. All rights reserved.

警告のコールバック。

```
- (void)onWarning:(int)code
message:(NSString *)message
NS_SWIFT_NAME(onWarning(code:message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	NSString	警告メッセージ。

onDebugLog

Logコールバック。

```
- (void)onDebugLog:(NSString *)message
NS_SWIFT_NAME(onDebugLog(message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	NSString	ログ情報。

ルームイベントのコールバック

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。

```
- (void)onRoomDestroy:(NSString *)roomId
NS_SWIFT_NAME(onRoomDestroy(roomId:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	NSString	ルーム ID。

onRoomInfoChange

©2013-2019 Tencent Cloud. All rights reserved.

入室に成功後、このインターフェースをコールバックします。roomInfoの情報は、管理者がルームを作成するときに渡されます。

```
- (void)onRoomInfoChange:(VoiceRoomInfo *)roomInfo
NS_SWIFT_NAME(onRoomInfoChange(roomInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomInfo	VoiceRoomInfo	ルーム情報。

onUserMicrophoneMute

ユーザーのマイクがミュートになっているかどうかについて、ユーザーがmuteLocalAudioを呼び出すと、ルームの他のユーザーがこの通知を受信します。

```
- (void)onUserMicrophoneMute:(NSString *)userId mute:(BOOL)mute
NS_SWIFT_NAME(onUserMicrophoneMute(userId:mute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID 。
mute	BOOL	YES：マイクミュート； NO：ミュート解除。

onUserVolumeUpdate

音量レベルリマインダを有効にして、各メンバーの音量を通知します。

```
- (void)onUserVolumeUpdate:(NSArray<TRTCVolumeInfo *> *)userVolumes totalVolume:(NSInteger)totalV
olume
NS_SWIFT_NAME(onUserVolumeUpdate(userVolumes:totalVolume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userVolumes	NSArray	ユーザーリスト。
totalVolume	NSInteger	音量の大きさ。値：0 - 100。

©2013-2019 Tencent Cloud. All rights reserved.

マイクコールバック

onSeatListChange

全量のマイクリストの変更は、全てのマイクリストを含みます。

```
- (void)onSeatInfoChange:(NSArray<VoiceRoomSeatInfo *> *)seatInfolist
NS_SWIFT_NAME(onSeatListChange(seatInfoList:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatInfoList	NSArray&dxlt;VoiceRoomSeatInfo&dxgt;	全量のマイクリスト。

onAnchorEnterSeat

発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。

```
- (void)onAnchorEnterSeat:(NSInteger)index
user:(VoiceRoomUserInfo *)user
NS_SWIFT_NAME(onAnchorEnterSeat(index:user:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	NSInteger	メンバーがマイク・オンするマイク。
user	VoiceRoomUserInfo	マイク・オンユーザーの詳細情報。

onAnchorLeaveSeat

視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。

```
- (void)onAnchorLeaveSeat:(NSInteger)index
user:(VoiceRoomUserInfo *)user
NS_SWIFT_NAME(onAnchorLeaveSeat(index:user:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	NSInteger	マイク・オフのマイク。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
user	VoiceRoomUserInfo	マイク・オンユーザーの詳細情報。

onSeatMute

管理者のマイクミュート。

```
- (void)onSeatMute:(NSInteger)index
isMute:(BOOL)isMute
NS_SWIFT_NAME(onSeatMute(index:isMute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	NSInteger	操作するマイク。
isMute	BOOL	YES：マイクミュート； NO：ミュート解除。

onSeatClose

管理者のマイククローズ。

```
- (void)onSeatClose:(NSInteger)index
isClose:(BOOL)isClose
NS_SWIFT_NAME(onSeatClose(index:isClose:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	NSInteger	操作するマイク。
isClose	BOOL	YES：マイクのクローズ； NO： マイクのクローズ解除。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。

```
- (void)onAudienceEnter:(VoiceRoomUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceEnter(userInfo:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	VoiceRoomUserInfo	入室したリスナーの情報。

onAudienceExit

リスナー退室通知の受信。

```
- (void)onAudienceExit:(VoiceRoomUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceExit(userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	VoiceRoomUserInfo	退室したリスナーの情報。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

```
- (void)onRecvRoomTextMsg:(NSString *)message
userInfo:(VoiceRoomUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomTextMsg(message:userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	NSString	テキストメッセージ。
userInfo	VoiceRoomUserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージの受信。

```
- (void)onRecvRoomCustomMsg:(NSString *)command
message:(NSString *)message
```

©2013-2019 Tencent Cloud. All rights reserved.

```
userInfo:(VoiceRoomUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomCustomMsg(command:message:userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
command	NSString	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	NSString	テキストメッセージ。
userInfo	VoiceRoomUserInfo	送信者のユーザー情報。

招待シグナリングイベントのコールバック

onReceiveNewInvitation

新規招待リクエストの受信。

```
- (void)onReceiveNewInvitation:(NSString *)identifier
inviter:(NSString *)inviter
cmd:(NSString *)cmd
content:(NSString *)content
NS_SWIFT_NAME(onReceiveNewInvitation(id:inviter:cmd:content:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	NSString	招待ID。
inviter	NSString	招待者のユーザーID。
cmd	NSString	業務指定のコマンドワードは、開発者がカスタマイズします。
content	NSString	業務指定のコンテンツ。

onInviteeAccepted

被招待者が招待に同意。

```
- (void)onInviteeAccepted:(NSString *)identifier
invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeAccepted(id:invitee:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	NSString	招待ID。
invitee	NSString	被招待者のユーザーID。

onInviteeRejected

被招待者による招待の拒否。

```
- (void)onInviteeRejected:(NSString *)identifier
invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeRejected(id:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	NSString	招待ID。
invitee	NSString	被招待者のユーザーID。

onInvitationCancelled

招待者が招待を取り消し。

```
- (void)onInvitationCancelled:(NSString *)identifier
invitee:(NSString *)invitee NS_SWIFT_NAME(onInvitationCancelled(id:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	NSString	招待ID。
inviter	NSString	招待者のユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCVoiceRoom (Android)

最終更新日：：2021-12-14 16:32:30

TRTCVoiceRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- 管理者が新しいボイスチャットルームを作成して配信を開始し、リスナーがボイスチャットルームに参加して視聴/インタラクティブなコミュニケーションを行います。
- 管理者は、リスナーを招待してマイク・オンにしたり、マイク・オンのキャスターをキックアウトしたりすることもできます。
- 管理者はまた、座席をクローズすることができ、その他のリスナーはマイク・オンを申請することができなくなります。
- リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、他者と音声インタラクションを行うことができます。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
- 各種のテキストメッセージやカスタマイズメッセージの発信をサポートし、カスタムメッセージは弾幕、「いいね」、ギフトなどに使用することきます。

TRTCVoiceRoomは、オープンソースのClassであり、Tencent Cloudの2つのクローズソースであるSDKに依存しています。具体的な実装プロセスは ボイスチャットルーム（Android）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用しています。
- IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCVoiceRoom API 概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。
setDelegate	イベントコールバックを設定します。
setDelegateHandler	イベントのコールバックが配置されているスレッドを設定します。
login	ログイン。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。 nullの場合は、ルーム内全員の情報を取得します。

マイク管理インターフェース

API	説明
enterSeat	ユーザーが発言者になる（リスナー側/管理者ともに呼び出し可）。
moveSeat	マイクの移動（マイク・オンするキャスター側で呼び出せます）。
leaveSeat	ユーザーが視聴者になる（キャスターが呼び出し）。
pickSeat	視聴者が発言できるように招待（管理者が呼び出し）。
kickSeat	キックアウトしてマイク・オフ（管理者が呼び出し）。
muteSeat	任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
closeSeat	任意のマイクのクローズ/解除（管理者が呼び出し）。

ローカルの音声操作インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
setAudioQuality	音質の設定。
muteLocalAudio	ローカルオーディオミュートの開始/停止。
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。
setVoiceEarMonitorEnable	インイヤーモニタリングのオン/オフ。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerの取得。

メッセージ送信関連インターフェース

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

招待シグナリング関連インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
sendInvitation	ユーザーに招待を送信。
acceptInvitation	招待の同意。
rejectInvitation	招待の辞退。
cancelInvitation	招待の取り消し。

TRTCVoiceRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが廃棄された時のコールバック。
onRoomInfoChange	ボイスチャットルーム情報変更のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

API	説明
onSeatListChange	全量のマイクリストの変更。
onAnchorEnterSeat	発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。
onAnchorLeaveSeat	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onSeatMute	管理者のマイクミュート。
onUserMicrophoneMute	ユーザーのマイクがミュートされているかどうか。
onSeatClose	管理者のマイククローズ。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

シグナリングイベントのコールバック

API	説明
onReceiveNewInvitation	新規招待リクエストの受信。
onInviteeAccepted	被招待者が招待に同意。
onInviteeRejected	被招待者が招待を拒否。
onInvitationCancelled	招待者が招待を取り消し。

SDK基本関数

sharedInstance

TRTCVoiceRoom シングルトンオブジェクトを取得します。

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
context	Context	Androidコンテキスト。内部ではApplicationContextに変換してシステムAPIの呼び出しに使用します

destroySharedInstance

TRTCVoiceRoom シングルトンオブジェクトを破棄します。

> 説明：
> インスタンスの破棄後は、外部にキャッシュされた TRTCVoiceRoomインスタンスの再使用ができません。改めて sharedInstance を呼び出し、インスタンスを新規取得する必要があります。

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

setDelegate

TRTCVoiceRoom イベントコールバック、 TRTCVoiceRoomDelegate を介して TRTCVoiceRoom の各種ステータス通知を受け取ることができます。

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

> 説明：
> setDelegate は、 TRTCVoiceRoom のプロキシコールバックです。

setDelegateHandler

イベントコールバックが配置されているスレッドを設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
handler	Handler	TRTCChatSalonの各種ステータス通知は、指定したhandlerスレッドに発信されます。

login

ログイン

```
public abstract void login(int sdkAppId,
String userId, String userSig,
TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	**TRTCコンソール >アプリケーション管理>** アプリケーション情報でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法についてはUserSigの計算方法をご参照ください。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

logout

ログアウト。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ログアウトのコールバック。成功時にcodeは0になります。

setSelfProfile

個人情報の修正。

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像のアドレス。
callback	ActionCallback	個人情報設定のコールバック。成功時にcodeは0になります。

ルーム関連インターフェース関数

createRoom

ルームの作成（管理者が呼び出し）。

```
public abstract void createRoom(int roomId, TRTCVoiceRoomDef.RoomParam roomParam, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomIDを、1つのボイスチャットルームリストにまとめることができます。Tencent Cloudでは現在、ボイスチャットルームリストの管理サービスを行っていませんので、ご自身でボイスチャットルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報は、ルーム名、マイク情報、カバー情報など、ルームを説明するために用いる情報に使用します。マイク管理が必要な場合は、ルームのマイク数を記入する必要があります。
callback	ActionCallback	ルームの新規作成結果のコールバック。成功時にcodeは0になります。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

1. 管理者は、 `createRoom` を呼び出して新しいボイスチャットルームを作成します。この時にルームID、マイク・オンに管理者の確認の要否、マイク数などルームのプロパティ情報を渡します。
2. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
3. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
4. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

enterRoom

入室（リスナーが呼び出し）。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

リスナーが入室し聴取する際の通常の呼び出しプロセスは次のとおりです。

1. リスナーがサーバーから取得する最新のボイスチャットルームリストには、多くのボイスチャットルームのroomIdおよびルーム情報が含まれる場合があります。
   2. リスナーは1つのボイスチャットルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
2. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、マイク・オンの際の管理者への同意リクエストの要否の記録など。

©2013-2019 Tencent Cloud. All rights reserved.

3. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
4. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。

exitRoom

ルームから退出します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIdList	List&dxlt;Integer&dxgt;	ルームナンバーリスト。
callback	RoomInfoCallback	ルーム詳細情報のコールバック。

getUserInfoList

指定されたuserIdのユーザー情報を取得します。

```
public abstract void getUserInfoList(List<String> userIdList, TRTCVoiceRoomCallback.UserListCallback userlistcallback);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIdList	List&dxlt;String&dxgt;	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。
userlistcallback	UserListCallback	ユーザーの詳細情報のコールバック。

マイク管理インターフェース

enterSeat

ユーザーが発言者になります（リスナー側/管理者ともに呼び出し可）。

> 説明：
> マイク・オンの成功後、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。

```
public abstract void enterSeat(int seatIndex, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	マイク・オンの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、 `onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

moveSeat

マイクの移動 (マイク・オンするキャスター側で呼び出せます)。

> 説明：

©2013-2019 Tencent Cloud. All rights reserved.

マイクの移動に成功したら、ルーム内のすべてのメンバーが `onSeatListChange`、`onAnchorLeaveSeat`、`onAnchorEnterSeat` のイベント通知を受け取ります。(キャスターが呼び出した後に変更できるのは、マイクのシート番号情報のみで、ユーザーのキャスターロールを切り替えることはできません。)

```
public abstract int moveSeat(int seatIndex, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	移動の必要があるマイク番号。
callback	ActionCallback	操作コールバック。

戻り値：

戻り値	タイプ	意味
code	int	マイク移動操作の結果（0は成功、その他は失敗、10001はインターフェース呼び出し頻度制限）。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、`onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

leaveSeat

ユーザーが視聴者になる（キャスターが呼び出し）。

説明：
マイク・オフの成功後、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

```
public abstract void leaveSeat(TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

pickSeat

視聴者が発言できるように招待（管理者が呼び出し）。

> 説明：
> 、管理者が視聴者を発言できるように招待すると、ルーム内の全メンバーは、 `onSeatListChange` と `onAnchorEnterSeat` というイベント通知を受信します。

```
public abstract void pickSeat(int seatIndex, String userId, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	視聴者が発言できるように招待する必要があるマイク番号。
userId	String	ユーザーID。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。管理者がリスナーの同意がなければマイク・オンできないケースの場合は、まず `sendInvitation` を呼び出してからリスナーに申請し、 `onInvitationAccept` を受信すると、この関数をコールできるようになります。

kickSeat

キックアウトしてマイク・オフ（管理者が呼び出し）。

> 説明：
> 管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

```
public abstract void kickSeat(int seatIndex, TRTCVoiceRoomCallback.ActionCallback callback);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	キックアウトしてマイク・オフの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

muteSeat

任意のマイクのミュート/ミュート解除（管理者が呼び出し）。

> 説明：
> 任意のマイクのミュート/ミュート解除では、ルーム内の全メンバーが `onSeatListChange` および `onSeatMute` というイベント通知を受信します。

```
public abstract void muteSeat(int seatIndex, boolean isMute, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isMute	boolean	true：該当するマイクのミュート；false：該当するマイクのミュート解除。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。seatIndexの座席に該当するキャスターは、muteAudioを自動的に呼び出してミュート/ミュートオフにします。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。

> 説明：

©2013-2019 Tencent Cloud. All rights reserved.

管理者は、該当するマイクをクローズ/解除し、ルーム内の全メンバーは `onSeatListChange` および `onSeatClose` というイベント通知を受信します。

```
public abstract void closeSeat(int seatIndex, boolean isClose, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isClose	boolean	true：該当するマイクのクローズ； false：該当するマイクのクローズ解除。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。該当するseatIndexの座席上のキャスターはクローズされ、自動的にマイク・オフになります。

ローカル音声操作のインターフェース

startMicrophone

マイクの集音開始。

```
public abstract void startMicrophone();
```

stopMicrophone

マイクの集音停止。

```
public abstract void stopMicrophone();
```

setAudioQuality

音質の設定。

```
public abstract void setAudioQuality(int quality);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
quality	int	音声品質。詳細は TRTC SDKをご参照ください。

muteLocalAudio

ローカルの音声のミュート/ミュート取り消し。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	ミュート/ミュート取り消し。詳細は TRTC SDKをご参照ください。

setSpeaker

スピーカーの起動設定。

```
public abstract void setSpeaker(boolean useSpeaker);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
useSpeaker	boolean	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。

```
public abstract void setAudioCaptureVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、0 - 100、 デフォルト100。

setAudioPlayoutVolume

再生音量の設定。

©2013-2019 Tencent Cloud. All rights reserved.

```
public abstract void setAudioPlayoutVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	再生音量、0 - 100、 デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定ユーザーID。
mute	boolean	true：ミュート起動；false：ミュート停止。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動；false：ミュート停止。

setVoiceEarMonitorEnable

インイヤーモニタリングのオン/オフ。

```
public abstract void setVoiceEarMonitorEnable(boolean enable);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
enable	boolean	true：インイヤーモニタリングをオン。false：インイヤーモニタリングをオフ。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

バックグラウンド・サウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

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

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

招待シグナリング関連インターフェース

sendInvitation

ユーザーに招待を送信。

```
public abstract String sendInvitation(String cmd, String userId, String content, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	業務カスタマイズコマンド。
userId	String	招待ユーザーID。
content	String	招待コンテンツ。
callback	ActionCallback	送信結果のコールバック。

戻り値：

戻り値	タイプ	意味
inviteId	String	今回の招待IDの識別に使用。

acceptInvitation

招待の同意。

```
public abstract void acceptInvitation(String id, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

rejectInvitation

招待の拒否。

```
public abstract void rejectInvitation(String id, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

cancelInvitation

招待の取り消し。

```
public abstract void cancelInvitation(String id, TRTCVoiceRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

TRTCVoiceRoomDelegateイベントのコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

説明：
SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラーメッセージ。

onWarning

警告のコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	警告メッセージ。

onDebugLog

Logコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	ログ情報。

ルームイベントのコールバック

©2013-2019 Tencent Cloud. All rights reserved.

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	String	ルームID。

onRoomInfoChange

入室に成功後、このインターフェースをコールバックします。roomInfoの情報は、管理者がルームを作成するときに渡されます。

```
void onRoomInfoChange(TRTCVoiceRoomDef.RoomInfo roomInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomInfo	RoomInfo	ルーム情報。

onUserMicrophoneMute

ユーザーのマイクがミュートになっているかどうかについて、ユーザーがmuteLocalAudioを呼び出すと、ルームの他のユーザーがこの通知を受信します。

```
void onUserMicrophoneMute(String userId, boolean mute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
mute	boolean	true：マイクミュート； false：ミュート解除。

onUserVolumeUpdate

音量レベルリマインダを有効にして、各メンバーの音量を通知します。

```
void onUserVolumeUpdate(List<TRTCCloudDef.TRTCVolumeInfo> userVolumes, int totalVolume);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userVolumes	ListList<trtcclouddef.trtcvolumeinfo>	ユーザーリスト。
totalVolume	int	音量の大きさ。値：0～100。

マイクコールバック

onSeatListChange

全量のマイクリストの変更は、全てのマイクリストを含みます。

```
void onSeatListChange(List<SeatInfo> seatInfoList);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatInfoList	List&dxlt;SeatInfo&dxgt;	全量のマイクリスト。

onAnchorEnterSeat

発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。

```
void onAnchorEnterSeat(int index, TRTCVoiceRoomDef.UserInfo user);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	メンバーがマイク・オンのマイク。
user	UserInfo	マイク・オンのユーザーの詳細情報。

onAnchorLeaveSeat

視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。

```
void onAnchorLeaveSeat(int index, TRTCVoiceRoomDef.UserInfo user);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
index	int	マイク・オフのマイク。
user	UserInfo	マイク・オンのユーザーの詳細情報。

onSeatMute

管理者のマイクミュート。

```
void onSeatMute(int index, boolean isMute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	操作するマイク。
isMute	boolean	true：マイクミュート； false：ミュート解除。

onSeatClose

管理者のマイククローズ。

```
void onSeatClose(int index, boolean isClose);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	操作するマイク。
isClose	boolean	true：マイクのクローズ； false：マイクのクローズ解除。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。

```
void onAudienceEnter(TRTCVoiceRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userInfo	UserInfo	入室したリスナーの情報。

onAudienceExit

リスナー退室通知の受信。

```
void onAudienceExit(TRTCVoiceRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	UserInfo	退室したリスナーの情報。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

```
void onRecvRoomTextMsg(String message, TRTCVoiceRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージの受信。

```
void onRecvRoomCustomMsg(String cmd, String message, TRTCVoiceRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

招待シグナリングイベントのコールバック

onReceiveNewInvitation

新規招待リクエストの受信。

```
void onReceiveNewInvitation(String id, String inviter, String cmd, String content);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
inviter	String	招待者のユーザーID。
cmd	String	業務指定のコマンドワードは、開発者がカスタマイズします。
content	String	業務指定のコンテンツ。

onInviteeAccepted

被招待者が招待に同意。

```
void onInviteeAccepted(String id, String invitee);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInviteeRejected

©2013-2019 Tencent Cloud. All rights reserved.

被招待者による招待の拒否。

```
void onInviteeRejected(String id, String invitee);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInvitationCancelled

招待者が招待を取り消し。

```
void onInvitationCancelled(String id, String inviter);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
inviter	String	招待者のユーザーID。
</trtcclouddef.trtcvolumeinfo>		

©2013-2019 Tencent Cloud. All rights reserved.

多人数ビデオミーティング
多人数ビデオミーティング(iOS)

最終更新日：：2022-03-02 11:30:43

デモンストレーション

当社のAppをダウンロードしてインストールし、多人数ビデオ会話の効果をご体験いただけます。これには、画面共有、美顔、低遅延ビデオ通話など、TRTCの多人数オーディオビデオシーン関連の機能が含まれています。

ソリューションの優位性

- 超低遅延のオーディオビデオ通話、画面共有、美顔などの機能を統合し、多人数オーディオビデオルームの一般的な機能をカバーします。
- 二次開発の需要に応じて、カスタムUIやレイアウトを迅速に実装し、ビジネスの迅速な立ち上げに役立ちます。
- TRTCとIMの基本SDKをカプセル化し、基本的なロジックコントロールを実装し、呼び出しを容易にするインターフェースを提供します。

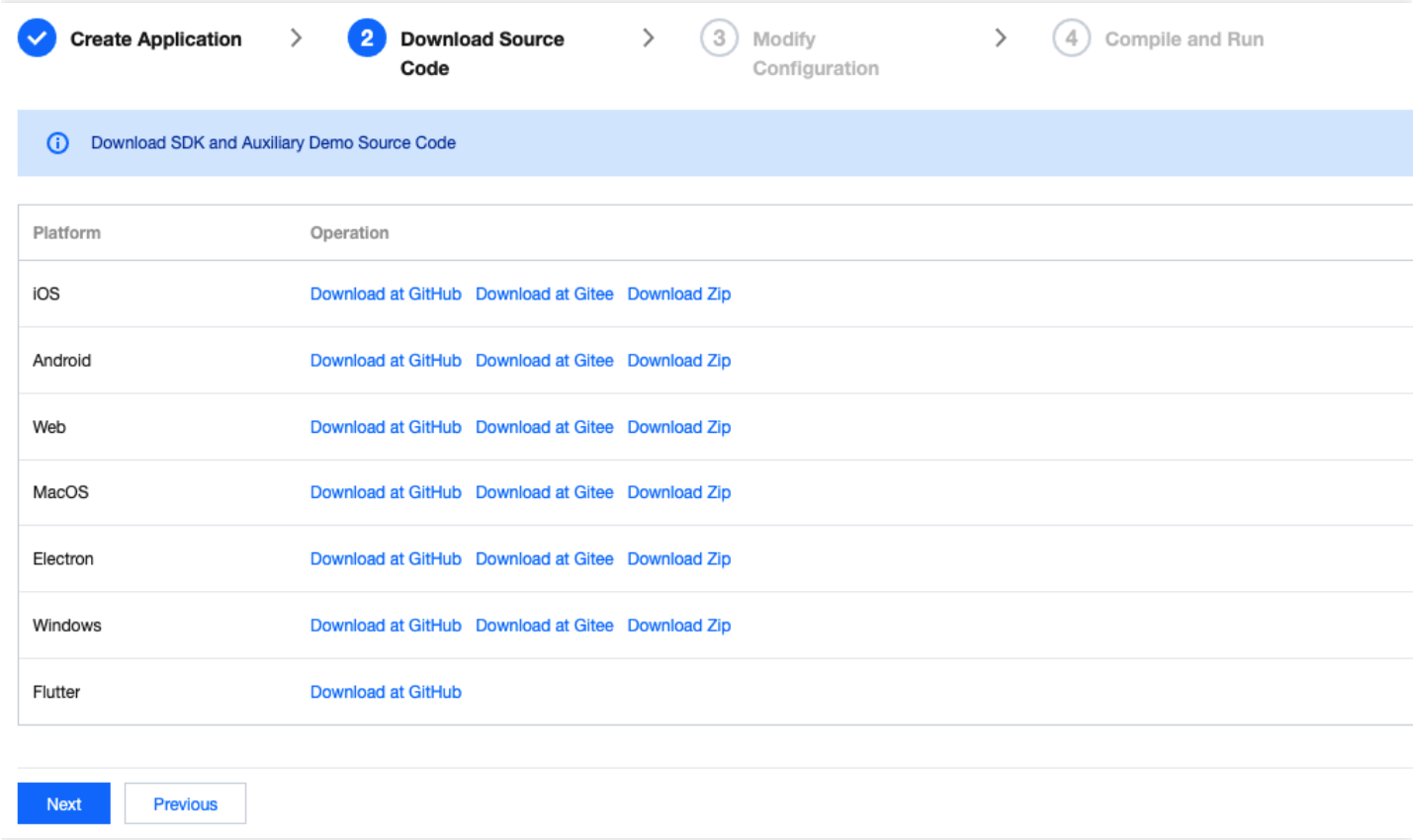
アクセスガイド

多人数オーディオビデオルームの機能に素早くアクセスしたい場合は、当社が提供するAppを直接修正して適応させるか、App内のModuleを使用してカスタマイズしたUIを実装することができます。

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援**>Demoクイックスタートを選択します。
2. `TestTUIRoom` などのアプリケーション名を入力して、**作成**をクリックします。
3. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。

©2013-2019 Tencent Cloud. All rights reserved.



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUIRoomに進み、ソースコードをCloneまたはダウンロードします。

手順3：Appプロジェクトファイルの設定

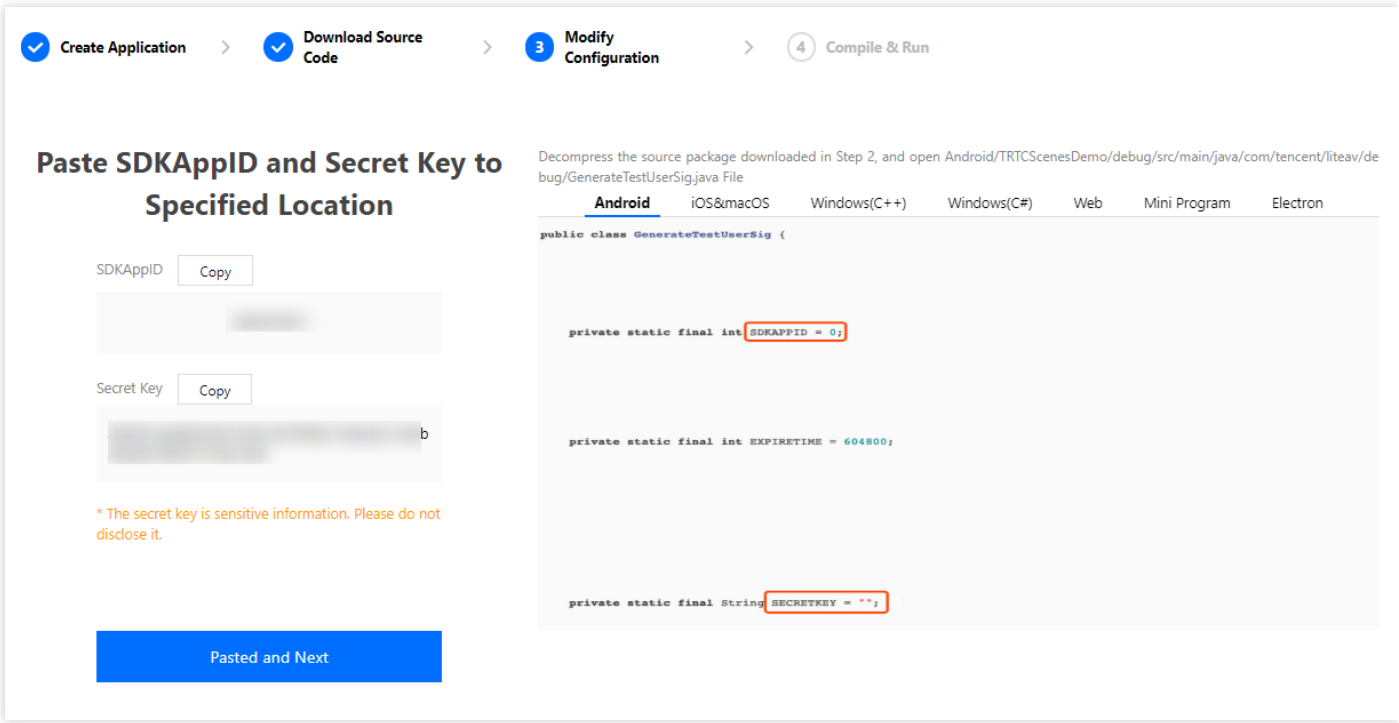
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `Example/Debug/GenerateTestUserSig.swift` ファイルを見つけて開きます。

3. `GenerateTestUserSig.swift` ファイル内の関連パラメータを設定します。

   - SDKAPPID：デフォルトは0。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトは空文字列。実際のキー情報を設定してください。

©2013-2019 Tencent Cloud. All rights reserved.



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

説明：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：プロジェクトの実行

Xcode（11.0およびそれ以降のバージョン）を使用してソースコードプロセス `Example/DemoApp.xcworkspace` を開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

手順5：プロジェクトソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードの `Source` にはUIフォルダが含まれ、このフォルダはいずれもUIに関連するコードです。以下の表には各フォルダに対応するUIをリストアップし、二次調整に役立つようにしています。

フォルダ	機能説明
TUIRoomEnter	TUIRoomエントリーに関連する実装コードです。 TUIRoomEntranceViewControllerクラスはPodsを外部に公開するパブリッククラスです。
TUIRoomMain	TUIRoomメインインターフェースに関連するUI実装コードです。このクラスはPodsプライベートクラスです。
TUIRoomMemberList	メンバーリストインターフェースに関連するUI実装コードです。このクラスはPodsプライベートクラスです。
TUIRoomSet	インターフェース設定に関連するUI実装コードです。このクラスはPodsプライベートクラスです。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

エントリーインターフェース

図のように、**ルームの作成**または**ルームへの参加**を選択してください。

ルーム作成ページ

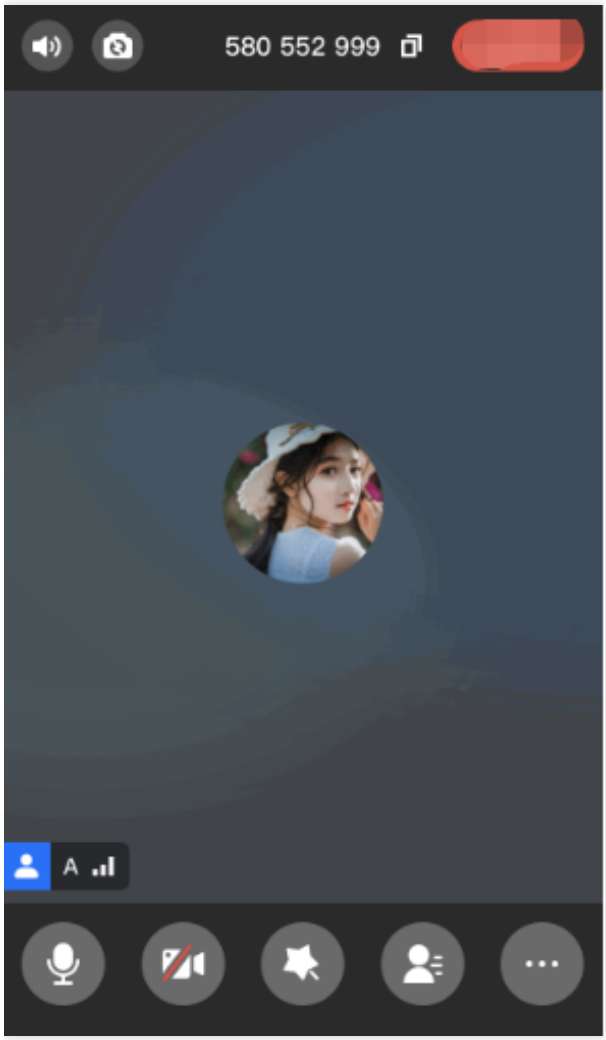
ユーザーAが作成し、ルームナンバーはデフォルトで発行されます。**ルーム作成**をクリックすると、メインページに進むことができます。

ルーム参加ページ

ユーザーBはユーザーAのルームナンバーを入力し、**ルームへの参加**をクリックすると、メインページに進むことができます。

メインページ（ユーザーA）

©2013-2019 Tencent Cloud. All rights reserved.



配信側リスト

配信側リストには現在のルームのメンバーを表示できます。相手がカメラとマイクを起動すると、相手の画面を見ること、相手の声を聞くことができます。

上部のコントロールバー

イヤホン/マイク切り替え、フロント/リアカメラ切り替え、ルーム情報、退出の機能を実装しました。

下部のツールバー

自身のマイク/カメラの制御、美顔、メンバーリスト、設定などの機能を実装しました。

美顔

ライブストリーミング中に画面に対し美顔およびエフェクト表示を行うことができます。

設定ウィンドウ

オーディオビデオ関連パラメータを設定でき、**画面共有**の開始をサポートします。

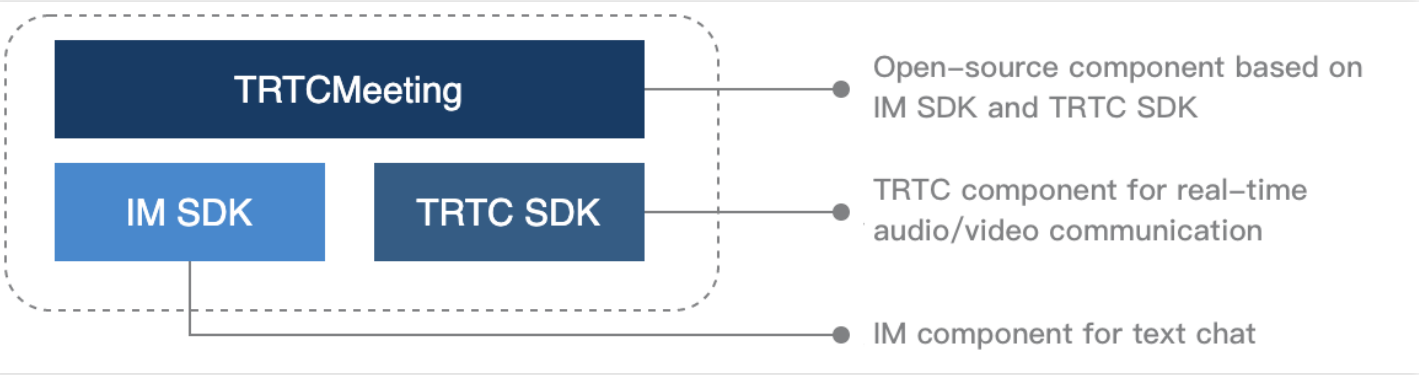
©2013-2019 Tencent Cloud. All rights reserved.

退室

- **キャスター**：ルームを解散し、全員を退室させます。
- **キャスター以外**：自ら退室します。

UIカスタマイズの実装

ソースコードの `Source` フォルダにはUIとPresenterが含まれ、Presenterフォルダには再利用できるオープンソースコンポーネントTUIRoomCoreが含まれています。 `TUIRoomCore.h` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタマイズしたUIを実装することができます。



手順1：SDKの統合

多人数オーディオビデオコンポーネントTUIRoomは、TRTC SDKとIM SDKに依存しています。次の手順で2つのSDKをプロジェクトに統合することができます。

- **方法1：cocoapodsリポジトリを介する依存**

```
pod 'TXIMSDK_iOS'
pod 'TXLiteAVSDK_TRTC'
```

説明：
2つのSDK製品の最新バージョン番号は、TRTCおよびIMのGithubトップページで取得することができます。

- **方法2：ローカルを介する依存**
  開発環境でのcocoapodsリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

©2013-2019 Tencent Cloud. All rights reserved.

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定

`info.plist` ファイルに `Privacy > Camera Usage Description` と `Privacy > Microphone Usage Description` を追加し、カメラとマイクの権限をリクエストする必要があります。

手順3：TUIRoomコンポーネントのインポート

cocoapodsを使用してコンポーネントをインポートすることができます。具体的な手順は、以下のとおりです。

1. プロジェクトディレクトリの `Source` 、 `Resources` 、 `TCBeautyKit` 、 `TXAppBasic` フォルダ、 `TUIRoom.podspec` ファイルをお客様のプロジェクトディレクトリ下へコピーします。
2. `Podfile` ファイル内に以下の依存関係を追加します。その後、 `pod install` コマンドを実行すると、インポートが完了します。

```
# ローカル依存ライブラリ
def local
pod 'TXAppBasic', :path => "../../TXAppBasic/"
pod 'TCBeautyKit', :path => "../../TCBeautyKit/"
pod 'TXLiteAVSDK_TRTC',:path => "../../../SDK/"
end
def pod_local(type)
loadLocalPod('TUIRoom', type)
end
def loadLocalPod(name, type)
pod "#{name}/#{type}", :path => "../"
end
target 'DemoApp' do
local
pod_local('TRTC')
end
```

手順4：コンポーネントの作成およびログイン

1. TUICoreのTUILoginを呼び出してログインします。次の例をご参照ください。

```
TUILogin.initWithSdkAppID(Int32(SDKAPPID))
let userID = ProfileManager.shared().curUserID()
```

©2013-2019 Tencent Cloud. All rights reserved.

```
let userSig = GenerateTestUserSig.genTestUserSig(userID)
TUILogin.login(userID, userSig: userSig, succ: {
debugPrint("login success")
}, fail: { (code, errorDes) in
debugPrint("login failed, code:¥(code), error: ¥(errorDes ?? "nil")")
})
```

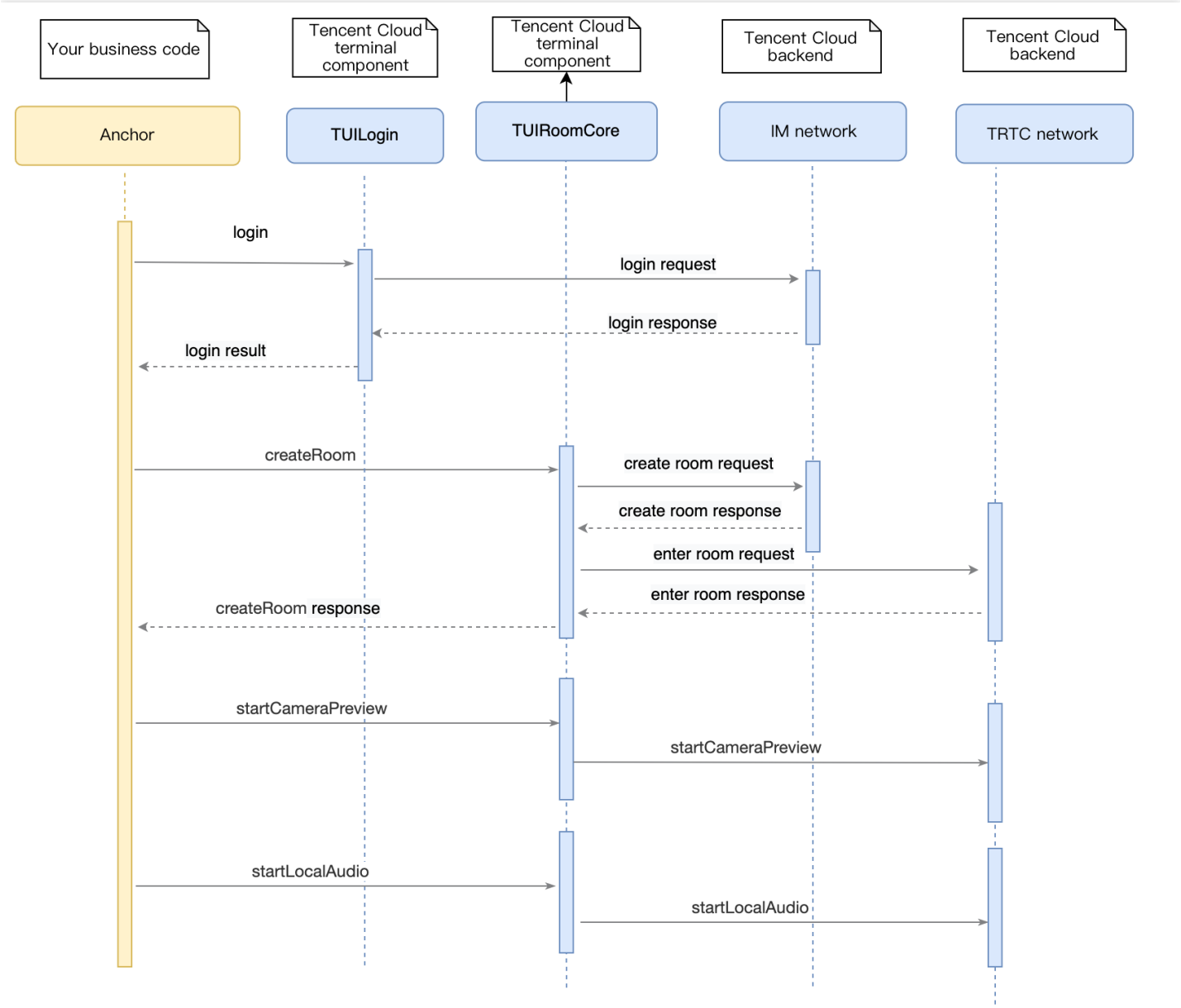
2. 正常にログインした後、TUIRoomCoreを呼び出してルームを作成することができます。

```
let roomId = 123
TUIRoomCore.shareInstance().createRoom("¥(roomId)",speechMode: .freeSpeech,callback: { [weak s
elf] code, message in
if code == 0 {
debugPrint("create room success")
}else{
}
})
```

3. 作成に成功し、メインページに進みます。

```
let vc = TUIRoomMainViewController(roomId: roomId, isVideoOn: openCameraSwitch.isOn, isAudioO
n: openMicSwitch.isOn)
TUIRoomCore.shareInstance().setDelegate(vc)
navigationController?.pushViewController(vc, animated: true)
```

©2013-2019 Tencent Cloud. All rights reserved.



ステップ5：メンバーの入室

1. メンバーはTUICoreのTUILoginを呼び出してログインします。次の例をご参照ください。

```
TUILogin.initWithSdkAppID(Int32(SDKAPPID))
let userID = ProfileManager.shared().curUserID()
let userSig = GenerateTestUserSig.genTestUserSig(userID)
TUILogin.login(userID, userSig: userSig, succ: {
debugPrint("login success")
}, fail: { (code, errorDes) in
debugPrint("login failed, code:¥(code), error: ¥(errorDes ?? "nil")")
})
```

©2013-2019 Tencent Cloud. All rights reserved.

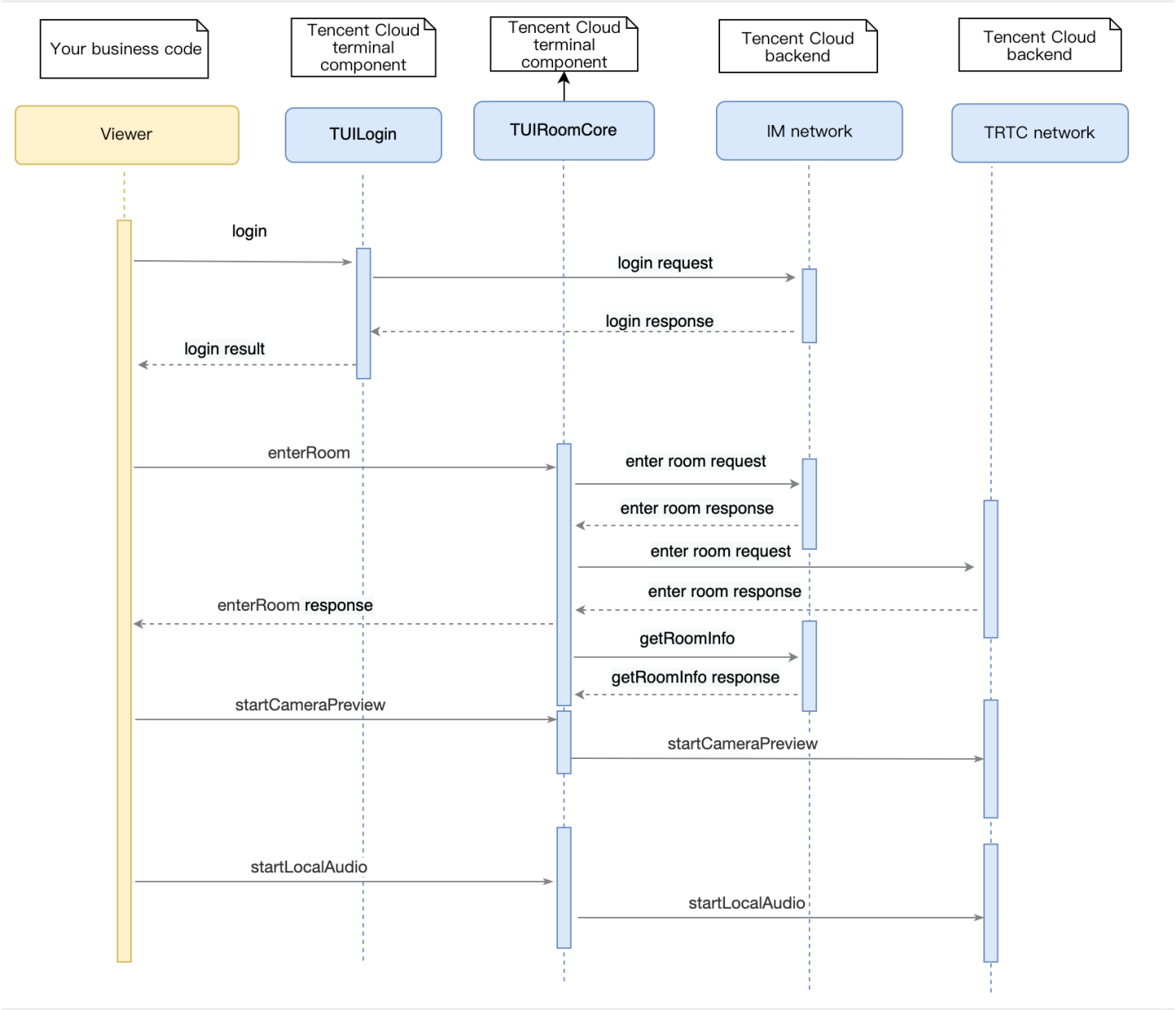
2. 正常にログインした後、TUIRoomCrreを呼び出して入室できます。

```
let roomId = 123
TUIRoomCore.shareInstance().enterRoom("¥(roomId)", callback: { [weak self] code, message in
if code == 0 {
debugPrint("enter room success")
}else{
}
})
```

3. 入室に成功した後、再びメインページに進みます。

```
let vc = TUIRoomMainViewController(roomId: roomId, isVideoOn: openCameraSwitch.isOn, isAudioO
n: openMicSwitch.isOn)
TUIRoomCore.shareInstance().setDelegate(vc)
navigationController?.pushViewController(vc, animated: true)
```

©2013-2019 Tencent Cloud. All rights reserved.



手順6：画面共有

1. TUIRoomCoreの `startScreenCapture` を呼び出して共有します。
2. ルーム内のその他メンバーは `onRemoteUserScreenVideoAvailable` のイベント通知を受信します。

```
// ボタンのクリックで画面共有を実現します
if #available(iOS 12.0, *) {
// 画面共有
let params = TRTCVideoEncParam()
params.videoResolution = TRTCVideoResolution._1280_720
params.resMode = TRTCVideoResolutionMode.portrait
params.videoFps = 10
params.enableAdjustRes = false
```

©2013-2019 Tencent Cloud. All rights reserved.

```
params.videoBitrate = 1500
TUIRoomCore.shareInstance().startScreenCapture(params)
TUIRoomBroadcastExtensionLauncher.launch()
}else{
view.window?.makeToast(.versionLowToastText)
}
```

ステップ7：退室

- **キャスター**はdestoryRoomインターフェースを呼び出してルームを解散し、IMグループチャットを解散し、TRTCルームを退室します。 メンバー側は、グループを解散し、TRTCルームを退室するよう通知するonDestroyRoomコールバックメッセージを受信します。
- **メンバー**はleaveRoomインターフェースを呼び出してルームを解散し、IMグループチャットから退出し、TRTCルームを退室します。 その他のメンバー側は、メンバーが退室したことを通知するonRemoteUserLeaveコールバックメッセージを受信します。

```
if isHomeowner {
TUIRoomCore.shareInstance().destroyRoom { [weak self] _, _ in
guard let self = self else { return }
self.navigationController?.popViewController(animated: true)
}
}else{
TUIRoomCore.shareInstance().leaveRoom { [weak self] _, _ in
guard let self = self else { return }
self.navigationController?.popViewController(animated: true)
}
}
```

©2013-2019 Tencent Cloud. All rights reserved.

多人数ビデオミーティング(Android)

最終更新日：：2022-03-14 16:07:21

デモンストレーション

当社のAppをダウンロードしてインストールし、多人数オーディオビデオ通話の効果をご体験いただけます。これには、画面共有、美顔、低遅延ビデオ通話など、TRTCの多人数オーディオビデオシーン関連の機能が含まれています。

ソリューションの優位性

- 超低遅延のオーディオビデオ通話、画面共有、美顔などの機能を統合し、多人数オーディオビデオルームの一般的な機能をカバーします。
- 二次開発の需要に応じて、カスタムUIやレイアウトを迅速に実装し、ビジネスの迅速な立ち上げに役立ちます。
- TRTCとIMの基本SDKをカプセル化し、基本的なロジックコントロールを実装し、呼び出しを容易にするインターフェースを提供します。

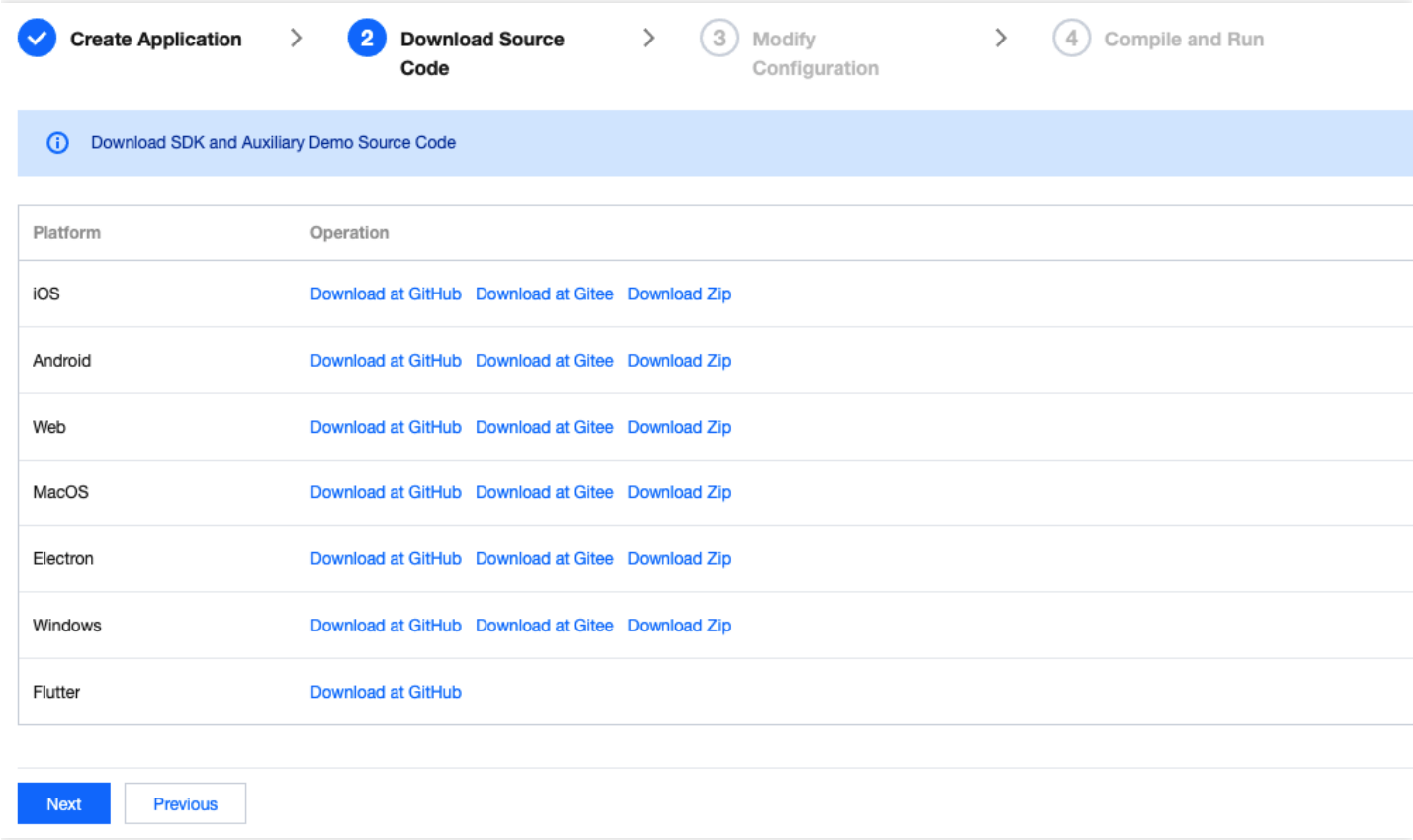
アクセスガイド

多人数オーディオビデオルームの機能に素早くアクセスしたい場合は、当社が提供するAppを直接修正して適応させるか、App内のModuleを使用してカスタマイズしたUIを実装することができます。

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援** > **Demoクイックスタート**を選択します。
2. `TestTUIRoom` などのアプリケーション名を入力して、**作成**をクリックします。
3. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。

©2013-2019 Tencent Cloud. All rights reserved.



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期してアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUIRoomに進み、ソースコードをCloneまたはダウンロードします。

手順3：Appプロジェクトファイルの設定

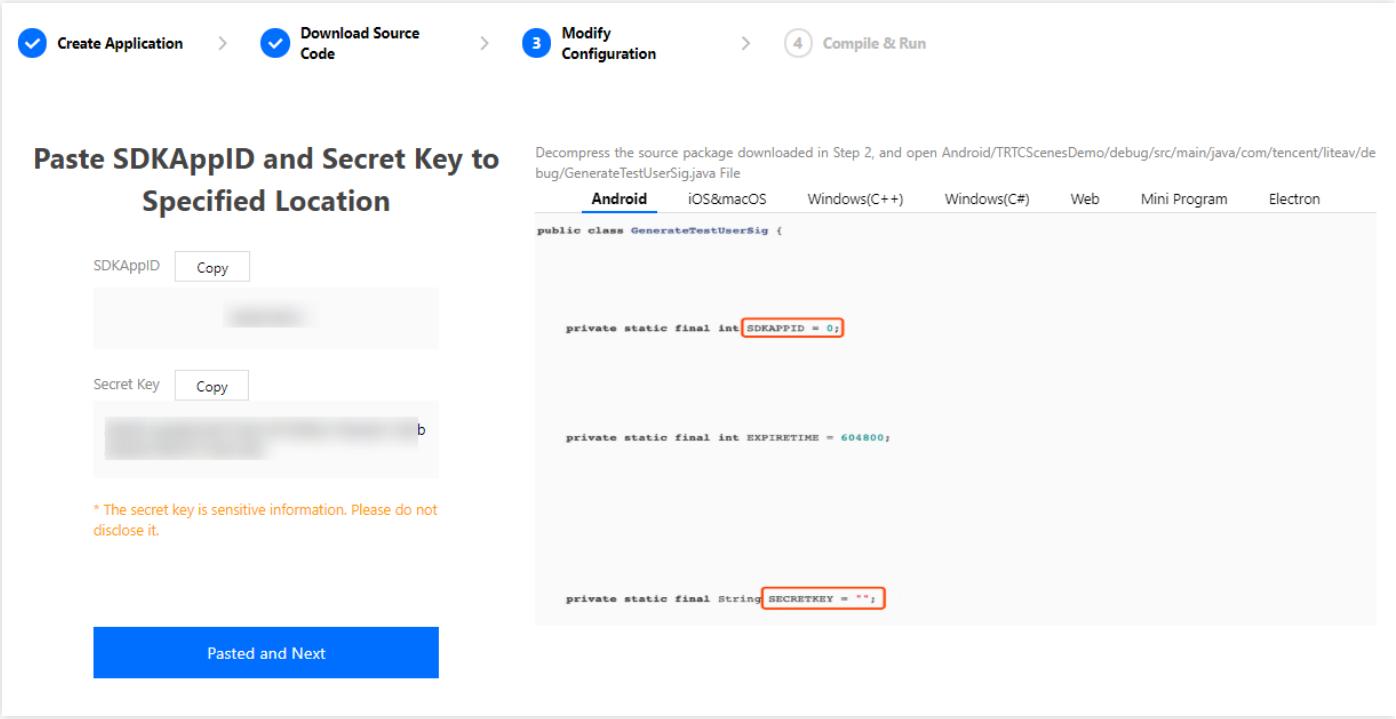
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `Android/Debug/src/main/java/com/tencent/liteav/debug/GenerateTestUserSig.java` のファイルを見つけて開きます。

3. `GenerateTestUserSig.java` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはプレースホルダー(PLACEHOLDER)。実際のSDKAppIDを設定してください。

©2013-2019 Tencent Cloud. All rights reserved.

- SECRETKEY：デフォルトはプレースホルダー(PLACEHOLDER)。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

説明：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Android Studio（バージョン3.5以上）を使用してソースコード `TUIRoom` プロジェクトを開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

手順5：プロジェクトソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードの `Source` フォルダには、uiとmodelの2つのサブフォルダが含まれ、uiフォルダの中はいずれもインターフェースコードです。以下の表にはファイルまたはフォルダおよび対応するUIをリストアップし、二次調整に役立つようにしています。

ファイルまたはフォルダ	機能説明
remote	リモートユーザーのリストのインターフェース
widget	一般的なUIコンポーネント
CreateRoomActivity.java	オーディオビデオルーム作成のインターフェース
RoomMainActivity.java	オーディオビデオルームのメインインターフェース
RoomVideoView.java	TRTCのTXCloudVideoViewをパッケージ化し、自分とリモートユーザーのビデオデータの表示に利用します
MemberEntity.java	UI層のユーザーデータ
MemberListAdapter.java	オーディオビデオルームのメインインターフェースのAdapter

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

エントリーインターフェース

図のように、「ルームの作成」または「ルームへの参加」を選択してください。

ルーム作成ページ

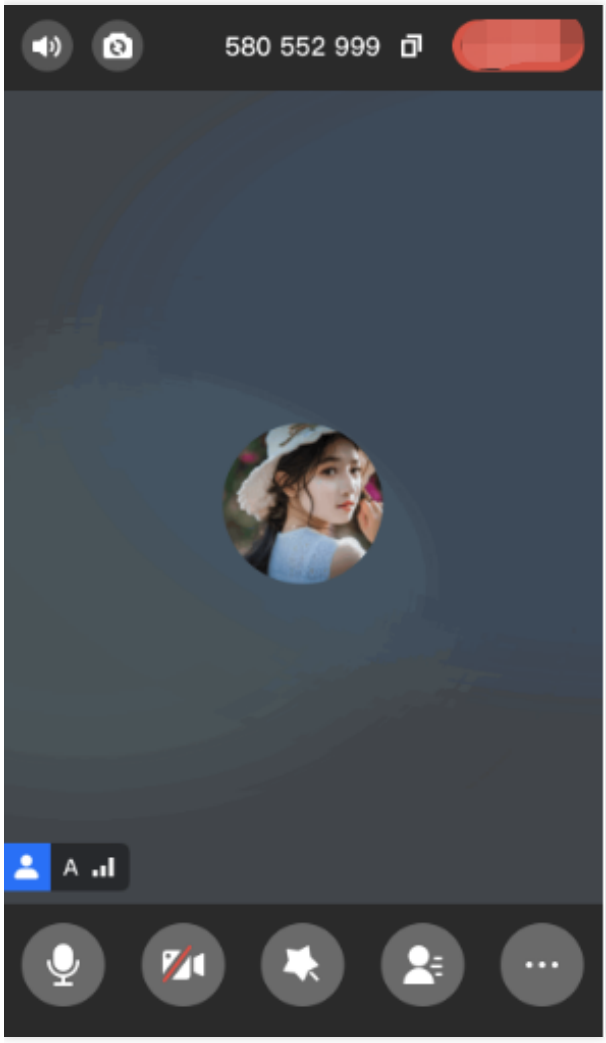
ユーザーAが作成し、ルームナンバーはデフォルトで発行されます。**ルーム作成**をクリックすると、メインページに進むことができます。

ルーム参加ページ

ユーザーBはユーザーAのルームナンバーを入力し、**ルームへの参加**をクリックすると、メインページに進むことができます。

メインページ（ユーザーA）

©2013-2019 Tencent Cloud. All rights reserved.



配信側リスト

配信側リストには現在のルームのメンバーを表示できます。相手がカメラとマイクを起動すると、相手の画面を見ること、相手の声を聞くことができます。

上部のコントロールバー

イヤホン/マイク切り替え、フロント/リアカメラ切り替え、ルーム情報、退出の機能を実装しました。

下部のツールバー

自身のマイク/カメラの制御、美顔、メンバーリスト、設定などの機能を実装しました。

美顔

ライブストリーミング中に画面に対し美顔およびエフェクト表示を行うことができます。

設定ウィンドウ

オーディオビデオ関連パラメータを設定でき、**画面共有**の開始をサポートします。

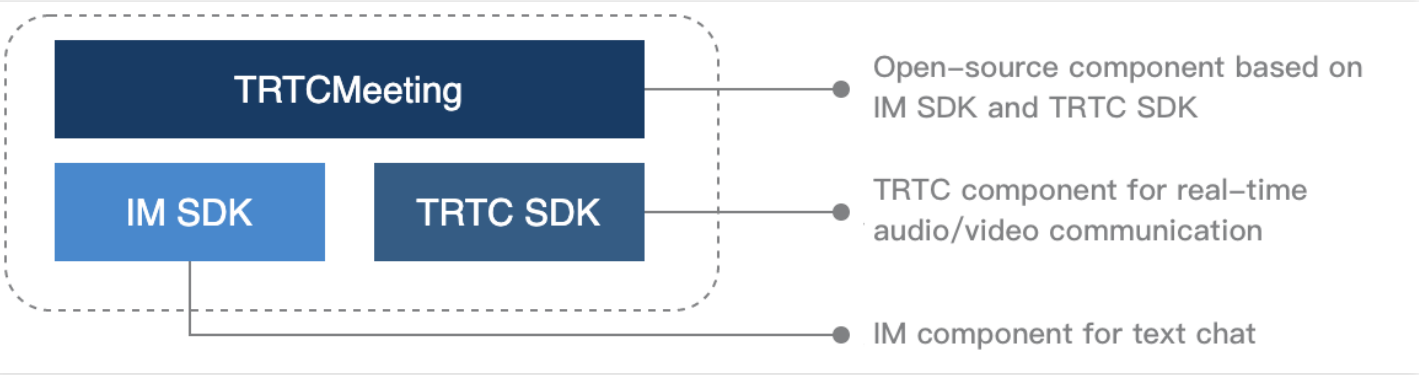
©2013-2019 Tencent Cloud. All rights reserved.

退室

- **キャスター**：ルームを解散し、全員を退室させます。
- **キャスター以外**：自ら退室します。

UIカスタマイズの実装

ソースコードの `Source` フォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTUIRoomCoreが含まれています。 `TUIRoomCore.java` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタマイズしたUIを実装することができます。



手順1：SDKの統合

多人数オーディオビデオルームのコンポーネントTUIRoomCoreは、TRTC SDKとIM SDKに依存しています。次の手順で2つのSDKをプロジェクトに統合することができます。

- **方法1：Mavenリポジトリを介する依存**
  i. dependenciesにTRTCSDKとIMSDKの依存を追加します。

```
dependencies {
compile "com.tencent.liteav:LiteAVSDK_TRTC:latest.release"
compile 'com.tencent.imsdk:imsdk:latest.release'
}
```

2. defaultConfigでAppが使用するCPUアーキテクチャを指定します。

```
defaultConfig {
ndk {
abiFilters "armeabi-v7a"
```

©2013-2019 Tencent Cloud. All rights reserved.

```
}
}
```

3. **Sync Now**をクリックし、自動でSDKをダウンロードし、プロジェクトに統合します。

- **方法2：ローカルAARを介する依存**
  開発環境でのMavenリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定および難読化ルール

1. `AndroidManifest.xml` にAppの権限を設定します。SDKには次の権限が必要です（6.0以上のAndroidシステムではカメラ、ストレージ読み取りの権限を動的に申請する必要があります）。

```
<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. proguard-rules.proファイルでは、SDK関連クラスを非難読化リストに追加します。

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

ステップ3：TUIRoomCoreコンポーネントのインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
src/main/java/com/tencent/liteav/tuiroom/model
```

©2013-2019 Tencent Cloud. All rights reserved.

手順4：コンポーネントの作成およびログイン

TUICoreのTUILoginを呼び出してログインします。次の例をご参照ください。

```
TUILogin.init(this, "sdkAppid", null, new V2TIMSDKListener() {
@Override
public void onKickedOffline() {
}
@Override
public void onUserSigExpired() {
}
});
TUILogin.login("userId", "userSig", new V2TIMCallback() {
@Override
public void onError(int code, String msg) {
}
@Override
public void onSuccess() {
}
});
```

ステップ5：ニックネームの設定

正常にログインした後、setSelfProfileを呼び出し、ユーザー情報を設定します。

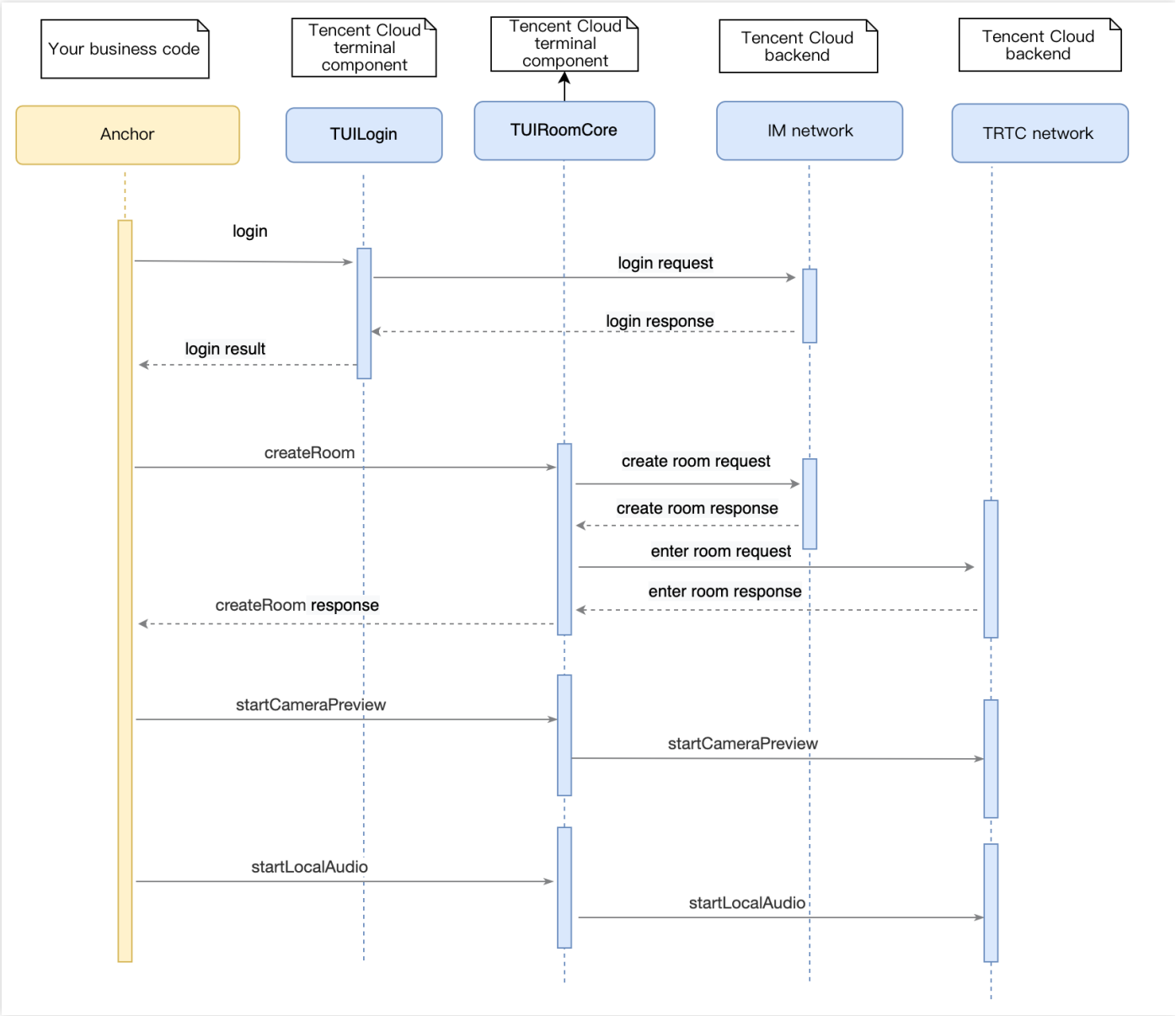
ステップ6：ルームの作成

1. メンバーがcreateRoomインターフェースを呼び出し、新たなルームを作成します。ルームの作成後、キャスターとしてチャットルームとTRTCルームを含むルームに入室します。

```
TUIRoomCore tuiRoomCore = TUIRoomCore.getInstance(context);
tuiCore.setListener(listener);
tuiCore.createRoom("roomId", TUIRoomCoreDef.SpeechMode.FREE_SPEECH,
new TUIRoomCoreCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
// ルーム作成の成功
}else{
}
}
});
```

©2013-2019 Tencent Cloud. All rights reserved.

2. startCameraPreviewインターフェースを呼び出し 、ローカルビデオのキャプチャと表示を開始します。



ステップ7：ルームの解散

- キャスターがdestoryRoomインターフェースを呼び出し 、ルームを解散します。IMグループチャットを解散して、TRTCルームを退室します。
- メンバー側は、グループを解散し、TRTCルームを退室するよう通知するonDestroyRoomコールバックメッセージを受信します。

ステップ8：ルームへの参加

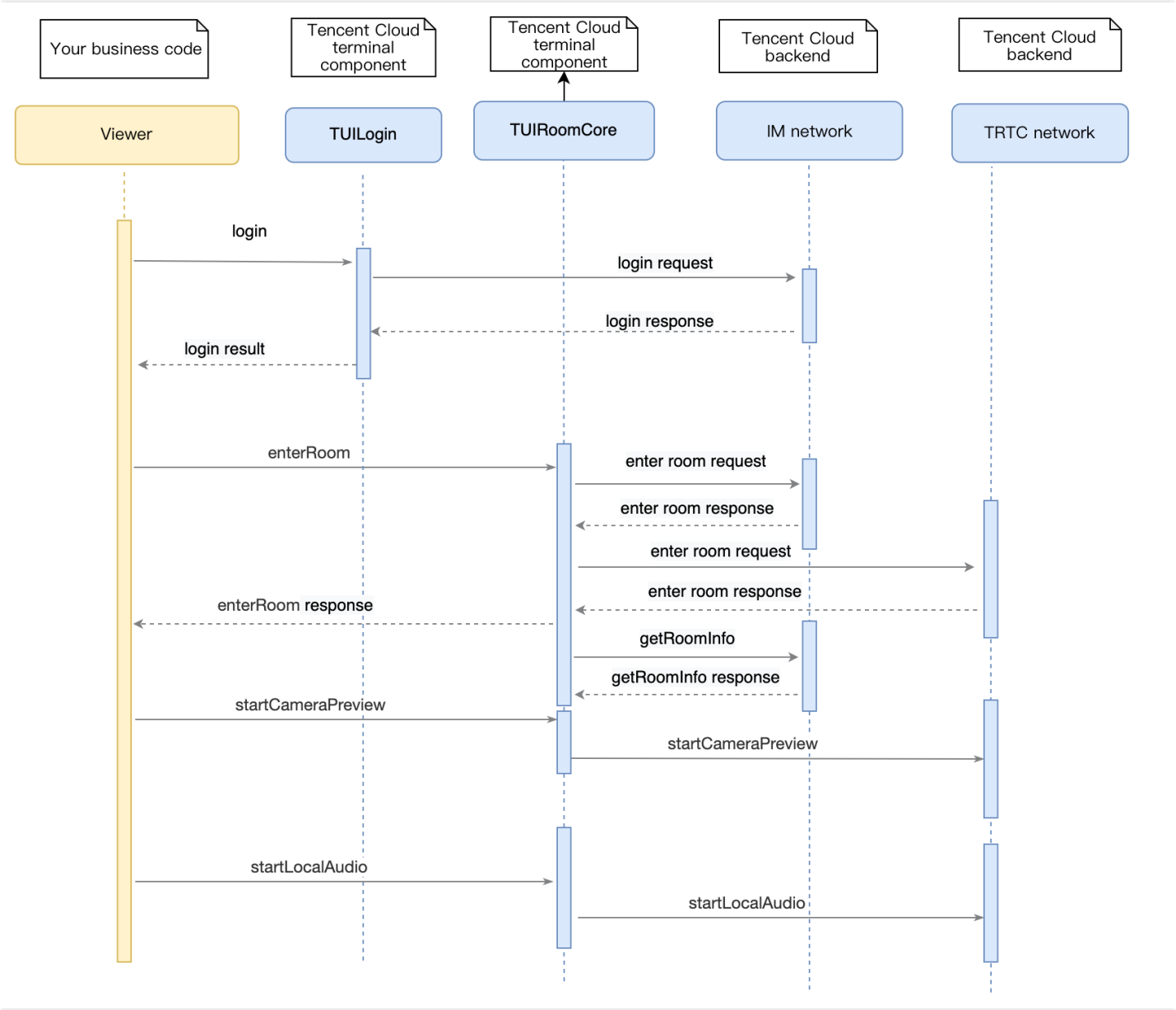
1. ルーム参加手順は、作成手順と基本的に同じです。メンバーはenterRoomインターフェースを呼び出して入室する必要があります。

©2013-2019 Tencent Cloud. All rights reserved.

2. その他のメンバーは、メンバーの入室があったことを通知するonRemoteUserEnterコールバックを受信します。

```
TUIRoomCore tuiRoomCore = TUIRoomCore.getInstance(context);
tuiCore.setListener(listener);
tuiCore.enterRoom("roomId", new TUIRoomCoreCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
// 入室に成功
}else{
}
}
});
```

©2013-2019 Tencent Cloud. All rights reserved.



ステップ9：退室

1. メンバーがleaveRoomインターフェースを呼び出し、IMルームとTRTCルームを退室します。
2. その他メンバーグループ側は、メンバーが退室したことを通知するonRemoteUserLeaveコールバックを受信します。

手順10：画面共有

1. 画面共有機能は、システムにフローティングウィンドウの権限を申請する必要があり、自分でUIの中で特定のロジックを実装する必要があります。
2. TUIRoomCoreの `startScreenCapture` を呼び出して共有を行います。エンコードパラメータとスクリーンキャプチャのプロセスのフローティングウィンドウを渡すと、画面共有機能を実装できます。具体的な情報につ

©2013-2019 Tencent Cloud. All rights reserved.

いては、TRTC SDKをご参照ください。

3. ルーム内のその他メンバーはonRemoteUserScreenVideoAvailableのイベント通知を受信します。

> 注意：
> 画面共有とカメラキャプチャは相互排他的な操作です。画面共有機能をアクティブにしたい場合は、先にstopCameraPreviewを呼び出してカメラキャプチャを停止してください。

```
// 1.AndroidManifest.xmlのファイルの中にSDKのスクリーンレコーディング機能のactivityと権限を追加します
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW">
<application>
<activity android:name="com.tencent.rtmp.video.TXScreenCapture$TXScreenCaptureAssistantActivity"
android:theme="@android:style/Theme.Translucent">
</activity></application>
```

```
// 2.自分のインターフェースの中でフローティングウィンドウの権限を申請します
if (Build.VERSION.SDK_INT >= 23) {
if (!Settings.canDrawOverlays(getApplicationContext())) {
Intent intent = new Intent(Settings.ACTION_MANAGE_OVERLAY_PERMISSION, Uri.parse("package:" + getP
ackageName()));
startActivityForResult(intent, 100);
}else{
startScreenCapture();
}
}else{
startScreenCapture();
}

// 3.システムのコールバックの結果
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
if (requestCode == 100) {
if (Build.VERSION.SDK_INT >= 23) {
if (Settings.canDrawOverlays(this)) {
// ユーザーの権限承認が成功
startScreenCapture();
}else{
}
}
}
}

// 4.画面共有の起動
private void startScreenCapture() {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
TRTCCloudDef.TRTCVideoEncParam encParams = new TRTCCloudDef.TRTCVideoEncParam();
encParams.videoResolution = TRTCCloudDef.TRTC_VIDEO_RESOLUTION_1280_720;
encParams.videoResolutionMode = TRTCCloudDef.TRTC_VIDEO_RESOLUTION_MODE_PORTRAIT;
encParams.videoFps = 10;
encParams.enableAdjustRes = false;
encParams.videoBitrate = 1200;

TRTCCloudDef.TRTCScreenShareParams params = new TRTCCloudDef.TRTCScreenShareParams();
mTUIRoom.stopCameraPreview();
mTUIRoom.startScreenCapture(encParams, params);
}
```

©2013-2019 Tencent Cloud. All rights reserved.

多人数ビデオミーティング(Flutter)

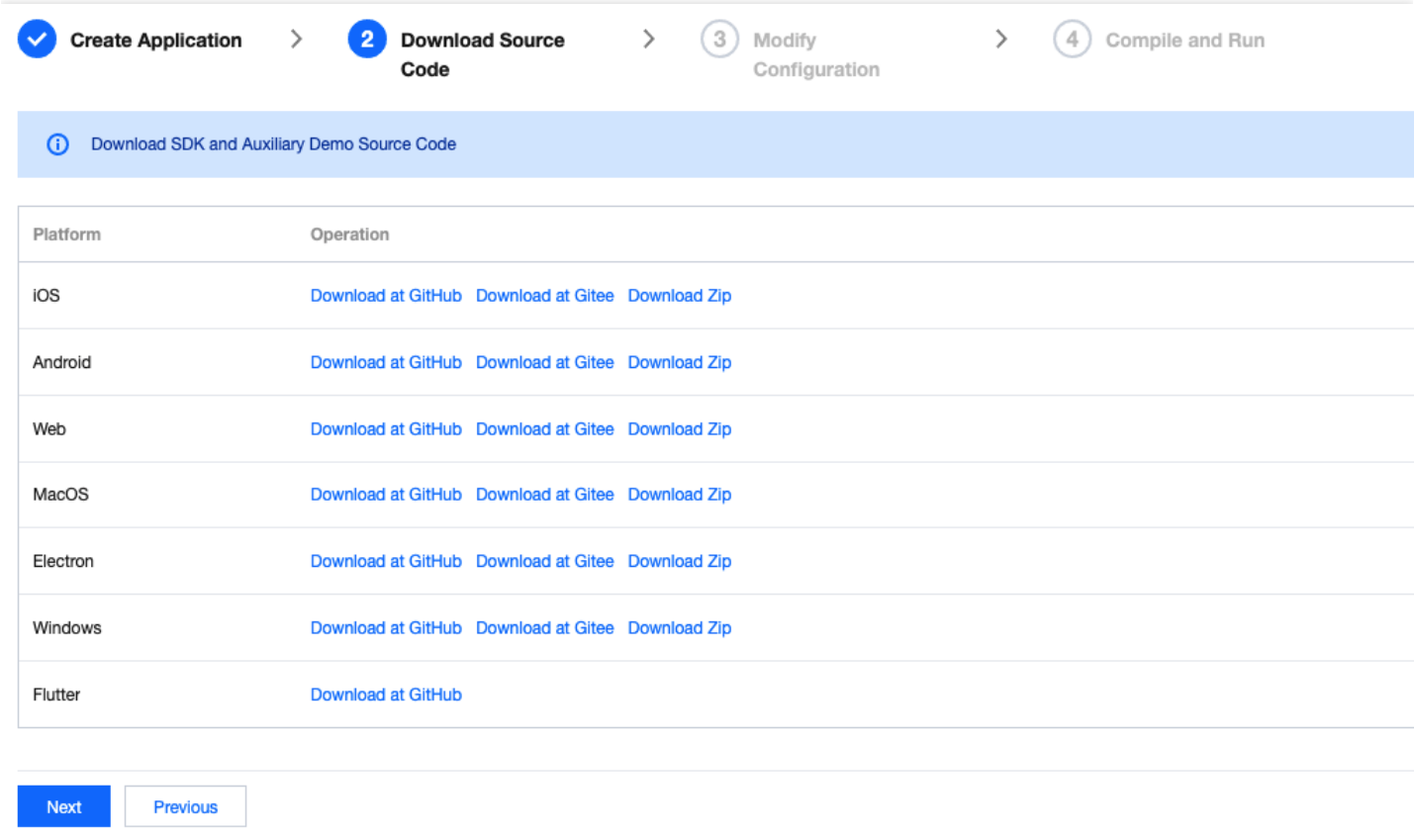
最終更新日：：2022-03-01 12:18:47

当社のAppをダウンロードしてインストールし、多人数オーディオビデオルームの効果をご体験いただけます。これには、画面共有、美顔、低遅延ミーティングなど、TRTCの多人数オーディオビデオルームシーン関連の機能が含まれています。

AppのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、**開発支援** > **Demoクイックスタート**を選択します。
2. アプリケーション名（例： `TestMeetingRoom` ）を入力し、**作成**をクリックします。
3. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。



注意：

©2013-2019 Tencent Cloud. All rights reserved.

本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については 、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

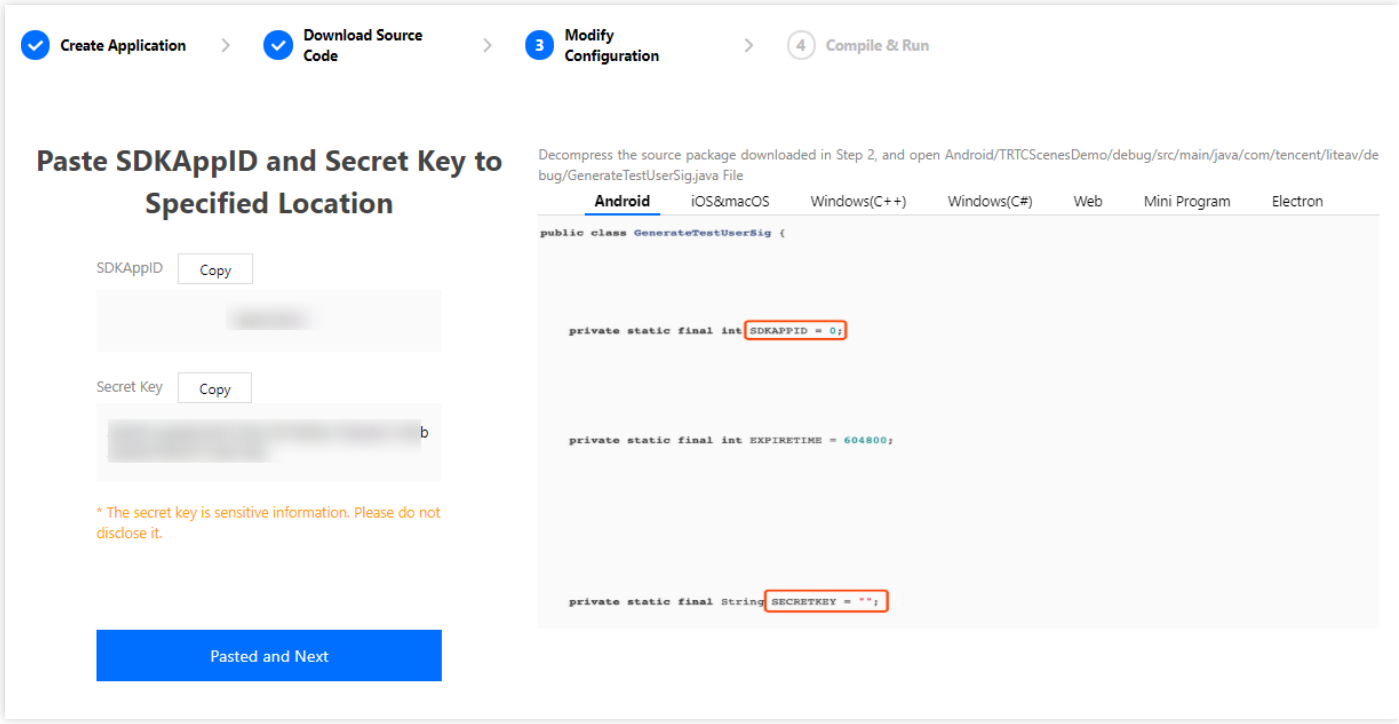
TRTCFlutterScenesDemoクリックして当該ページに入り、ソースコードをCloneまたはダウンロードします。

手順3：Appファイルの設定

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `/lib/debug/GenerateTestUserSig.dart` ファイルを見つけて開きます。

3. `GenerateTestUserSig.dart` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはPLACEHOLDER。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトはPLACEHOLDER。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

©2013-2019 Tencent Cloud. All rights reserved.

注意：

- ここで言及したUserSigの新規作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：コンパイル実行

注意：
Android端末は実際のマシンで実行する必要があり、エミュレーターのデバッグはサポートされていません。

1. `flutter pub get` を実行します。
2. コンパイルを実行し、デバッグを行います。
   - iOS 端末
   - Android 端末
   i. XCode（11.0およびそれ以降のバージョン）を使用して、ソースコードディレクトリの `/iosプロジェクト` を開きます。
   ii. コンパイルを行い、Demoプロジェクトを実行します。

手順5：Demoソースコードの修正

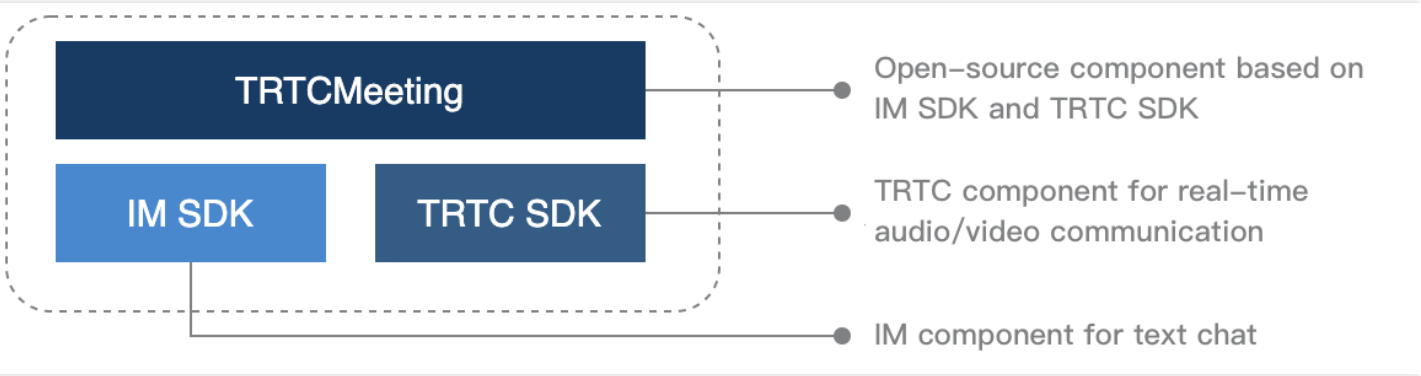
ソースコードのTRTCMeetingDemoフォルダには、uiとmodelの2つのサブフォルダが含まれ、uiフォルダの中はいずれもインターフェースコードです。以下の表にはファイルまたはフォルダおよび対応するUIをリストアップし、二次調整に役立つようにしています。

ファイルまたはフォルダ	機能説明
TRTCMeetingIndex.dart	ミーティング作成または参加インターフェース
TRTCMeetingRoom.dart	ビデオミーティングのメインインターフェース
TRTCMeetingMemberList.dart	参加者リストインターフェース
TRTCMeetingSetting.dart	ビデオミーティング関連パラメータ設定インターフェース

©2013-2019 Tencent Cloud. All rights reserved.

UIカスタマイズの実装

ソースコードの中のTRTCMeetingDemoフォルダにはuiとmodelの2つのサブフォルダが含まれ、modelフォルダには再利用可能なオープンソースコンポーネントTRTCMeetingが含まれています。 `TRTCMeeting.dart` ファイルの中からこのコンポーネントが提供するインターフェース関数を見つけ、対応するインターフェースを使用してカスマイズUIを実装することが可能です。



手順1：SDKの統合

インタラクティブライブストリーミングコンポーネントTRTCMeetingは、TRTC SDKとIM SDKに依存しています。 `pubspec.yaml` を設定することで、自動的にダウンロードして更新できます。

プロジェクトの `pubspec.yaml` に次の依存関係を記述します。

```
dependencies:
tencent_trtc_cloud: 最新バージョン番号
tencent_im_sdk_plugin: 最新バージョン番号
```

手順2：権限の設定および難読化ルール

- iOS 端末
- Android 端末

`Info.plist` にカメラとマイクの権限申請を追加する必要があります。

```
<key>NSMicrophoneUsageDescriptionkey>
<string>通常の音声通話が行えるようにマイクの権限を承認しますstring>
```

手順3：TRTCMeetingコンポーネントのインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

©2013-2019 Tencent Cloud. All rights reserved.

```
lib/TRTCMeetingDemo/model/
```

手順4：コンポーネントの作成およびログイン

1. `sharedInstance` インターフェースを呼び出すと、TRTCMeetingコンポーネントのインスタンスオブジェクトを作成できます。
2. `registerListener` 関数を呼び出して、コンポーネントのイベント通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	作用
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算、使用方法をご参照ください。

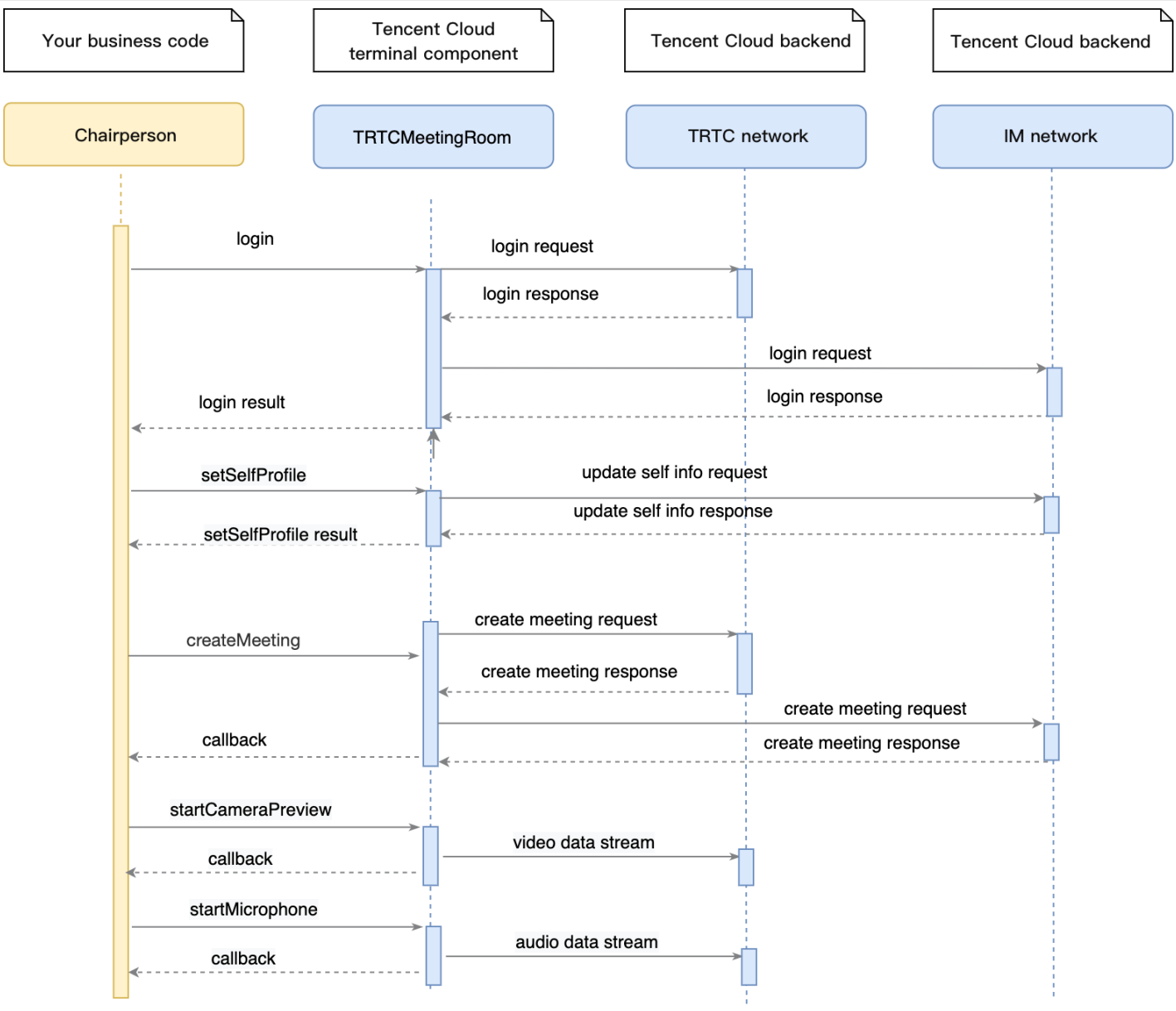
```
TRTCMeeting trtcMeeting = TRTCMeeting.sharedInstance();
trtcMeeting.registerListener(onListener);
ActionCallback res = await trtcMeeting.login(
GenerateTestUserSig.sdkAppId,
userId,
GenerateTestUserSig.genTestSig(userId),
);
if (res.code == 0) {
// ログイン成功
}
```

手順5：多人数ミーティングの新規作成

1. キャスターは、手順4でログインした後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. キャスターがcreateMeetingを呼び出し、新しいミーティングルームを作成します。
3. キャスターは、 `startCameraPreview` を呼び出してビデオ画面をキャプチャすることができ、 `startMicrophone` を呼び出して音声をキャプチャすることもできます。
4. キャスターに美顔のニーズがある場合は、インターフェース上に美顔調節ボタンを設定して呼び出すことができ、getBeautyManager を介して美顔設定を行います。

©2013-2019 Tencent Cloud. All rights reserved.

説明：
エンタープライズ版以外のSDKは変顔やスタンプの機能をサポートしていません。




```
// 1. キャスターがニックネームとプロフィール画像を設定
trtcMeeting.setSelfProfile('my_name', 'my_avatar');

// 2. キャスターがミーティングを作成
ActionCallback res = await trtcMeeting.createMeeting(roomId);
if (res.code == 0) {
// ミーティング作成に成功
// 3. カメラと音声キャプチャを起動
```

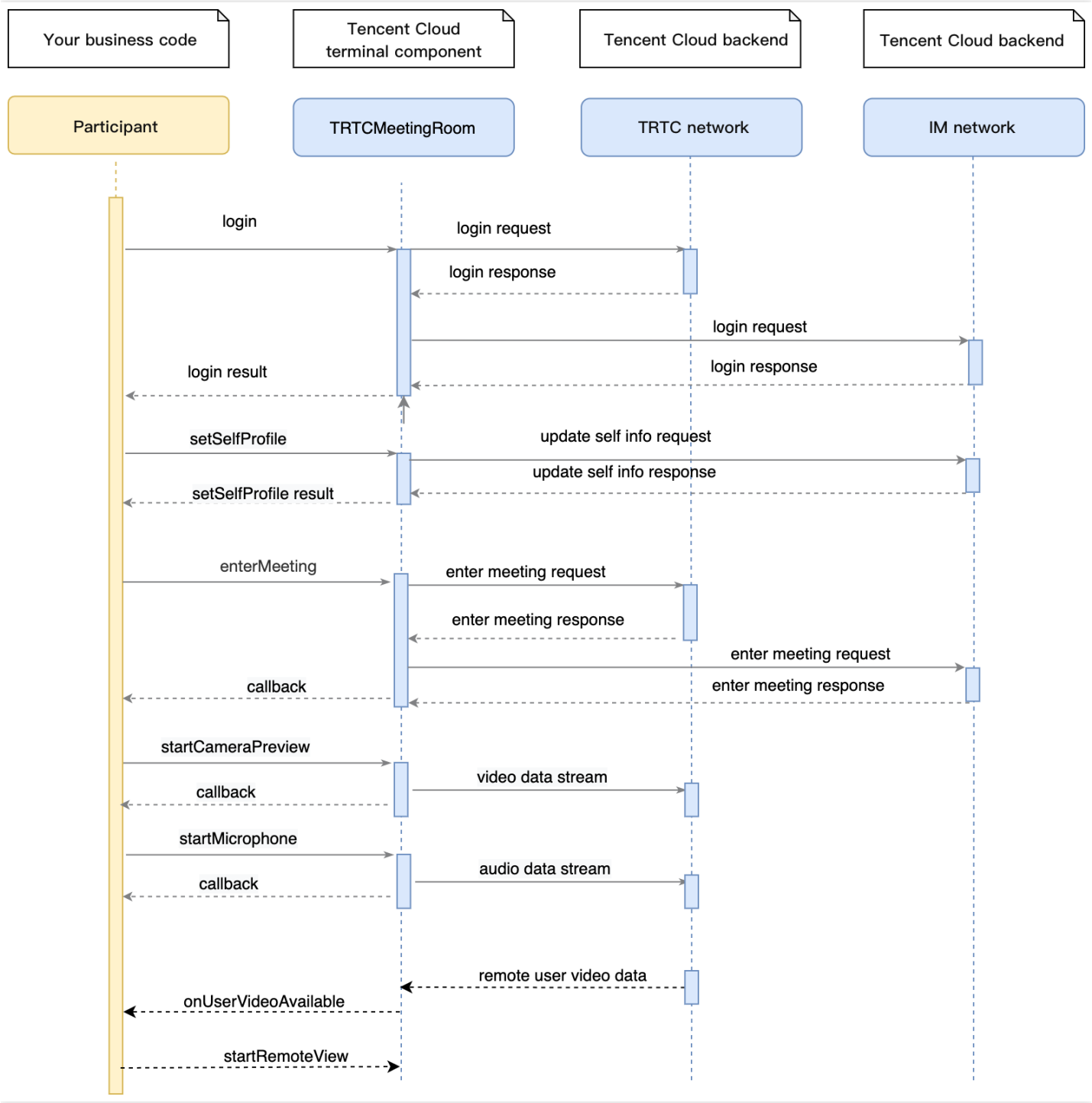
©2013-2019 Tencent Cloud. All rights reserved.

```
trtcMeeting.startCameraPreview(true, TRTCCloudVideoViewId);
trtcMeeting.startMicrophone();
// 4. 美顔設定
trtcMeeting.getBeautyManager().setBeautyStyle(TRTCCloudDef.TRTC_BEAUTY_STYLE_PITU);
trtcMeeting.getBeautyManager().setBeautyLevel(6);
}
```

手順6：参加者の多人数ミーティングへの参加

1. 参加者は、手順4でログインした後、setSelfProfileを呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 参加者は、enterMeeting呼び出してミーティングルーム番号を渡すと、ミーティングルームに参加できます。
3. 参加者は、startCameraPreviewを呼び出してビデオ画面をキャプチャすることができ、またstartMicrophoneを呼び出して音声をキャプチャすることもできます。
4. 他の参加者がカメラを起動すると、onUserVideoAvailableのイベントを受け取ります。この時、startRemoteViewを呼び出してuserIdを渡すと再生を開始できます。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1. 参加者がニックネームとプロフィール画像を設定
trtcMeeting.setSelfProfile('my_name', 'my_avatar');

// 2. 参加者がenterMeetingを呼び出してミーティングルームに参加
ActionCallback res = await trtcMeeting.enterMeeting(roomId);
if (res.code == 0) {
// ミーティング参加に成功
// 3. カメラと音声キャプチャを起動
```

©2013-2019 Tencent Cloud. All rights reserved.

```
trtcMeeting.startCameraPreview(true, TRTCCloudVideoViewId);
trtcMeeting.startMicrophone();
// 4. 美顔設定
trtcMeeting.getBeautyManager().setBeautyStyle(TRTCCloudDef.TRTC_BEAUTY_STYLE_PITU);
trtcMeeting.getBeautyManager().setBeautyLevel(6);
}

// 5. 参加者が他のメンバーのカメラ起動の通知を受信し、再生が開始されます
trtcMeeting.registerListener(onListener);
onListener(TRTCMeetingDelegate type, param) {
switch (type) {
case TRTCMeetingDelegate.onUserVideoAvailable:
if (param['available']) {
trtcMeeting.startCameraPreview(
param['userId'],
TRTCCloudDef.TRTC_VIDEO_STREAM_TYPE_BIG,
TRTCCloudVideoViewId
);
}else{
trtcMeeting.stopRemoteView(
param['userId'],
TRTCCloudDef.TRTC_VIDEO_STREAM_TYPE_BIG
);
}
break;
}
}
```

手順7：画面共有

1. 画面共有機能は、システムにフローティングウィンドウの権限を申請する必要があり、自分でUIの中で特定のロジックを実装する必要があります。
2. startScreenCaptureを呼び出し、エンコードパラメータとスクリーンキャプチャのプロセスのフローティングウィンドウを渡すと、画面共有機能を実装できます。具体的な情報については、TRTC SDKをご参照ください。
3. ミーティング中の他メンバーがonUserVideoAvailableのイベント通知を受け取ります。

> 注意：
> 画面共有とカメラキャプチャは相互排他的な操作です。画面共有機能をアクティブにしたい場合は、先にstopCameraPreviewを呼び出してカメラキャプチャを停止してください。詳細については、TRTC SDKをご参照ください。

©2013-2019 Tencent Cloud. All rights reserved.

```
await trtcMeeting.stopCameraPreview();
trtcMeeting.startScreenCapture(
videoFps: 10,
videoBitrate: 1600,
videoResolution: TRTCCloudDef.TRTC_VIDEO_RESOLUTION_1280_720,
videoResolutionMode: TRTCCloudDef.TRTC_VIDEO_RESOLUTION_MODE_PORTRAIT,
appGroup: iosAppGroup,
);
```

手順8：文字チャットとミュートメッセージの実現

- sendRoomTextMsgにより、ノーマルなテキストメッセージを送信でき、そのミーティングのすべての参加者がonRecvRoomTextMsgのコールバックを受信します。IMバックエンドにはデフォルトのセンシティブワードフィルタリングルールが備わっており、センシティブワードと判定されたテキストメッセージがクラウドで転送されることはありません。

```
// 発信側：テキストメッセージの発信
trtcMeeting.sendRoomTextMsg('Hello Word!');
// 受信側：テキストメッセージのモニタリング
trtcMeeting.registerListener(onListener);
onListener(TRTCMeetingDelegate type, param) {
switch (type) {
case TRTCMeetingDelegate.onRecvRoomTextMsg:
print( param['userName'] + 'からのメッセージ：' + param['message']+'を受信' );
break;
}
}
```

- sendRoomCustomMsgにより、カスタム（コマンド）メッセージを送信でき、そのミーティングのすべての参加者がonRecvRoomCustomMsgのコールバックを受信します。カスタムメッセージは、通常カスタマイズしたシグナリングの転送に使用されます（例：ミュートの会場内制御など）。

```
// 送信側：カスタマイズしたcmdでミュートの通知を区別できます
// eg: "CMD_MUTE_AUDIO"はミュートの通知を表します
trtcMeeting.sendRoomCustomMsg('CMD_MUTE_AUDIO', '1');
// 受信側：カスタムメッセージのモニタリング
trtcMeeting.registerListener(onListener);
onListener(TRTCMeetingDelegate type, param) {
switch (type) {
case TRTCMeetingDelegate.onRecvRoomCustomMsg:
if (param['command'] == 'CMD_MUTE_AUDIO') {
// ミュートの通知を受信します
```

©2013-2019 Tencent Cloud. All rights reserved.

```
print(param['userName'] +'から' + 'のミュートの通知：' + param['message']'を受信');
trtcMeeting.muteLocalAudio(message == '1');
}
break;
}
}
```

©2013-2019 Tencent Cloud. All rights reserved.

多人数オーディオビデオルーム (Windows&Mac)

最終更新日：：2022-02-18 15:32:27

ここで紹介するPC版TUIRoomコンポーネントは、柔軟なレイアウトで、適用性の高いオーディオビデオコミュニケーションツールです。コラボレーションオフィス、リモート採用、リモート問診、保険金査定、オンラインカスタマーサービス、ビデオ面接、デジタル行政、金融のデジタル化、オンラインミーティング、eラーニングなどのシーンで使用することができます。 各業界のシナリオと深く結びつき、企業のコスト削減と効率アップを助け、デジタルトランスフォーメーションを加速し、競争力を向上させます。
Appをダウンロードしてインストールすると、体験することができます。

ソリューションの優位性

- 超低遅延のオーディオビデオ通話、チャットルーム、画面共有、美顔、デバイス検出、データ統計などの機能を統合し、多人数オーディオビデオルームの一般的な機能をカバーします。
- 二次開発の需要に応じて、カスタムUIやレイアウトを迅速に実装し、ビジネスの迅速な立ち上げに役立ちます。
- TRTCとIMの基本SDKをカプセル化し、基本的なロジックコントロールを実装し、呼び出しを容易にするインターフェースを提供します。

アクセスガイド

多人数オーディオビデオルームの機能に素早くアクセスしたい場合は、当社が提供するAppを直接修正して適応させるか、App内のModuleを使用してカスタマイズしたUIを実装することができます。

手順1：Appソースコードのダウンロード

RoomAppをクリックして移動し、ソースコードをCloneまたはダウンロードします。

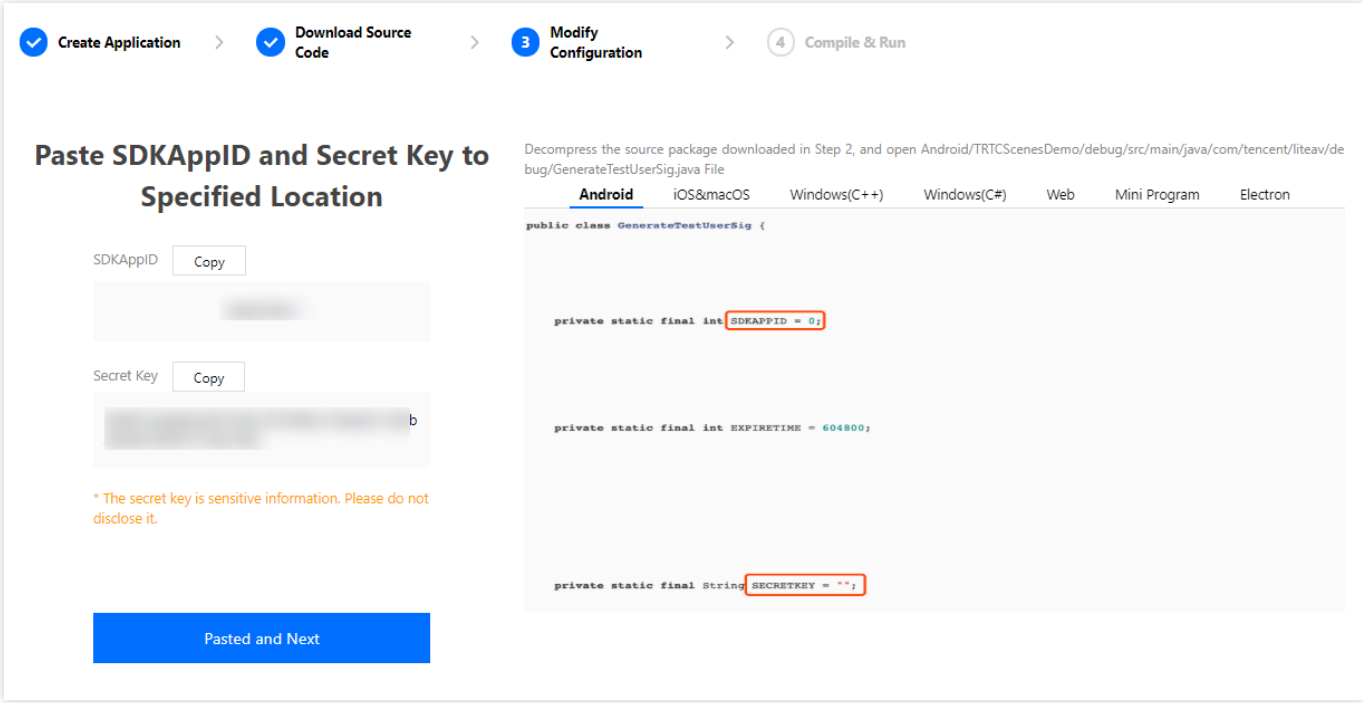
手順2：Appファイルの設定

- Windows端末の設定
- Mac端末の設定

1. `Windows¥RoomApp¥utils¥usersig¥win¥GenerateTestUserSig.h` ファイルを見つけて開きます。
2. `GenerateTestUserSig.h` 内のパラメータを設定します。
   - **SDKAPPID**：デフォルトは0。実際のSDKAPPIDを設定してください。

©2013-2019 Tencent Cloud. All rights reserved.

- **SECRETKEY**：デフォルトはブランク、実際のSECRETKEYを設定してください。



注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

ステップ3：Appの設定および実行

- Windows端末の設定
- Mac端末の設定

1. コンパイル環境にはVirtual Studio（2015バージョン以上）を使用します。
2. AppのUIには、現在主流のQTフレームワーク5.9バージョンを使用します。QTをダウンロードしてインストールし、VSでQTツールを設定してください。またはバージョンに応じて、AppプロジェクトでQT設定を変更して設定を完了してください。

©2013-2019 Tencent Cloud. All rights reserved.

3. RoomAppのRoomApp.vcxproj ソースコードプロジェクトを開き、プログラムをコンパイルし、実行します。

手順4：Appソースコードの修正

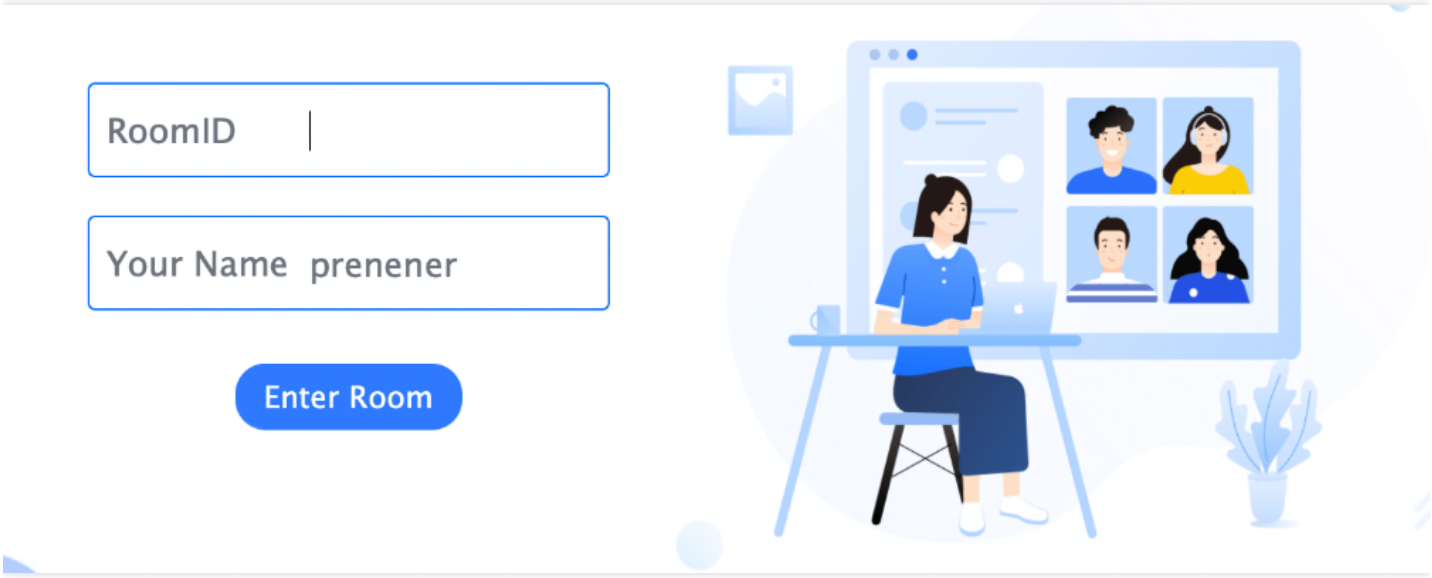
- Module層は、TRTCとIMをカプセル化して、基本的なロジックコントロールを実装し、UI層の呼び出しを容易にする機能インターフェースを提供します。
- AppモジュールはUIモジュールであり、インターフェースの機能実装が含まれています。必要に応じて二次開発を行ったり、UI全体を置き換えたりすることができます。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ログインインターフェース

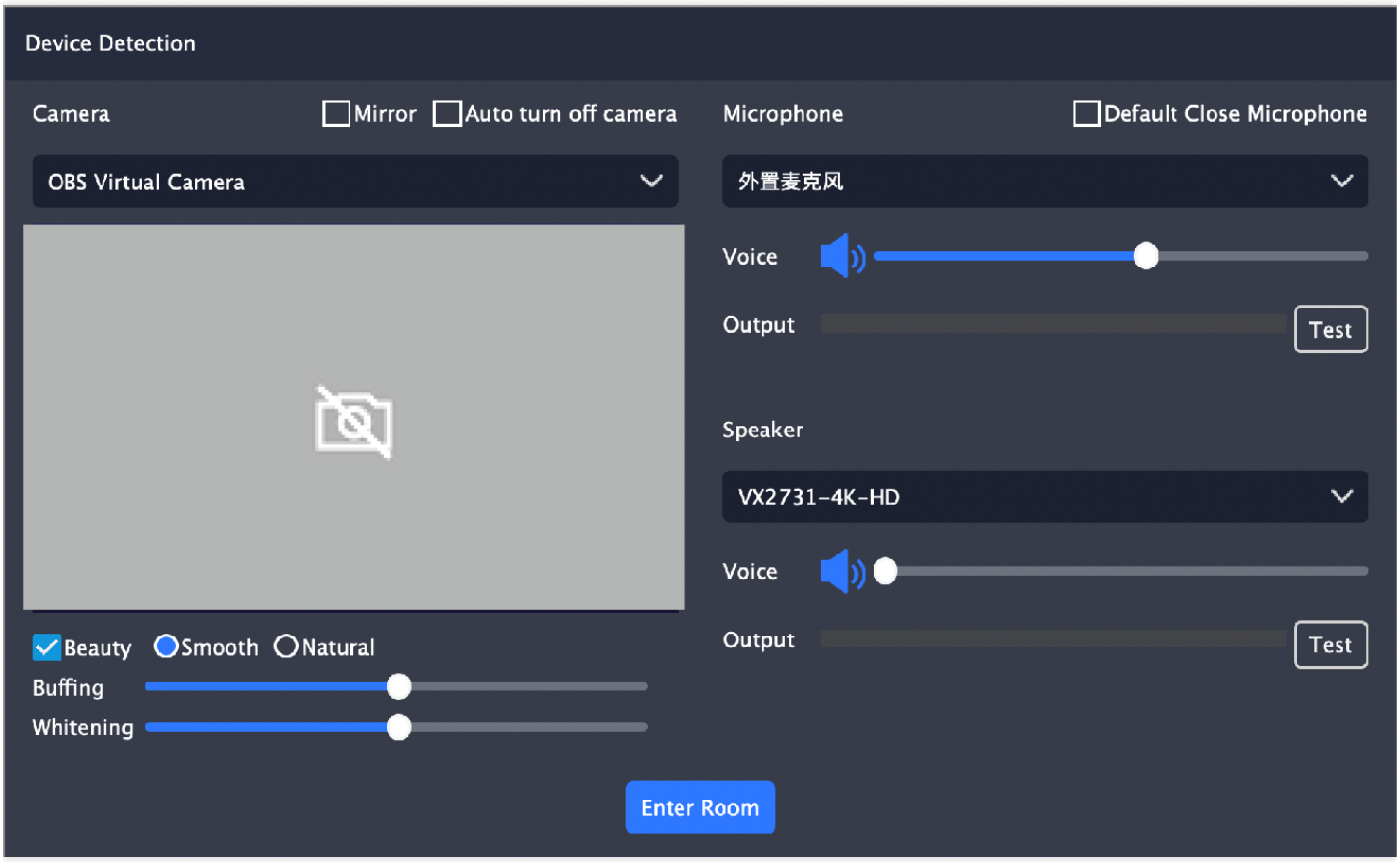
図のように、ルーム番号とユーザー名（ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください）を入力してログインします。



デバイス検出ページ

ここでは、デバイスの検出や、入室に使用するデバイスの設定、美顔効果の設定などを必要に応じて行うことができます。

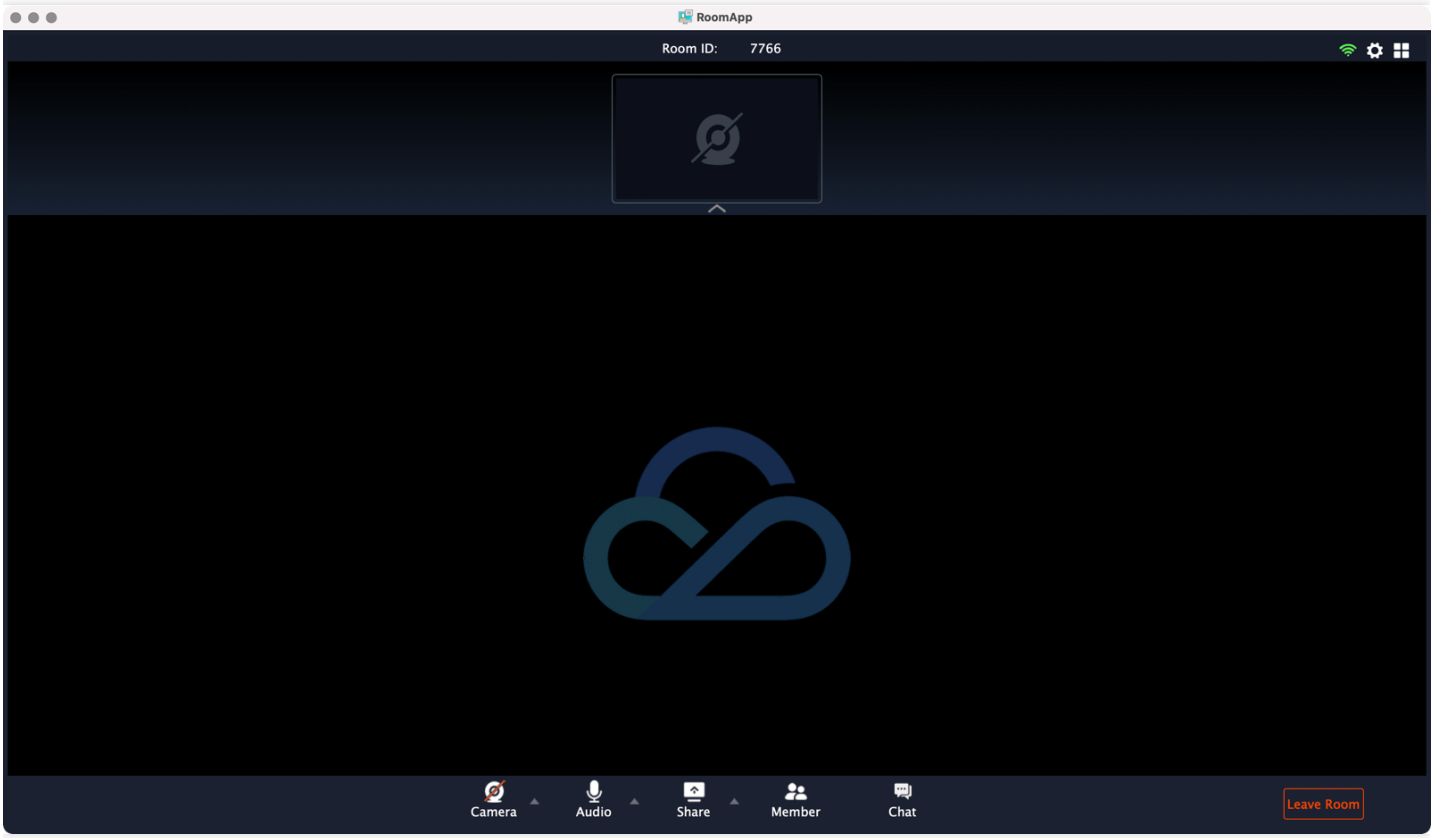
©2013-2019 Tencent Cloud. All rights reserved.



メインインターフェース

最初に入室するのはキャスターです。メインインターフェースには配信側リスト、下部のツールバー、上部のツールバーなどが含まれます。

©2013-2019 Tencent Cloud. All rights reserved.



配信側リスト

配信側リストには現在のルームのメンバーを表示できます。相手がカメラとマイクを起動すると、相手の画面を見ること、相手の声を聞くことができます。

下部のツールバー

自身のマイク/カメラの制御、画面共有の開始、メンバーリスト、チャットルームなどの機能を実装しました。

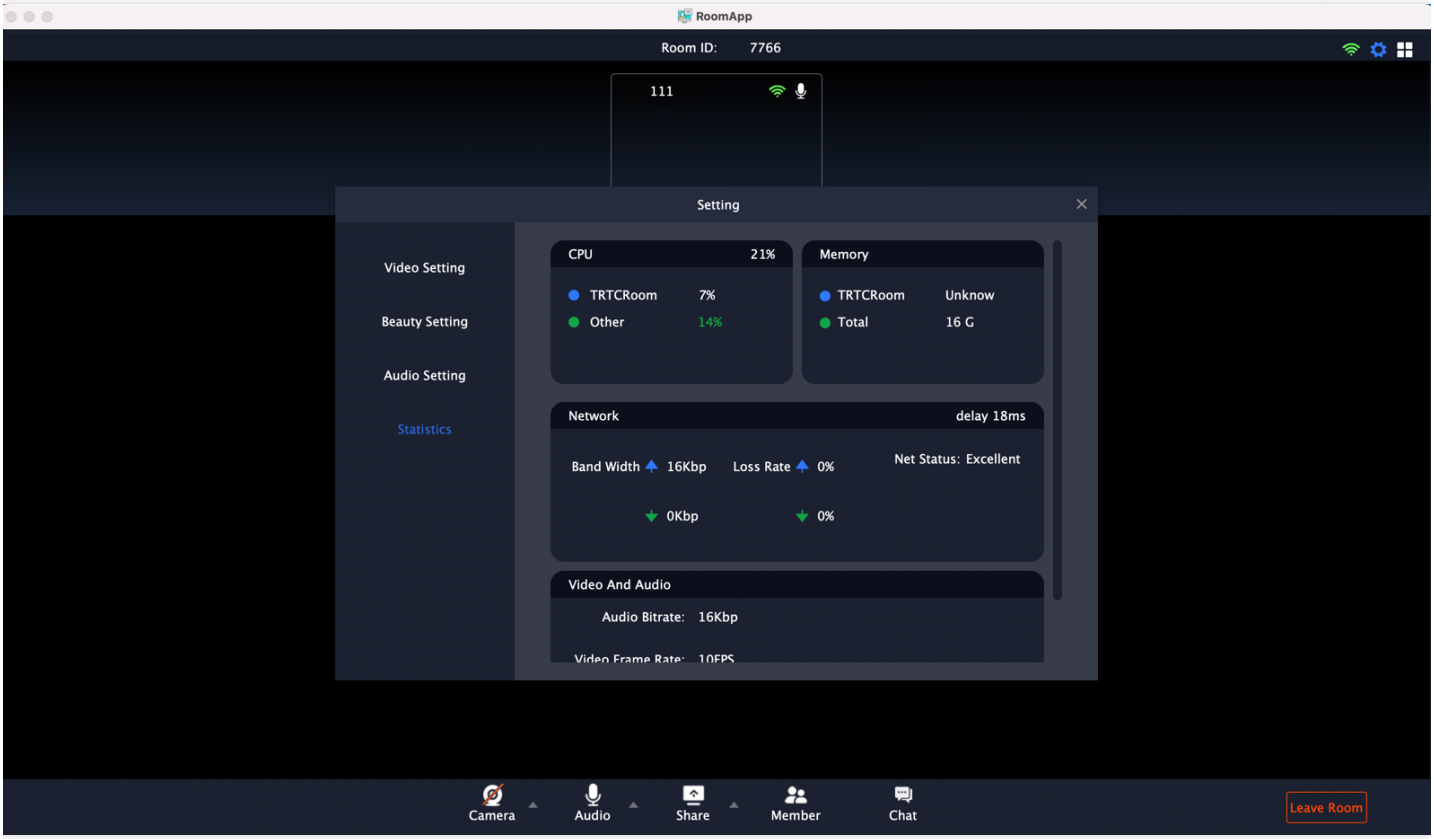
上部のコントロールバー

ネットワーク情報、設定ページ、配信側リストレイアウト機能を実装しました。

設定ウィンドウ

©2013-2019 Tencent Cloud. All rights reserved.

設定ウィンドウでは、デバイスを選択し、美顔などのパラメータを設定することができます。

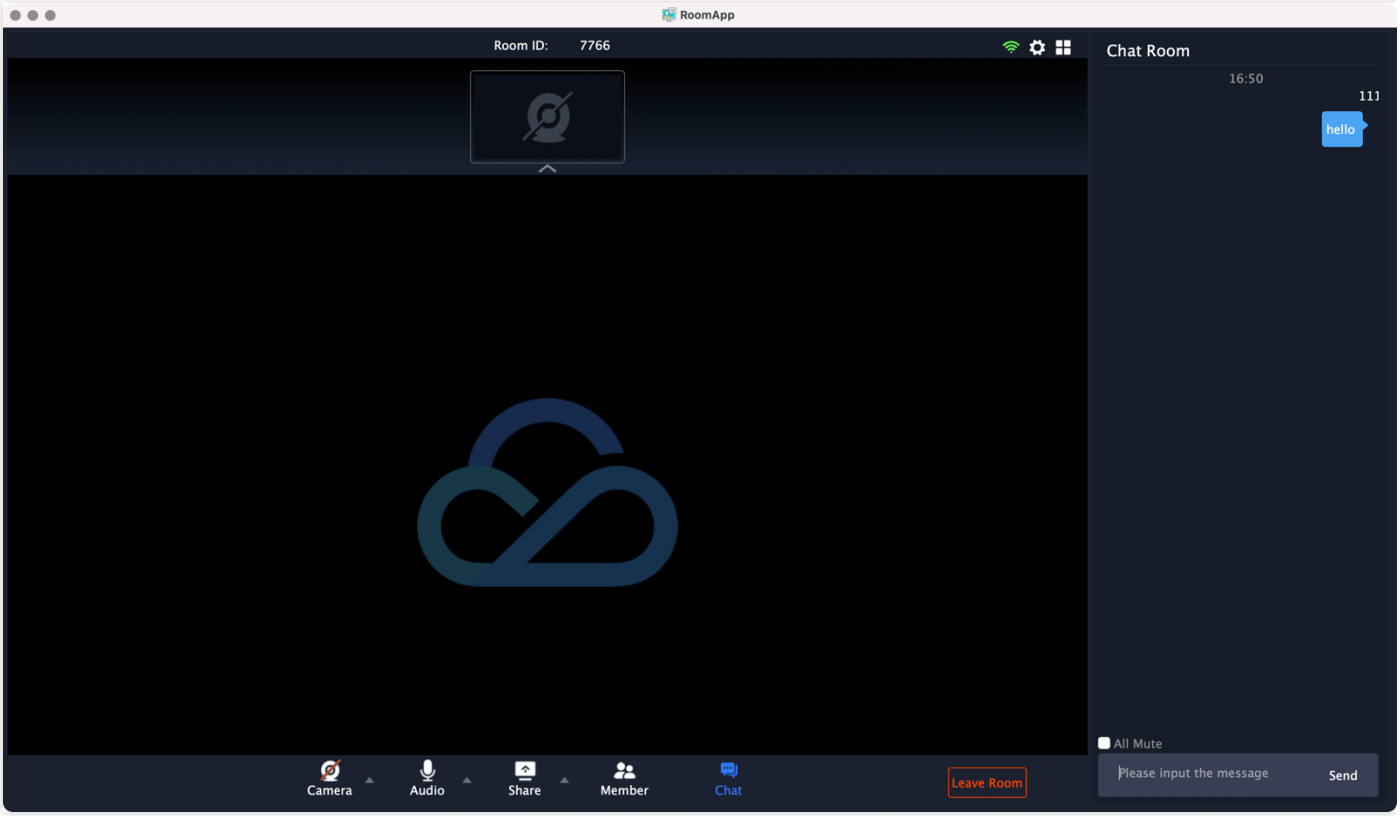


チャットルーム

- チャットルームはグループメンバーのチャットを行うことができます。

©2013-2019 Tencent Cloud. All rights reserved.

- キャスターは全員に対してミュート/ミュート解除を行うことができます。

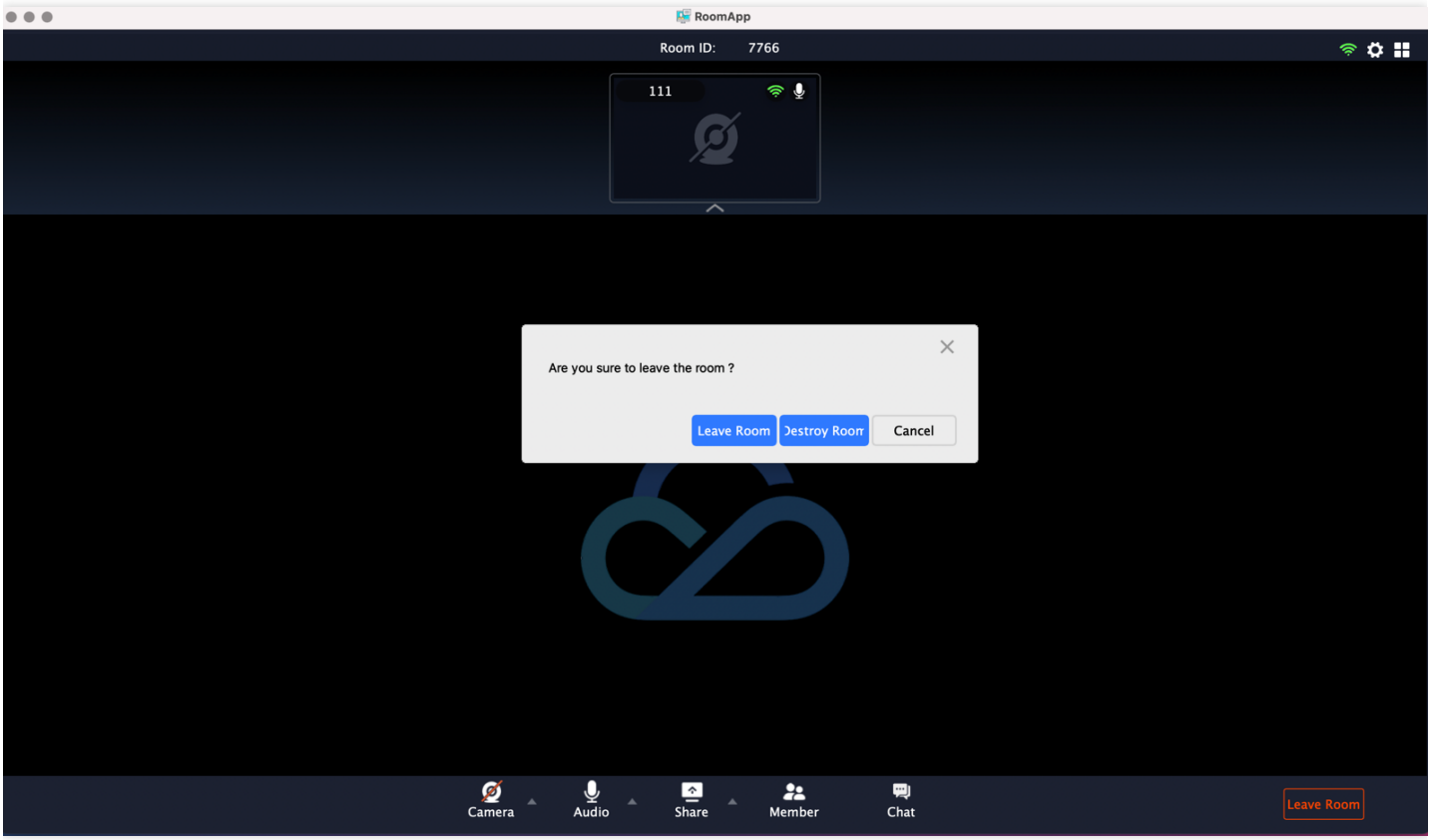


退室

キャスターが退室する場合、2つの選択肢があります。

- **ルームの解散**：全員が退室することを指します。全員が退室する必要があります。
- **自身が退室**：キャスターの権限を渡して、他のユーザーがキャスターとなります。

©2013-2019 Tencent Cloud. All rights reserved.



UIカスタマイズの実装

ソースコードのModuleには、TRTC SDKおよびIM SDK のカプセル化が含まれています。 `TUIRoomCore.h` 、 `TUIRoomCoreCallback.h` 、 `TUIRoomDef.h` などのファイルでこのモジュールが提供するインターフェース関数およびその他定義を確認し、対応するインターフェースを使用してカスタマイズされたUIを実装できます。

ステップ1：SDKの統合

公式サイトから TRTC SDKとIM SDKをダウンロードし、外部のSDKディレクトリのIM SDKとLiteAVSDKディレクトリのファイルを置き換えます。

> 注意：
> IM SDKはC++バージョンを使用します。

SDK	ダウンロードページ	統合ガイド

©2013-2019 Tencent Cloud. All rights reserved.

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD統合ドキュメント	

ステップ2：プロジェクト設定SDKディレクトリおよびライブラリファイル

- **Windows端末**：VSでヘッダーファイルディレクトリ、静的ライブラリディレクトリ、および依存する静的ライブラリファイルを設定します。
- **Mac端末**：proおよびpriファイルで依存ライブラリとファイルディレクトリを追加します。

ステップ3：作成およびログイン

1. `IUIRoomCore::Instance` インターフェースを使用してIUIRoomCoreシングルトンを作成し使用します。
2. AddCallbackインターフェースを呼び出し、コールバッククラスを設定し、SDKのコールバック通知を受信します。
3. Loginインターフェースを呼び出し、ログインを行います。

パラメータ名	説明
sdk_appid	TRTCコンソール で SDKAppIDを表示できます。
user_id	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
user_sig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算、使用方法をご参照ください。

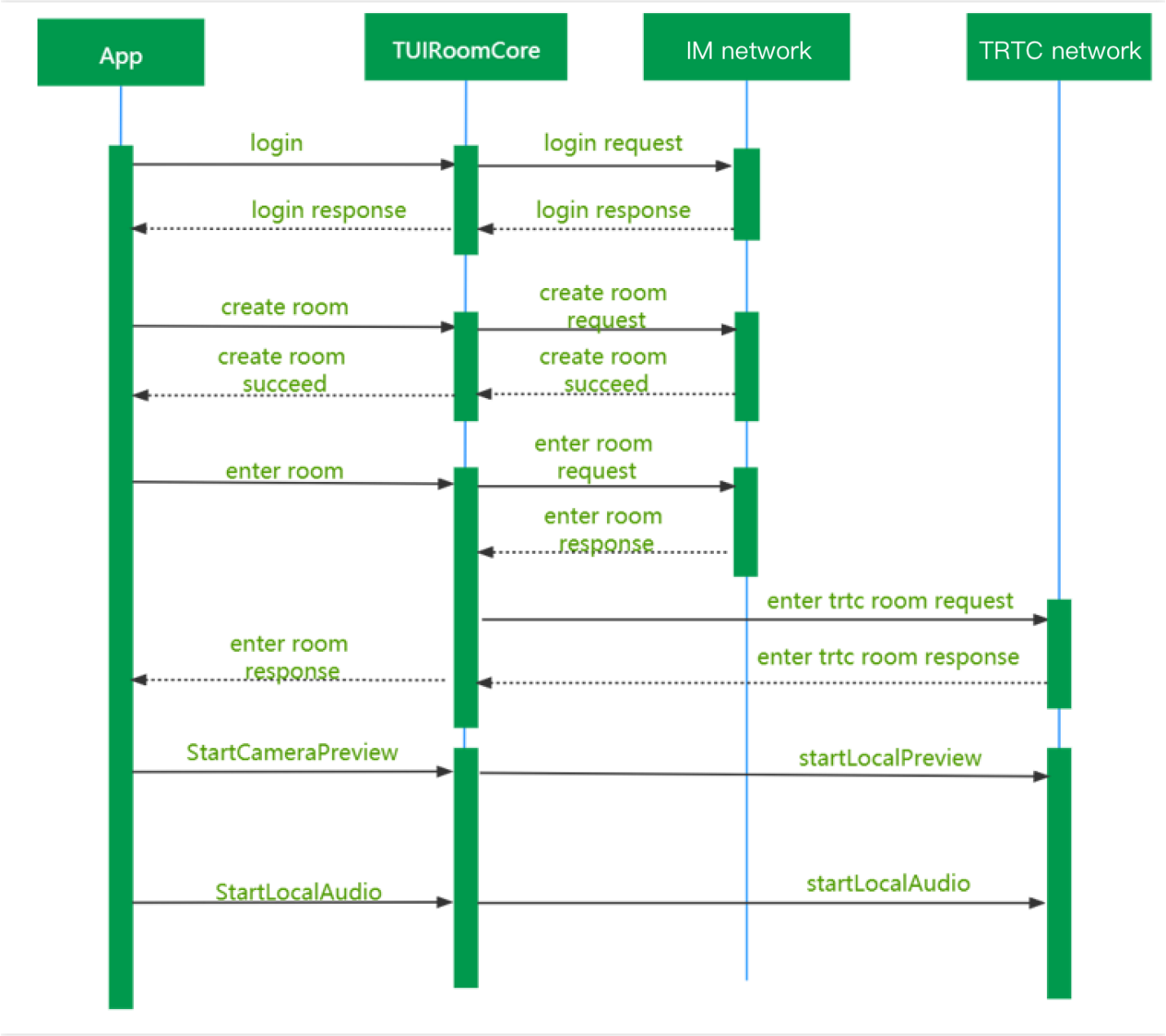
ステップ4：ニックネームの設定

正常にログインした後、SetSelfProfileを呼び出し、ユーザー情報を設定します。

ステップ5：ルームの作成

1. メンバーがCreateRoomインターフェースを呼び出し、新たなルームを作成します。ルームの作成後、キャスターとしてチャットルームとTRTC ルームを含むルームに入室します。
2. メンバーがStartCameraPreviewインターフェースを呼び出し 、ローカルビデオのキャプチャと表示を開始します。

©2013-2019 Tencent Cloud. All rights reserved.



ステップ6：ルームの解散

1. キャスターがDestoryRoomインターフェースを呼び出し 、ルームを解散します。IMグループチャットを解散して、TRTCルームを退室します
2. メンバー側は、グループを解散し、TRTCルームを退室するよう通知するOnExitRoomコールバックメッセージを受信します

ステップ7：ルームの譲渡

1. キャスターが、退室操作のためにTransferRoomMasterインターフェースを呼び出し、ルームを他の人に譲り渡します。

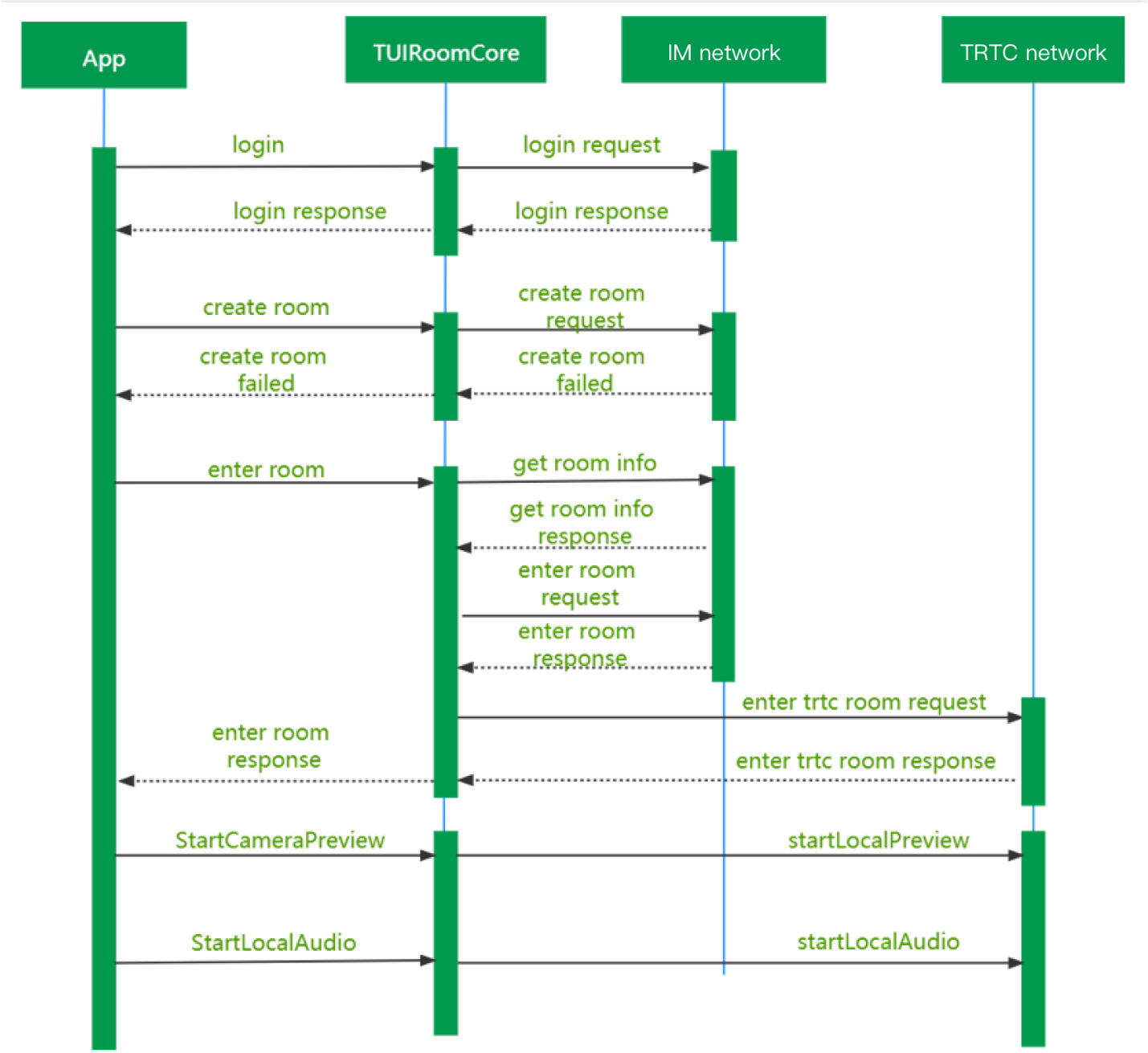
©2013-2019 Tencent Cloud. All rights reserved.

2. メンバー側は、ルームキャスターの変更を通知するOnRoomMasterChangedコールバックメッセージを受信します。
3. 変更後のキャスターは、キャスター権限を取得し、グループメンバーをコントロールできます

ステップ8：ルームへの参加

1. ルーム参加手順は、作成手順と基本的に同じです。メンバーは、入室時にまずルームを作成する必要があり、作成に失敗した場合、メンバーとして入室します。
2. ルームの作成に失敗した場合、まずルーム情報を取得し、その後、再度入室します。
3. その他メンバーは、メンバーの入室があったことを通知するOnRemoteUserEnter コールバックを受信します。

©2013-2019 Tencent Cloud. All rights reserved.



ステップ9：退室

1. メンバーがLeaveRoomインターフェースを呼び出し、IM ルームとTRTC ルームを退室します。
2. その他メンバーグループ側は、メンバーが退室したことを通知するOnRemoteUserLeaveコールバックを受信します。

手順10：画面共有

1. StartScreenCaptureインターフェース呼び出し 、画面共有を行います。

©2013-2019 Tencent Cloud. All rights reserved.

2. その他メンバーは、メンバーが画面共有中であることを通知するOnRemoteUserScreenAvailableコールバックを受信します。

> 説明：
> 画面共有モジュールは、ウィンドウ選択のロジックを実行する必要があります。具体的な実装については、Appの `ScreenShareWindow.h` の実装をご参照ください。

手順11：文字チャットとミュートメッセージの実現

1. SendChatMessageインターフェースを呼び出し、テキストを送信すると、ルーム内のメンバーはOnRecevieChatMessageコールバックメッセージを受信し、チャットメッセージを取得し表示できます。
2. キャスターはMuteChatRoomインターフェースを呼び出し、チャットをミュートし、または解除できます。メンバー側はOnChatRoomMutedコールバックを受信し、UI層が対応する処理を行います。

©2013-2019 Tencent Cloud. All rights reserved.

TUIRoom(iOS)

最終更新日：：2022-03-01 12:28:18

TUIRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- キャスターがルームを作成し、入室者はルームナンバーを入力した後に入室できます。
- 入室者の間で画面共有を行います。
- 各種のテキストメッセージとカスタムメッセージの送信をサポートします。

TUIRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実装プロセスについては、 多人数オーディオビデオルーム(iOS)をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオルームコンポーネントとして使用します。
- IM SDK：IM SDK を利用してチャットルームの機能（**IM SDKはiOSバージョンを使用**）を実装します。

TUIRoom API概要

TUIRoomCore基本関数

API	説明
shareInstance	シングルトンオブジェクトを取得します。
destroyInstance	シングルトンオブジェクトを破棄します。
setDelegate	イベントコールバックを設定します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（キャスターが呼び出し）。
destroyRoom	ルームの破棄（キャスターが呼び出し）。
enterRoom	入室（入室メンバーが呼び出し）。
leaveRoom	退室（その他のルームメンバーが呼び出し）。
getRoomInfo	ルーム情報の取得。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
getRoomUsers	ルーム内全メンバー情報の取得。
getUserInfo	特定ユーザーの情報の取得。
transferRoomMaster	キャスター権限の移転（キャスターが呼び出し）。

ローカルのオーディオビデオ操作インターフェース

API	説明
startCameraPreview	ローカルビデオのプレビュー画面を立ち上げます。
stopCameraPreview	ローカルのビデオキャプチャおよびプレビューを停止します。
startLocalAudio	マイクキャプチャを起動します。
stopLocalAudio	マイクキャプチャを停止します。
setVideoMirror	ローカル画面のミラーモードのプレビューを設定します。
setSpeaker	スピーカーの起動を設定します。

リモートユーザーに関するインターフェース

API	説明
startRemoteView	指定メンバーのリモートビデオ画面をサブスクリプションし再生します。
stopRemoteView	リモートビデオ画面のサブスクリプションをキャンセルし再生を停止します。

チャットメッセージ送信インターフェース

API	説明
sendChatMessage	チャットメッセージを送信します。
sendCustomMessage	カスタムメッセージを送信します。

フィールドコントロール関連インターフェース

API	説明
muteUserMicrophone	特定ユーザーのマイクを無効化/再有効化します。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
muteAllUsersMicrophone	全ユーザーのマイクを無効化/再有効化し、ステータスをルーム情報に同期させます。
muteUserCamera	特定ユーザーのカメラを無効化/再有効化します。
muteAllUsersCamera	全ユーザーのカメラを無効化/再有効化し、ステータスをルーム情報に同期させます。
muteChatRoom	チャットルームのミュートを開始/停止します（キャスターが呼び出し）。
kickOffUser	ルーム内の特定ユーザーをリムーブします（キャスターが呼び出し）。
startCallingRoll	キャスターが点呼を開始します。
stopCallingRoll	キャスターが点呼を終了します。
replyCallingRoll	メンバーがキャスターの点呼に応答します。
sendSpeechInvitation	キャスターがメンバーに発言するようインビテーションを送信します。
cancelSpeechInvitation	キャスターがメンバーの発言のためのインビテーションをキャンセルします。
replySpeechInvitation	メンバーがキャスターの発言申請に同意/拒否します。
sendSpeechApplication	メンバーが発言を要請します。
replySpeechApplication	キャスターがメンバーの発言申請に同意/拒否します。
forbidSpeechApplication	キャスターが発言申請を禁止します。
sendOffSpeaker	キャスターがメンバーに発言を停止するよう命令します。
sendOffAllSpeakers	キャスターが全員に発言を停止するよう命令します。
exitSpeechState	メンバーは発言を停止し、視聴者になります。

画面共有インターフェース

API	説明
startScreenCapture	画面共有を開始。
stopScreenCapture	画面キャプチャの停止。

©2013-2019 Tencent Cloud. All rights reserved.

美顔フィルタに関するインターフェース関数

API	説明
getBeautyManager	美顔管理オブジェクトTXBeautyManagerを取得します。

関連設定インターフェース

API	説明
setVideoQosPreference	ネットワークトラフィックコントロール関連パラメータを設定します。

SDKバージョンインターフェース関数の取得

API	説明
getSDKVersion	SDKバージョンを取得します。

TUIRoomCoreDelegate API概要

エラーイベントコールバック

API	説明
onError	エラーのコールバック。

基本イベントコールバック

API	説明
onDestroyRoom	ルーム解散のコールバック。
onUserVoiceVolume	音量の大きさのコールバック。
onRoomMasterChanged	キャスター変更のコールバック。

リモートユーザーイベントコールバック

API	説明
onRemoteUserEnter	リモートユーザー入室コールバック。
onRemoteUserLeave	リモートユーザー退室コールバック。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onRemoteUserCameraAvailable	リモートユーザーがカメラビデオを起動するかどうかのコールバック。
onRemoteUserScreenVideoAvailable	リモートユーザーが画面共有を開始するかどうかのコールバック。
onRemoteUserAudioAvailable	リモートユーザーがオーディオのアップストリームを開始するかどうかのコールバック。
onRemoteUserEnterSpeechState	リモートユーザーの発言開始のコールバック。
onRemoteUserExitSpeechState	リモートユーザーの発言終了のコールバック。

メッセージイベントのコールバック

API	説明
onReceiveChatMessage	テキストメッセージ受信のコールバック。

フィールドコントロールイベントコールバック

API	説明
onReceiveSpeechInvitation	ユーザーがキャスターの発言要請を受信した場合のコールバック。
onReceiveInvitationCancelled	ユーザーがキャスターの発言要請キャンセルを受信する場合のコールバック。
onReceiveSpeechApplication	キャスターがユーザーの発言要請を受信する場合のコールバック。
onSpeechApplicationCancelled	ユーザーが発言申請をキャンセルする場合のコールバック。
onSpeechApplicationForbidden	キャスターが発言申請を禁止する場合のコールバック。
onOrderedToExitSpeechState	メンバーが発言の停止をリクエストされる場合のコールバック。
onCallingRollStarted	キャスターが点呼を開始し、メンバーが受信する場合のコールバック。
onCallingRollStopped	キャスターが点呼を終了し、メンバーが受信する場合のコールバック。
onMemberReplyCallingRoll	メンバーが点呼に応答し、キャスターが受信する場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onChatRoomMuted	キャスターがチャットルームのミュートを変更する場合のコールバック。
onMicrophoneMuted	キャスターがマイクの無効化を設定する場合のコールバック。
onCameraMuted	キャスターがカメラの無効化を設定する場合のコールバック。
onReceiveKickedOff	メンバーがキャスターからキックアウトされた場合のコールバック。

統計および品質コールバック

API	説明
onStatistics	技術指標統計のコールバック。
onNetworkQuality	ネットワーク品質のコールバック。

画面共有関連コールバック

API	説明
onScreenCaptureStarted	画面共有開始のコールバック。
onScreenCaptureStopped	画面共有停止のコールバック。

TUIRoomCore基本関数

getInstance

TUIRoomCore シングルトンオブジェクトを取得します。

```
+ (instancetype)shareInstance;
```

destroyInstance

```
+ (void)destroyInstance;
```

setDelegate

©2013-2019 Tencent Cloud. All rights reserved.

TUIRoomCore イベントコールバック。TUIRoomCoreDelegateを介してTUIRoomCoreの各種ステータス通知を取得できます。

```
- (void)setDelegate:(id)delegate;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
delegate	TUIRoomCoreDelegate	イベントコールバッククラスを受信します。

createRoom

ルームの作成（キャスターが呼び出し）。

```
- (void)createRoom:(NSString *)roomId
speechMode:(TUIRoomSpeechMode)speechMode
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	NSString	ルームID。ご自身でアサインし、一元管理する必要があります。
speechMode	TUIRoomSpeechMode	発言モード。
callback	TUIRoomActionCallback	ルームの作成結果のコールバック。

キャスターの通常の呼び出しフローは以下のとおりです。

1. **キャスター**が `createRoom()` を呼び出し、ルームを作成します。ルーム作成の成否はTUIRoomActionCallbackでキャスターに通知されます。
2. **キャスター**が `startCameraPreview()` を呼び出し 、カメラキャプチャとプレビューを起動します。
3. **キャスター**が `startLocalAudio()` を呼び出し、ローカルマイクを起動します。

destroyRoom

ルームの破棄（キャスターが呼び出し）。キャスターは、ルームの作成後、この関数を呼び出して、ルームを破棄できます。

```
- (void)destroyRoom:(TUIRoomActionCallback)callback;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomActionCallback	ルームの破棄結果のコールバック。

enterRoom

入室（ルーム参加メンバーが呼び出し）。

```
- (void)enterRoom:(NSString *)roomId
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	NSString	ルームID。
callback	TUIRoomActionCallback	結果のコールバック。

ルーム参加メンバーが入室する場合の通常の呼び出し手順は次のとおりです。

1. **入室メンバー**が `enterRoom` を呼び出し、roomIdを渡せば、入室できます。
2. **入室メンバー**が `startCameraPreview()` を呼び出してカメラプレビューを起動し、 `startLocalAudio()` を呼び出して、マイクキャプチャを起動します。
3. **入室メンバー**が `onRemoteUserCameraAvailable` のイベントを受信し、 `startRemoteView()` を呼び出して、ビデオ再生を開始します。

leaveRoom

退室（入室メンバーが呼び出し）。

```
- (void)leaveRoom:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomActionCallback	結果のコールバック。

getRoomInfo

ルーム情報を取得します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (nullable TUIRoomInfo *)getRoomInfo;
```

getRoomUsers

ルームの全メンバー情報を取得します。

```
- (nullable NSArray *)getRoomUsers;
```

getUserInfo

ルームのメンバー情報を取得します。

```
- (void)getUserInfo:(NSString *)userId
callback:(TUIRoomUserInfoCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
callback	TUIRoomUserInfoCallback	ルームメンバーの詳細情報のコールバック。

setSelfProfile

ユーザー情報を設定します。

```
- (void)setSelfProfile:(NSString *)userName
avatarURL:(NSString *)avatarURL
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	NSString	ユーザーの氏名。
avatarURL	NSString	ユーザーのプロフィール画像URL。
callback	TUIRoomActionCallback	設定が成功したかどうかの結果のコールバック。

transferRoomMaster

グループを他のユーザーに引き渡します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)transferRoomMaster:(NSString *)userId
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
callback	TUIRoomActionCallback	結果のコールバック。

ローカルプッシュインターフェース

startCameraPreview

ローカルカメラプレビューを起動します。

```
- (void)startCameraPreview:(BOOL)isFront
view:(UIView *)view;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFront	BOOL	YES：フロントカメラ、NO：リアカメラ。
view	UIView	ビデオ画像をロードするウィジェット。

stopCameraPreview

ローカルカメラプレビューを停止します。

```
- (void)stopCameraPreview;
```

startLocalAudio

マイクの集音開始。

```
- (void)startLocalAudio:(TRTCAudioQuality)quality;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
quality	TRTCAudioQuality	キャプチャの音質。

stopLocalAudio

マイクの集音停止。

```
- (void)stopLocalAudio;
```

setVideoMirror

ローカル画面のミラーモードのプレビューを設定します。

```
- (void)setVideoMirror:(TRTCVideoMirrorType)type;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
type	TRTCVideoMirrorType	イメージタイプ。

setSpeaker

スピーカーの起動設定。

```
- (void)setSpeaker:(BOOL)isUseSpeaker;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isUseSpeaker	BOOL	YES：スピーカー、NO：ヘッドホン。

リモートユーザーに関するインターフェース

startRemoteView

リモートユーザーのビデオストリームのサブスクリプション。

```
- (void)startRemoteView:(NSString *)userId
view:(UIView *)view
```

©2013-2019 Tencent Cloud. All rights reserved.

```
streamType:(TUIRoomStreamType)streamType
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	再生が必要なユーザー ID。
view	UIView	ビデオ画像をロードするviewウィジェット。
streamType	TUIRoomStreamType	ストリームのタイプ。
callback	TUIRoomActionCallback	結果のコールバック。

stopRemoteView

サブスクリプションをキャンセルし、リモートビデオ画面の再生を停止します。

```
- (void)stopRemoteView:(NSString *)userId
streamType:(TUIRoomStreamType)streamType
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	再生停止が必要なユーザーのID。
streamType	TUIRoomStreamType	ストリームのタイプ。
callback	TUIRoomActionCallback	結果のコールバック。

switchCamera

フロント/リアカメラを切り替えます。

```
- (void)switchCamera:(BOOL)isFront;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFront	BOOL	YES：フロントカメラ、NO：リアカメラ。

©2013-2019 Tencent Cloud. All rights reserved.

メッセージ送信インターフェース

sendChatMessage

ルーム内でテキストメッセージをブロードキャストします。通常、テキストによるチャットに使用します。

```
- (void)sendChatMessage:(NSString *)message
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	NSString	メッセージの内容。
callback	TUIRoomActionCallback	送信結果のコールバック。

フィールドコントロール関連インターフェース

muteUserMicrophone

特定ユーザーのマイクを無効化/再有効化します。

```
- (void)muteUserMicrophone:(NSString *)userId
mute:(BOOL)mute
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
mute	BOOL	無効にするかどうか。
callback	TUIRoomActionCallback	結果のコールバック。

muteAllUsersMicrophone

全ユーザーのマイクを無効化/再有効化します。

```
- (void)muteAllUsersMicrophone:(BOOL)mute
callback:(TUIRoomActionCallback)callback;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	BOOL	無効にするかどうか。
callback	TUIRoomActionCallback	結果のコールバック。

muteUserCamera

特定ユーザーのカメラを無効化/再有効化します。

```
- (void)muteUserCamera:(NSString *)userId
mute:(BOOL)mute
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
mute	BOOL	無効にするかどうか。
callback	TUIRoomActionCallback	結果のコールバック。

muteAllUsersCamera

全ユーザーのカメラを無効化/再有効化します。

```
- (void)muteAllUsersCamera:(BOOL)mute
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	BOOL	無効にするかどうか。
callback	TUIRoomActionCallback	結果のコールバック。

muteChatRoom

テキストチャットのミュート/再有効化。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)muteChatRoom:(BOOL)mute
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	BOOL	無効にするかどうか。
callback	TUIRoomActionCallback	結果のコールバック。

kickOffUser

キャスターがキックアウトします。

```
- (void)kickOffUser:(NSString *)userId
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
callback	TUIRoomActionCallback	結果のコールバック。

startCallingRoll

キャスターが点呼を開始します。

```
- (void)startCallingRoll:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomActionCallback	結果のコールバック。

stopCallingRoll

キャスターが点呼を終了します。

```
- (void)stopCallingRoll:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	TUIRoomActionCallback	結果のコールバック。

replyCallingRoll

メンバーがキャスターの点呼に応答します。

```
- (void)replyCallingRoll:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomActionCallback	結果のコールバック。

sendSpeechInvitation

キャスターがメンバーの発言を要請します。

```
- (void)sendSpeechInvitation:(NSString *)userId
callback:(TUIRoomInviteeCallback)callback
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
callback	TUIRoomInviteeCallback	結果のコールバック。

cancelSpeechInvitation

キャスターがメンバーへの発言要請をキャンセルします。

```
- (void)cancelSpeechInvitation:(NSString *)userId
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
callback	TUIRoomActionCallback	結果のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

replySpeechInvitation

メンバーがキャスターの発言要請に同意/を拒否します。

```
- (void)replySpeechInvitation:(BOOL)agree
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
agree	BOOL	同意するかどうか。
callback	TUIRoomActionCallback	結果のコールバック。

sendSpeechApplication

メンバーが発言を要請します。

```
- (void)sendSpeechApplication:(TUIRoomInviteeCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomInviteeCallback	結果のコールバック。

cancelSpeechApplication

メンバーが発言申請をキャンセルします。

```
- (void)cancelSpeechApplication:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomActionCallback	結果のコールバック。

replySpeechApplication

キャスターがメンバーの発言申請に同意/を拒否します。

```
- (void)replySpeechApplication:(BOOL)agree
userId:(NSString *)userId
```

©2013-2019 Tencent Cloud. All rights reserved.

```
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
agree	BOOL	同意するかどうか
userId	NSString	ユーザーID。
callback	TUIRoomActionCallback	結果のコールバック。

forbidSpeechApplication

キャスターが発言申請を禁止します。

```
- (void)forbidSpeechApplication:(BOOL)forbid
callback:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
forbid	BOOL	禁止するかどうか。
callback	TUIRoomActionCallback	結果のコールバック。

sendOffSpeaker

キャスターがメンバーに発言の停止を命令します。

```
- (void)sendOffSpeaker:(NSString *)userId
callback:(TUIRoomInviteeCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
callback	TUIRoomInviteeCallback	結果のコールバック。

sendOffAllSpeakers

キャスターが全メンバーに発言の停止を命令します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)sendOffAllSpeakers:(TUIRoomInviteeCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomInviteeCallback	結果のコールバック。

exitSpeechState

メンバーが発言を停止し、視聴者になります。

```
- (void)exitSpeechState:(TUIRoomActionCallback)callback;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomActionCallback	結果のコールバック。

画面共有インターフェース

startScreenCapture

画面共有を開始。

```
- (void)startScreenCapture:(TRTCVideoEncParam *)encParam API_AVAILABLE(ios(11.0));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
encParams	TRTCVideoEncParam	画面共有時のエンコードパラメータを設定します。

> 説明：
> 詳細については、TRTC SDKをご参照ください。

stopScreenCapture

画面キャプチャの停止。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)stopScreenCapture API_AVAILABLE(ios(11.0));
```

美顔フィルタに関するインターフェース関数

getBeautyManager

美顔管理オブジェクトTXBeautyManagerを取得します。

```
- (TXBeautyManager *)getBeautyManager;
```

美顔管理では、次の機能を使用できます。

- 「美顔のスタイル」、「美白」、「肌色補正（血色・つや感）」、「デカ眼」、「顔痩せ」、「V顔」、「下あご」、「面長補正」、「小鼻」、「キラキラ目」、「白い歯」、「目の弛み除去」、「シワ除去」、「ほうれい線除去」などの美容効果を設定します。
- 「髪の生え際」、「眼と眼の距離」、「眼角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
- 人の顔のスタンプ（素材）等のダイナミック効果を設定します。
- メイクアップを追加します。
- ジェスチャー認識を行います。

関連設定インターフェース

setVideoQosPreference

ネットワークトラフィックコントロール関連パラメータを設定します。

```
- (void)setVideoQosPreference:(TRTCNetworkQosParam *)preference;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
preference	TRTCNetworkQosParam	ネットワークトラフィックコントロールポリシー。

setAudioQuality

音質の設定。

```
- (void)setAudioQuality:(TRTCAudioQuality)quality;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	TRTCAudioQuality	音質。詳細については、TRTC SDKをご参照ください。

setVideoResolution

解像度の設定。

```
- (void)setVideoResolution:(TRTCVideoResolution)resolution;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
resolution	TRTCVideoResolution	ビデオの解像度。詳細については、TRTC SDKをご参照ください。

setVideoFps

フレームレートの設定。

```
- (void)setVideoFps:(int)fps;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
fps	int	ビデオキャプチャのフレームレート。

説明：
推奨する値：15fpsまたは20fps。5fps以下ではラグ感が目立ち、10fps以下では軽微なラグ感があります。20fps以上は高すぎて浪費になります（映画のフレームレートは24fps）。

setVideoBitrate

ビットレートの設定。

```
- (void)setVideoBitrate:(int)bitrate;
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
bitrate	int	ビットレート。SDKは、目標ビットレートに応じてエンコードを行い、ネットワークの状態が良くない場合のみ、ビデオビットレートを動的に引き下げます。詳細については、TRTC SDKをご参照ください。

説明：
推奨する値：TRTCVideoResolutionの各クラスに注記する最適ビットレートをご参照ください。これをもとにより高いレートに適宜調整することも可能です。例えば、TRTC_VIDEO_RESOLUTION_1280_720に対応する目標ビットレートが1200kbpsであるならば、設定を1500kbpsにし、より鮮明な画像を得ることができます。

enableAudioEvaluation

音量レベルリマインダを有効にします。

```
- (void)enableAudioEvaluation:(BOOL)enable;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	BOOL	YES：オン、NO：オフ。

説明：
有効化すると、onUserVolumeUpdateの中でSDKの音量のボリュームに対する評価を取得できます。

setAudioPlayVolume

再生音量の設定。

```
- (void)setAudioPlayVolume:(NSInteger)volume;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	再生音量、0-100、 デフォルト100。

©2013-2019 Tencent Cloud. All rights reserved.

setAudioCaptureVolume

マイクの集音音量設定。

```
- (void)setAudioCaptureVolume:(NSInteger)volume;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、0-100、 デフォルト100。

startFileDumping

録音の開始。

```
- (void)startFileDumping:(TRTCAudioRecordingParams *)params;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
params	TRTCAudioRecordingParams	録音パラメータ。詳細については、TRTC SDKをご参照ください。

説明：
この方法で呼び出した後、 SDKは通話プロセスの中のすべての音声（ローカル音声、リモート音声、BGMなど）を1つのファイルにレコーディングします。ルームに参加しているか否かにかかわらず、このインターフェースを呼び出せば有効となります。leaveRoomを呼び出した時に録音中であれば、録音は自動的に停止します。

stopFileDumping

録音の停止。

```
- (void)stopFileDumping;
```

SDKバージョンインターフェースの取得

getSdkVersion

©2013-2019 Tencent Cloud. All rights reserved.

SDKバージョン情報を取得します。

```
- (NSInteger)getSdkVersion;
```

エラーイベントコールバック

onError

```
- (void)onError:(NSInteger)code message:(NSString *)message;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	NSInteger	エラーコード。
message	NSString	エラー情報。

基本イベントコールバック

onDestroyRoom

ルーム解散のコールバック。

```
- (void)onDestroyRoom;
```

onUserVoiceVolume

ユーザー音量の大きさのコールバック。

```
- (void)onUserVoiceVolume:(NSString *)userId volume:(NSInteger)volume;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
volume	NSInteger	ユーザーの音量の大きさ。値の範囲は0～100。

onRoomMasterChanged

©2013-2019 Tencent Cloud. All rights reserved.

キャスター変更のコールバック。

```
- (void)onRoomMasterChanged:(NSString *)previousUserId
currentUserId:(NSString *)currentUserId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
previousUserId	NSString	変更前のキャスターユーザーID。
currentUserId	NSString	変更後のキャスターユーザーID。

リモートユーザーコールバックイベント

onRemoteUserEnter

リモートユーザー入室コールバック。

```
- (void)onRemoteUserEnter:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。

onRemoteUserLeave

リモートユーザー退室コールバック。

```
- (void)onRemoteUserLeave:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。

onRemoteUserCameraAvailable

リモートユーザーが、カメラ、ビデオを起動しているかどうか。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onRemoteUserCameraAvailable:(NSString *)userId
available:(BOOL)available;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
available	BOOL	YES：ビデオストリームデータあり、NO：ビデオストリームデータなし。

onRemoteUserScreenVideoAvailable

メンバーのビデオ共有**オン/オフ**の通知。

```
- (void)onRemoteUserScreenVideoAvailable:(NSString *)userId
available:(BOOL)available;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
available	BOOL	画面共有ストリームデータの有無。

onRemoteUserAudioAvailable

リモートユーザーがオーディオアップストリームを開始したかどうかのコールバック。

```
- (void)onRemoteUserAudioAvailable:(NSString *)userId
available:(BOOL)available;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
available	BOOL	オーディオデータの有無。

onRemoteUserEnterSpeechState

リモートユーザーが発言を開始します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onRemoteUserEnterSpeechState:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。

onRemoteUserExitSpeechState

リモートユーザーが発言を終了します。

```
- (void)onRemoteUserExitSpeechState:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。

チャットルームメッセージイベントコールバック

onReceiveChatMessage

テキストメッセージの受信。

```
- (void)onReceiveChatMessage:(NSString *)userId message:(NSString *)message;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。
message	NSString	テキストメッセージ。

フィールドコントロールメッセージコールバック

onReceiveSpeechInvitation

ユーザーがキャスターの発言要請を受信する場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onReceiveSpeechInvitation:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	キャスターユーザーID。

onReceiveInvitationCancelled

ユーザーがキャスターの発言要請キャンセルを受信する場合のコールバック。

```
- (void)onReceiveInvitationCancelled:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	キャスターユーザーID。

OnReceiveSpeechApplication

キャスターがユーザーの発言要請を受信する場合のコールバック。

```
void onReceiveSpeechApplication(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。

onSpeechApplicationCancelled

ユーザーが発言申請をキャンセルする場合のコールバック。

```
- (void)onSpeechApplicationCancelled:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

onSpeechApplicationForbidden

キャスターが発言申請を禁止する場合のコールバック。

```
- (void)onSpeechApplicationForbidden:(BOOL)isForbidden userId:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isForbidden	BOOL	禁止するかどうか。
userId	NSString	ユーザーID。

onOrderedToExitSpeechState

メンバーが発言を停止するようリクエストされる場合のコールバック。

```
- (void)onOrderedToExitSpeechState:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	キャスターユーザーID。

onCallingRollStarted

キャスターが点呼を開始し、メンバーが受信する場合のコールバック。

```
- (void)onCallingRollStarted:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	キャスターユーザーID。

onCallingRollStopped

キャスターが点呼を終了し、メンバーが受信する場合のコールバック。

```
- (void)onCallingRollStopped:(NSString *)userId;
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	NSString	キャスターユーザーID。

onMemberReplyCallingRoll

メンバーが点呼に応答し、キャスターが受信する場合のコールバック。

```
- (void)onMemberReplyCallingRoll:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	ユーザーID。

onChatRoomMuted

キャスターがチャットルームのミュートを変更する場合のコールバック。

```
- (void)onChatRoomMuted:(BOOL)muted userId:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	BOOL	無効にするかどうか。
userId	NSString	キャスターユーザーID。

onMicrophoneMuted

キャスターがマイクの無効化を設定する場合のコールバック。

```
- (void)onMicrophoneMuted:(BOOL)muted userId:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	BOOL	無効にするかどうか。
userId	NSString	キャスターユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

onCameraMuted

キャスターがカメラの無効化を設定する場合のコールバック。

```
- (void)onCameraMuted:(BOOL)muted userId:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	BOOL	無効にするかどうか。
userId	NSString	キャスターユーザーID。

onReceiveKickedOff

キャスターによるキックアウトのコールバック。

```
- (void)onReceiveKickedOff:(NSString *)userId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	NSString	キャスター/管理者ユーザーID。

統計および品質コールバック

onStatistics

技術指標統計のコールバック。

```
- (void)onStatistics:(TRTCStatistics *)statistics;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
statis	TRTCStatistics	統計データ。

onNetworkQuality

ネットワーク状況のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onNetworkQuality:(TRTCQualityInfo *)localQuality remoteQuality:(NSArray *)remoteQuality;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
localQuality	TRTCQualityInfo	アップストリームネットワークの品質。
remoteQuality	NSArray<TRTCQualityInfo *>	ダウンストリームネットワークの品質。

説明：
詳細については、TRTC SDKをご参照ください。

画面共有イベントコールバック

onScreenCaptureStarted

画面共有開始のコールバック。

```
- (void)onScreenCaptureStarted;
```

onScreenCaptureStopped

画面共有停止のコールバック。

```
- (void)onScreenCaptureStopped:(NSInteger)reason;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
reason	NSInteger	停止の理由。0：ユーザーの自発的な停止。1：その他のアプリケーションに占有されたことによる停止。

©2013-2019 Tencent Cloud. All rights reserved.

TUIRoom(Android)

最終更新日：：2022-03-01 12:29:09

1. TUIRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- キャスターがルームを作成し、入室者はルームナンバーを入力した後に入室できます。
- 入室者の間で画面共有を行います。
- 各種のテキストメッセージとカスタムメッセージの送信をサポートします。

2. TUIRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実装プロセスについては、 多人数オーディオビデオルーム(Android)をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のオーディオビデオルームコンポーネントとして使用します。
- IM SDK：IM SDK を利用してチャットルームの機能（**IM SDKはAndroidバージョンを使用**）を実装します。

TUIRoom API概要

TUIRoomCore基本関数

API	説明
getInstance	シングルトンオブジェクトを取得します。
destroyInstance	シングルトンオブジェクトを破棄します。
setListener	イベントコールバックを設定します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（キャスターが呼び出し）。
destroyRoom	ルームの破棄（キャスターが呼び出し）。
enterRoom	入室（入室メンバーが呼び出し）。
leaveRoom	退室（その他のルームメンバーが呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
getRoomInfo	ルーム情報の取得。
getRoomUsers	ルーム内全メンバー情報の取得。
getUserInfo	特定ユーザーの情報の取得。
transferRoomMaster	キャスター権限の移転（キャスターが呼び出し）。

ローカルのオーディオビデオ操作インターフェース

API	説明
startCameraPreview	ローカルビデオのプレビュー画面を立ち上げます。
stopCameraPreview	ローカルのビデオキャプチャおよびプレビューを停止します。
startLocalAudio	マイクキャプチャを起動します。
stopLocalAudio	マイクキャプチャを停止します。
setVideoMirror	ローカル画面のミラーモードのプレビューを設定します。
setSpeaker	スピーカーの起動を設定します。

リモートユーザーに関するインターフェース

API	説明
startRemoteView	指定メンバーのリモートビデオ画面をサブスクリプションし再生します。
stopRemoteView	リモートビデオ画面のサブスクリプションをキャンセルし再生を停止します。

チャットメッセージ送信インターフェース

API	説明
sendChatMessage	チャットメッセージを送信します。
sendCustomMessage	カスタムメッセージを送信します。

フィールドコントロール関連インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
muteUserMicrophone	特定ユーザーのマイクを無効化/再有効化します。
muteAllUsersMicrophone	全ユーザーのマイクを無効化/再有効化し、ステータスをルーム情報に同期させます。
muteUserCamera	特定ユーザーのカメラを無効化/再有効化します。
muteAllUsersCamera	全ユーザーのカメラを無効化/再有効化し、ステータスをルーム情報に同期させます。
muteChatRoom	チャットルームのミュートを開始/停止します（キャスターが呼び出し）。
kickOffUser	ルーム内の特定ユーザーをリムーブします（キャスターが呼び出し）。
startCallingRoll	キャスターが点呼を開始します。
stopCallingRoll	キャスターが点呼を終了します。
replyCallingRoll	メンバーがキャスターの点呼に応答します。
sendSpeechInvitation	キャスターがメンバーに発言するようインビテーションを送信します。
cancelSpeechInvitation	キャスターがメンバーの発言のためのインビテーションをキャンセルします。
replySpeechInvitation	メンバーがキャスターの発言申請に同意/拒否します。
sendSpeechApplication	メンバーが発言を要請します。
replySpeechApplication	キャスターがメンバーの発言申請に同意/拒否します。
forbidSpeechApplication	キャスターが発言申請を禁止します。
sendOffSpeaker	キャスターがメンバーに発言を停止するよう命令します。
sendOffAllSpeakers	キャスターが全員に発言を停止するよう命令します。
exitSpeechState	メンバーは発言を停止し、視聴者になります。

画面共有インターフェース

API	説明
startScreenCapture	画面共有を開始。
stopScreenCapture	画面キャプチャの停止。

©2013-2019 Tencent Cloud. All rights reserved.

美顔フィルタに関するインターフェース関数

API	説明
getBeautyManager	美顔管理オブジェクトTXBeautyManager。を取得します。

関連設定インターフェース

API	説明
setVideoQosPreference	ネットワークトラフィックコントロール関連パラメータを設定します。

SDKバージョンインターフェース関数の取得

API	説明
getSDKVersion	SDKバージョンを取得します。

TUIRoomCoreListener API概要

エラーイベントコールバック

API	説明
onError	エラーのコールバック。

基本イベントコールバック

API	説明
onDestroyRoom	ルーム解散のコールバック。
onUserVoiceVolume	音量の大きさのコールバック。
onRoomMasterChanged	キャスター変更のコールバック。

リモートユーザーイベントコールバック

API	説明
onRemoteUserEnter	リモートユーザー入室コールバック。
onRemoteUserLeave	リモートユーザー退室コールバック。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onRemoteUserCameraAvailable	リモートユーザーがカメラビデオを起動するかどうかのコールバック。
onRemoteUserScreenVideoAvailable	リモートユーザーが画面共有を開始するかどうかのコールバック。
onRemoteUserAudioAvailable	リモートユーザーがオーディオのアップストリームを開始するかどうかのコールバック。
onRemoteUserEnterSpeechState	リモートユーザーの発言開始のコールバック。
onRemoteUserExitSpeechState	リモートユーザーの発言終了のコールバック。

メッセージイベントのコールバック

API	説明
onReceiveChatMessage	テキストメッセージ受信のコールバック。
onReceiveRoomCustomMsg	カスタムメッセージ受信のコールバック。

フィールドコントロールイベントコールバック

API	説明
onReceiveSpeechInvitation	ユーザーがキャスターの発言要請を受信した場合のコールバック。
onReceiveInvitationCancelled	ユーザーがキャスターの発言要請キャンセルを受信する場合のコールバック。
onReceiveSpeechApplication	キャスターがユーザーの発言要請を受信する場合のコールバック。
onSpeechApplicationCancelled	ユーザーが発言申請をキャンセルする場合のコールバック。
onSpeechApplicationForbidden	キャスターが発言申請を禁止する場合のコールバック。
onOrderedToExitSpeechState	メンバーが発言の停止をリクエストされる場合のコールバック。
onCallingRollStarted	キャスターが点呼を開始し、メンバーが受信する場合のコールバック。
onCallingRollStopped	キャスターが点呼を終了し、メンバーが受信する場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onMemberReplyCallingRoll	メンバーが点呼に応答し、キャスターが受信する場合のコールバック。
onChatRoomMuted	キャスターがチャットルームのミュートを変更する場合のコールバック。
onMicrophoneMuted	キャスターがマイクの無効化を設定する場合のコールバック。
onCameraMuted	キャスターがカメラの無効化を設定する場合のコールバック。
onReceiveKickedOff	メンバーがキャスターからキックアウトされた場合のコールバック。

統計および品質コールバック

API	説明
onStatistics	技術指標統計のコールバック。
onNetworkQuality	ネットワーク品質のコールバック。

画面共有関連コールバック

API	説明
onScreenCaptureStarted	画面共有開始のコールバック。
onScreenCaptureStopped	画面共有停止のコールバック。

TUIRoomCore基本関数

getInstance

TUIRoomCore シングルトンオブジェクトを取得します。

```
public static TUIRoomCore getInstance(Context context);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
context	Context	Androidコンテキスト。内部ではApplicationContextに変換してシステムAPIの呼び出しに使用します。

©2013-2019 Tencent Cloud. All rights reserved.

destroyInstance

```
void destroyInstance();
```

setListener

TUIRoomCore イベントコールバック。TUIRoomCoreListener を介してTUIRoomCoreの各種ステータス通知を取得できます。

```
void setListener(TUIRoomCoreListener listener);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
listener	TUIRoomCoreListener	イベントコールバッククラスを受信します。

createRoom

ルームの作成（キャスターが呼び出し）。

```
void createRoom(String roomId, TUIRoomCoreDef.SpeechMode speechMode, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	String	ルームID。ご自身でアサインし、一元管理する必要があります。
speechMode	TUIRoomCoreDef.SpeechMode	発言モード。
callback	TUIRoomCoreCallback.ActionCallback	ルームの作成結果のコールバック。

キャスターの通常の呼び出しフローは以下のとおりです。

1. **キャスター**が `createRoom()` を呼び出し、ルームを作成します。ルーム作成の成否は `TUIRoomCoreCallback.ActionCallback` でキャスターに通知されます。
2. **キャスター**が `startCameraPreview()` を呼び出し 、カメラキャプチャとプレビューを起動します。
3. **キャスター**が `startLocalAudio()` を呼び出し、ローカルマイクを起動します。

destroyRoom

©2013-2019 Tencent Cloud. All rights reserved.

ルームの破棄（キャスターが呼び出し）。キャスターは、ルームの作成後、この関数を呼び出して、ルームを破棄できます。

```
void destroyRoom(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	UIRoomCoreCallback.ActionCallback	ルームの破棄結果のコールバック。

enterRoom

入室（ルーム参加メンバーが呼び出し）。

```
void enterRoom(String roomId, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	String	ルームID。
callback	UIRoomCoreCallback.ActionCallback	結果のコールバック。

ルーム参加メンバーが入室する場合の通常の呼び出し手順は次のとおりです。

1. **入室メンバー**が `enterRoom` を呼び出し、roomIdを渡せば入室できます。
2. **入室メンバー**が `startCameraPreview()` を呼び出してカメラプレビューを起動し、 `startLocalAudio()` を呼び出して、マイクキャプチャを起動します。
3. **入室メンバー**が `onRemoteUserCameraAvailable` のイベントを受信し、 `startRemoteView()` を呼び出して、ビデオ再生を開始します。

leaveRoom

退室（入室メンバーが呼び出し）。

```
void leaveRoom(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	UIRoomCoreCallback.ActionCallback	結果のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

getRoomInfo

ルーム情報を取得します。

```
TUIRoomCoreDef.RoomInfo getRoomInfo();
```

getRoomUsers

ルームの全メンバー情報を取得します。

```
List getRoomUsers();
```

getUserInfo

ルームのメンバー情報を取得します。

```
void getUserInfo(String userId, TUIRoomCoreCallback.UserInfoCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
callback	UIRoomCoreCallback.UserInfoCallback	ルームメンバーの詳細情報のコールバック。

setSelfProfile

ユーザー情報を設定します。

```
void setSelfProfile(String userName, String avatarURL, TUIRoomCoreCallback.ActionCallback callbac
k);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ユーザーの氏名。
avatarURL	String	ユーザーのプロフィール画像URL。
callback	TUIRoomCoreCallback.ActionCallback	設定が成功したかどうかの結果のコールバック。

transferRoomMaster

©2013-2019 Tencent Cloud. All rights reserved.

グループを他のユーザーに引き渡します。

```
void transferRoomMaster(String userId, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

ローカルプッシュインターフェース

startCameraPreview

ローカルカメラプレビューを起動します。

```
void startCameraPreview(boolean isFront, TXCloudVideoView view);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFront	boolean	true：フロントカメラ、false：リアカメラ。
view	TXCloudVideoView	ビデオ画像をロードするウィジェット。

stopCameraPreview

ローカルカメラプレビューを停止します。

```
void stopCameraPreview();
```

startLocalAudio

マイクの集音開始。

```
void startLocalAudio(int quality);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
quality	int	キャプチャの音質： • TRTC_AUDIO_QUALITY_MUSIC • TRTC_AUDIO_QUALITY_DEFAULT • TRTC_AUDIO_QUALITY_SPEECH

stopLocalAudio

マイクの集音停止。

```
void stopLocalAudio();
```

setVideoMirror

ローカル画面のミラーモードのプレビューを設定します。

```
void setVideoMirror(int type);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
type	int	イメージタイプ。

setSpeaker

スピーカーの起動設定。

```
void setSpeaker(boolean isUseSpeaker);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isUseSpeaker	boolean	true：スピーカー、false：ヘッドホン。

リモートユーザーに関するインターフェース

startRemoteView

リモートユーザーのビデオストリームのサブスクリプション。

©2013-2019 Tencent Cloud. All rights reserved.

```
void startRemoteView(String userId, TXCloudVideoView view, TUIRoomCoreDef.SteamType streamType, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	再生が必要なユーザーのID。
view	TXCloudVideoView	ビデオ画像をロードするviewウィジェット。
streamType	TUIRoomCoreDef.SteamType	ストリームのタイプ。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

stopRemoteView

サブスクリプションをキャンセルし、リモートビデオ画面の再生を停止します。

```
void stopRemoteView(String userId, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	再生停止が必要なユーザーのID。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

switchCamera

フロント/リアカメラを切り替えます。

```
void switchCamera(boolean isFront);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFront	boolean	true：フロントカメラ、false：リアカメラ。

メッセージ送信インターフェース

©2013-2019 Tencent Cloud. All rights reserved.

sendChatMessage

ルーム内でテキストメッセージをブロードキャストします。通常、テキストによるチャットに使用します。

```
void sendChatMessage(String message, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	メッセージの内容。
callback	TUIRoomCoreCallback.ActionCallback	送信結果のコールバック。

sendCustomMessage

カスタムメッセージを送信します。

```
void sendCustomMessage(String data, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
data	String	メッセージの内容。
callback	TUIRoomCoreCallback.ActionCallback	送信結果のコールバック。

フィールドコントロール関連インターフェース

muteUserMicrophone

特定ユーザーのマイクを無効化/再有効化します。

```
void muteUserMicrophone(String userId, boolean mute, TUIRoomCoreCallback.ActionCallback callback)
;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
mute	boolean	無効にするかどうか。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

muteAllUsersMicrophone

全ユーザーのマイクを無効化/再有効化します。

```
void muteAllUsersMicrophone(boolean mute, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	無効にするかどうか。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

muteUserCamera

特定ユーザーのカメラを無効化/再有効化します。

```
void muteUserCamera(String userId, boolean mute, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
mute	boolean	無効にするかどうか。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

muteAllUsersCamera

全ユーザーのカメラを無効化/再有効化します。

```
void muteAllUsersCamera(boolean mute, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	無効にするかどうか。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

muteChatRoom

テキストチャットのミュート/再有効化。

```
void muteChatRoom(boolean mute, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	無効にするかどうか。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

kickOffUser

キャスターがキックアウトします。

```
void kickOffUser(String userId, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

startCallingRoll

キャスターが点呼を開始します。

```
void startCallingRoll(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

stopCallingRoll

キャスターが点呼を終了します。

```
void stopCallingRoll(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

replyCallingRoll

メンバーがキャスターの点呼に応答します。

```
void replyCallingRoll(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

sendSpeechInvitation

キャスターがメンバーの発言を要請します。

```
void sendSpeechInvitation(String userId, TUIRoomCoreCallback.InvitationCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
callback	TUIRoomCoreCallback.InvitationCallback	結果のコールバック。

cancelSpeechInvitation

キャスターがメンバーへの発言要請をキャンセルします。

```
void cancelSpeechInvitation(String userId, TUIRoomCoreCallback.ActionCallback callback);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

replySpeechInvitation

メンバーがキャスターの発言要請に同意/を拒否します。

```
void replySpeechInvitation(boolean agree, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
agree	boolean	同意するかどうか。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

sendSpeechApplication

メンバーが発言を要請します。

```
void sendSpeechApplication(TUIRoomCoreCallback.InvitationCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.InvitationCallback	結果のコールバック。

cancelSpeechApplication

メンバーが発言申請をキャンセルします。

```
void cancelSpeechApplication(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

replySpeechApplication

キャスターがメンバーの発言申請に同意/を拒否します。

```
void replySpeechApplication(boolean agree, String userId, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
agree	boolean	同意するかどうか。
userId	String	ユーザーID。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

forbidSpeechApplication

キャスターが発言申請を禁止します。

```
void forbidSpeechApplication(boolean forbid, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
forbid	boolean	禁止するかどうか。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

sendOffSpeaker

キャスターがメンバーに発言の停止を命令します。

```
void sendOffSpeaker(String userId, TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

sendOffAllSpeakers

©2013-2019 Tencent Cloud. All rights reserved.

キャスターが全メンバーに発言の停止を命令します。

```
void sendOffAllSpeakers(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

exitSpeechState

メンバーが発言を停止し、視聴者になります。

```
void exitSpeechState(TUIRoomCoreCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	TUIRoomCoreCallback.ActionCallback	結果のコールバック。

画面共有インターフェース

startScreenCapture

画面共有を開始。

```
void startScreenCapture(TRTCCloudDef.TRTCVideoEncParam encParams, TRTCCloudDef.TRTCScreenShareParams screenShareParams);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
encParams	TRTCCloudDef.TRTCVideoEncParam	画面共有時のエンコードパラメータを設定します。上記の推奨設定を採用することをお勧めします。encParamsにnullを指定した場合、startScreenCaptureを呼び出す前のエンコードパラメータ設定が使用されます。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
screenShareParams	TRTCCloudDef.TRTCScreenShareParams	画面共有の特殊なレイアウト設定については、その中のfloatingViewの設定を推奨します。一方で、Appがシステムから強制排除されるのを回避でき、もう一方で、ユーザーのプライバシー保護にも役立ちます。

説明：
詳細については、TRTC SDKをご参照ください。 |

stopScreenCapture

画面キャプチャの停止。

```
void stopScreenCapture();
```

美顔フィルタに関するインターフェース関数

getBeautyManager

美顔管理オブジェクト TXBeautyManagerを取得します。

```
TXBeautyManager getBeautyManager();
```

美顔管理では、次の機能を使用できます。

- 「美顔のスタイル」、「美白」、「肌色補正（血色・つや感）」、「デカ眼」、「顔痩せ」、「V顔」、「下あご」、「面長補正」、「小鼻」、「キラキラ目」、「白い歯」、「目の弛み除去」、「シワ除去」、「ほうれい線除去」などの美容効果を設定します。
- 「髪の生え際」、「眼と眼の距離」、「眼角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
- 人の顔のスタンプ（素材）等のダイナミック効果を設定します。
- メイクアップを追加します。
- ジェスチャー認識を行います。

©2013-2019 Tencent Cloud. All rights reserved.

関連設定インターフェース

setVideoQosPreference

ネットワークトラフィックコントロール関連パラメータを設定します。

```
void setVideoQosPreference(TRTCCloudDef.TRTCNetworkQosParam preference);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
preference	TRTCCloudDef.TRTCNetworkQosParam	ネットワークトラフィックコントロールポリシー。

setAudioQuality

音質の設定。

```
void setAudioQuality(int quality);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	int	オーディオ品質。詳細については、TRTC SDKをご参照ください。

setVideoResolution

解像度の設定。

```
void setVideoResolution(int resolution);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
resolution	int	ビデオの解像度。詳細については、TRTC SDKをご参照ください。

setVideoFps

フレームレートの設定。

```
void setVideoFps(int fps);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
fps	int	ビデオキャプチャのフレームレート。

説明：
推奨する値：15fpsまたは20fps。5fps以下ではラグ感が目立ち、10fps以下では軽微なラグ感があります。20fps以上は高すぎて浪費になります（映画のフレームレートは24fps）。

setVideoBitrate

ビットレートの設定。

```
void setVideoBitrate(int bitrate);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
bitrate	int	ビットレート。SDKは、目標ビットレートに応じてエンコードを行い、ネットワークの状態が良くない場合のみ、ビデオビットレートを動的に引き下げます。詳細については、TRTC SDKをご参照ください。

説明：
推奨する値：TRTCVideoResolutionの各クラスに注記する最適ビットレートをご参照ください。これをもとにより高いレートに適宜調整することも可能です。例えば、TRTC_VIDEO_RESOLUTION_1280_720に対応する目標ビットレートが1200kbpsであるならば、設定を1500kbpsにし、より鮮明な画像を得ることができます。

enableAudioEvaluation

音量レベルリマインダを有効にします。

```
void enableAudioEvaluation(boolean enable);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
enable	boolean	true：オン、false：オフ。

説明：
有効化すると、onUserVolumeUpdateの中でSDKの音量のボリュームに対する評価を取得できます。

setAudioPlayVolume

再生音量の設定。

```
void setAudioPlayVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	再生音量、0-100、 デフォルト100。

setAudioCaptureVolume

マイクの集音音量設定。

```
void setAudioCaptureVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、0-100、 デフォルト100。

startFileDumping

録音の開始。

```
void startFileDumping(TRTCCloudDef.TRTCAudioRecordingParams trtcAudioRecordingParams);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
trtcAudioRecordingParams	TRTCCloudDef.TRTCAudioRecordingParams	録音パラメータ。詳細については、TRTC SDKをご参照ください。

説明：
この方法で呼び出した後、 SDKは通話プロセスの中のすべての音声（ローカル音声、リモート音声、BGMなど）を1つのファイルにレコーディングします。ルームに参加しているか否かにかかわらず、このインターフェースを呼び出せば有効となります。leaveRoomを呼び出した時に録音中であれば、録音は自動的に停止します。

stopFileDumping

録音の停止。

```
void stopFileDumping();
```

SDKバージョンインターフェースの取得

getSdkVersion

SDKバージョン情報を取得します。

```
int getSdkVersion();
```

エラーイベントコールバック

onError

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラーメッセージ。

基本イベントコールバック

onDestroyRoom

ルーム解散のコールバック。

```
void onDestroyRoom();
```

onUserVoiceVolume

ユーザー音量の大きさのコールバック。

```
void onUserVoiceVolume(String userId, int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
volume	int	ユーザーの音量の大きさ、値の範囲0～100。

onRoomMasterChanged

キャスター変更のコールバック。

```
void onRoomMasterChanged(String previousUserId, String currentUserId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
previousUserId	String	変更前のキャスターユーザーID。
currentUserId	String	変更後のキャスターユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

リモートユーザーコールバックイベント

onRemoteUserEnter

リモートユーザー入室コールバック。

```
void onRemoteUserEnter(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

onRemoteUserLeave

リモートユーザー退室コールバック。

```
void onRemoteUserLeave(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

onRemoteUserCameraAvailable

リモートユーザーが、カメラ、ビデオを起動しているかどうか。

```
void onRemoteUserCameraAvailable(String userId, boolean available);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
available	boolean	true：ビデオストリームデータあり、false：ビデオストリームデータなし。

onRemoteUserScreenVideoAvailable

メンバーのビデオ共有**オン/オフ**の通知。

```
void onRemoteUserScreenVideoAvailable(String userId, boolean available);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
available	boolean	画面共有ストリームデータの有無。

onRemoteUserAudioAvailable

リモートユーザーがオーディオアップストリームを開始したかどうかのコールバック。

```
void onRemoteUserAudioAvailable(String userId, boolean available);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
available	boolean	オーディオデータの有無。

onRemoteUserEnterSpeechState

リモートユーザーが発言を開始します。

```
void onRemoteUserEnterSpeechState(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

onRemoteUserExitSpeechState

リモートユーザーが発言を終了します。

```
void onRemoteUserExitSpeechState(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	String	ユーザーID。

チャットルームメッセージイベントコールバック

onReceiveChatMessage

テキストメッセージの受信。

```
void onReceiveChatMessage(String userId, String message);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
message	String	テキストメッセージ。

onReceiveRoomCustomMsg

カスタムメッセージの受信。

```
void onReceiveRoomCustomMsg(String userId, String data);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
message	String	カスタムメッセージ。

フィールドコントロールメッセージコールバック

onReceiveSpeechInvitation

ユーザーがキャスターの発言要請を受信する場合のコールバック。

```
void onReceiveSpeechInvitation(String userId);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	キャスターユーザーID。

onReceiveInvitationCancelled

ユーザーがキャスターの発言要請キャンセルを受信する場合のコールバック。

```
void onReceiveInvitationCancelled(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	キャスターユーザーID。

onReceiveSpeechApplication

キャスターがユーザーの発言要請を受信する場合のコールバック。

```
void onReceiveSpeechApplication(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

onSpeechApplicationCancelled

ユーザーが発言申請をキャンセルする場合のコールバック。

```
void onSpeechApplicationCancelled(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

onSpeechApplicationForbidden

キャスターが発言申請を禁止する場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

```
void onSpeechApplicationForbidden(boolean isForbidden);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isForbidden	boolean	禁止するかどうか。

onOrderedToExitSpeechState

メンバーが発言を停止するようリクエストされる場合のコールバック。

```
void onOrderedToExitSpeechState(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	キャスターユーザーID。

onCallingRollStarted

キャスターが点呼を開始し、メンバーが受信する場合のコールバック。

```
void onCallingRollStarted(String userId);
```

onCallingRollStopped

キャスターが点呼を終了し、メンバーが受信する場合のコールバック。

```
void onCallingRollStopped(String userId);
```

onMemberReplyCallingRoll

メンバーが点呼に応答し、キャスターが受信する場合のコールバック。

```
void onMemberReplyCallingRoll(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

onChatRoomMuted

キャスターがチャットルームのミュートを変更する場合のコールバック。

```
void onChatRoomMuted(boolean muted);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	boolean	無効にするかどうか。

onMicrophoneMuted

キャスターがマイクの無効化を設定する場合のコールバック。

```
void onMicrophoneMuted(boolean muted);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	boolean	無効にするかどうか。

onCameraMuted

キャスターがカメラの無効化を設定する場合のコールバック。

```
void onCameraMuted(boolean muted);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	boolean	無効にするかどうか。

onReceiveKickedOff

キャスターによるキックアウトのコールバック。

```
void onReceiveKickedOff(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	String	キャスター/管理者ユーザーID。

統計および品質コールバック

onStatistics

技術指標統計のコールバック。

```
void onStatistics(TRTCStatistics statistics);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
statis	TRTCStatistics	統計データ。

onNetworkQuality

ネットワーク状況のコールバック。

```
void onNetworkQuality(TRTCCloudDef.TRTCQuality localQuality, List remoteQuality);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
localQuality	TRTCCloudDef.TRTCQuality	アップストリームのネットワーク品質。
remoteQuality	List<TRTCCloudDef.TRTCQuality>	ダウンストリームのネットワーク品質。

説明：
詳細については、 TRTC SDKをご参照ください。

画面共有イベントコールバック

onScreenCaptureStarted

画面共有開始のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

```
void onScreenCaptureStarted();
```

onScreenCaptureStopped

画面共有停止のコールバック。

```
void onScreenCaptureStopped(int reason);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
reason	int	停止の理由。0：ユーザーの自発的な停止。1：その他アプリケーションに占有されたことによる停止。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCMeeting(Flutter)

最終更新日：：2022-03-01 12:18:48

TRTCMeetingは、Tencent CloudのTRTCおよびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- キャスターがミーティングルームを作成し、参加者はルームナンバーを入力した後にミーティングに参加します。
- 参加者の間で画面共有を行います。
- 各種のテキストメッセージとカスタムメッセージの送信をサポートします。

TRTCMeetingは、1つのオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実現プロセスは、多人数オーディオビデオルーム（Flutter）をご参照ください。

- TRTC SDK： TRTC SDK を低遅延のビデオミーティングのコンポーネントとして使用します。
- IM SDK：IM SDKのMeetingRoomを利用して、ミーティング中のチャットルームの機能を実現します。

TRTCMeeting API概要

SDK基本インターフェース

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを破棄します。
registerListener	イベント監視を設定します。
unRegisterListener	イベント監視を破棄します。
login	ログイン。
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ミーティングルームに関するインターフェース

API	説明
createMeeting	ルームの作成（キャスターが呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
destroyMeeting	ルームの破棄（キャスターが呼び出し）。
enterMeeting	入室（参加者が呼び出し）。
leaveMeeting	退室（参加者が呼び出し）。
getUserInfoList	ルーム内の全メンバーのリストを取得します。enterMeeting()成功後に呼び出しが有効となります。
getUserInfo	ルーム内の指定メンバーの詳細情報を取得します。enterMeeting()成功後に呼び出しが有効となります。

リモートユーザーに関するインターフェース

API	説明
startRemoteView	指定メンバーのリモートビデオ画面を再生します。
stopRemoteView	指定メンバーのリモートビデオ画面の再生を停止します。
setRemoteViewParam	指定メンバーのリモート画像のレンダリングパラメータを設定します。
muteRemoteAudio	指定メンバーのリモート音声をミュート/ミュート取り消しにします。
muteAllRemoteAudio	全メンバーのリモート音声をミュート/ミュート取り消しにします。
muteRemoteVideoStream	指定メンバーのリモートビデオを一時停止/再開します。
muteAllRemoteVideoStream	全メンバーのリモートビデオストリームを一時停止/再開します。

ローカルのビデオ操作インターフェース

API	説明
startCameraPreview	ローカルビデオのプレビュー画面を立ち上げます。
stopCameraPreview	ローカルのビデオキャプチャおよびプレビューを停止します。
switchCamera	フロント/リアカメラを切り替えます。
setVideoEncoderParam	ビデオエンコーダ関連パラメータを設定します。
setLocalViewMirror	ローカル画面のミラーモードのプレビューを設定します。

©2013-2019 Tencent Cloud. All rights reserved.

ローカルの音声操作インターフェース

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
muteLocalAudio	ローカル音声のミュート起動/停止。
setSpeaker	スピーカーまたはヘッドホン起動の設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。
startAudioRecording	録音の開始。
stopAudioRecording	録音の停止。
enableAudioVolumeEvaluation	音声レベルリマインダを有効にします。

スクリーンキャプチャのインターフェース

API	説明
startScreenCapture	画面共有を開始。
stopScreenCapture	画面キャプチャの停止。
pauseScreenCapture	スクリーンキャプチャの一時停止。
resumeScreenCapture	スクリーンキャプチャの再開。

管理オブジェクト取得に関するインターフェース

API	説明
getDeviceManager	デバイス管理オブジェクトTXDeviceManagerを取得します。
getBeautyManager	美顔管理オブジェクトTXBeautyManagerを取得します。

メッセージ送信関連インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
sendRoomTextMsg	ミーティング中にテキストメッセージをブロードキャストします。通常、チャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

TRTCLiveRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onKickedOffline	他のユーザーが同じアカウントでログインすると、キックアウトされてオフラインになります。

ミーティングルームのイベントコールバック

API	説明
onRoomDestroy	ミーティングルームが破棄された時のコールバック。
onNetworkQuality	ネットワーク状態のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

参加者入退室のイベントコールバック

API	説明
onEnterRoom	ローカルの入室のコールバック。
onLeaveRoom	ローカルの退室のコールバック。
onUserEnterRoom	新しい参加者の入室のコールバック。
onUserLeaveRoom	参加者の退出のコールバック。

参加者の音声・ビデオのイベントコールバック

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onUserAudioAvailable	参加者がマイクを起動/停止したときのコールバック。
onUserVideoAvailable	参加者がカメラを起動/停止したときのコールバック。
onUserSubStreamAvailable	参加者がサブチャンネル画面を起動/停止したときのコールバック。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージ受信のコールバック。
onRecvRoomCustomMsg	カスタムメッセージ受信のコールバック。

スクリーンキャプチャのイベントコールバック

API	説明
onScreenCaptureStarted	スクリーンキャプチャ開始のコールバック。
onScreenCapturePaused	スクリーンキャプチャー時停止のコールバック。
onScreenCaptureResumed	スクリーンキャプチャ再開のコールバック。
onScreenCaptureStoped	スクリーンキャプチャ停止のコールバック。

SDK基本インターフェース

sharedInstance

TRTCMeetingシングルトンオブジェクトを取得します。

```
static Future sharedInstance();
```

destroySharedInstance

TRTCMeetingシングルトンオブジェクトを破棄します。

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

©2013-2019 Tencent Cloud. All rights reserved.

説明：
インスタンスを破棄すると、外部キャッシュのTRTCMeetingインスタンスは再利用できなくなります。あらためてsharedInstanceを呼び出し、新しいインスタンスを取得する必要があります。

registerListener

イベント監視を設定します。TRTCMeetingDelegateを介して、TRTCMeetingの各種ステータスの通知を取得できます。

```
void registerListener(MeetingListenerFunc func);
```

説明：
funcはTRTCMeetingのプロキシコールバックです。

unRegisterListener

イベント監視を破棄します。

```
void unRegisterListener(MeetingListenerFunc func);
```

login

ログイン

```
Future login(int sdkAppId, String userId, String userSig);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	TRTCコンソール > 【アプリケーション管理】> アプリケーション情報の中でSDKAppIDを確認できます。
userId	String	現在のユーザーのID。文字列タイプ。アルファベット（a-z および A-Z）、数字（0-9）、ハイフン（-）、アンダーバー（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算、使用方法をご参照ください。

©2013-2019 Tencent Cloud. All rights reserved.

logout

ログアウト。

```
Future logout();
```

setSelfProfile

個人情報の修正。

```
Future setSelfProfile(String userName, String avatarURL);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ユーザーニックネーム。
avatarURL	String	ユーザープロフィール画像のアドレス。

ミーティングルーム関連インターフェース

createMeeting

ミーティングの作成（キャスターの呼び出し）。

```
Future createMeeting(int roomId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ミーティングルームID。ご自身でアサインし、一括管理する必要があります。

キャスターの通常の呼び出しフローは以下のとおりです。

1. 【キャスター】 `createMeeting()` を呼び出し、 `roomId` を渡してミーティングを作成します。ミーティングルーム作成の成功の有無が `ActionCallback` によってキャスターに通知されます。
2. 【キャスター】 `startCameraPreview()` を呼び出し、カメラのプレビューを起動します。この時美顔パラメータを調整できます。
3. 【キャスター】 `startMicrophone()` を呼び出し、マイクキャプチャを起動します。

destroyMeeting

©2013-2019 Tencent Cloud. All rights reserved.

ミーティングルームを破棄します（キャスターが呼び出し）。キャスターは、ミーティング作成後、この関数を呼び出してミーティングを破棄できます。

```
Future destroyMeeting(int roomId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ミーティングルームID。ご自身でアサインし、一括管理する必要があります。

enterMeeting

ミーティングルームに参加します（参加者が呼び出し）。

```
Future enterMeeting(int roomId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ミーティングルームID。

参加者がミーティングに参加する通常の呼び出しフローは以下のとおりです。

1. 【参加者】 `enterMeeting()` を呼び出し、 `roomId` を渡せばミーティングルームに入れます。
2. 【参加者】 `startCameraPreview()` を呼び出して、カメラのプレビューを起動し、 `startMicrophone()` を呼び出し、マイクキャプチャを起動します。
3. 【参加者】 `onUserVideoAvailable` のイベントを受信します。 `startRemoteView()` を呼び出し、メンバーのuserIdを渡して再生を開始します

leaveMeeting

ミーティングルームから退出します（参加者が呼び出し）。

```
Future leaveMeeting();
```

getUserInfoList

ルーム内の全メンバーのリストを取得します。enterMeeting()成功後に呼び出しが有効となります。

```
Future getUserInfoList(List<String> userIdList);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIdList	List	取得する必要があるuserId リスト。

getUserInfo

ルーム内の指定メンバーの詳細情報を取得します。enterMeeting()成功後に呼び出しが有効となります。

```
Future getUserInfo(String userId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定メンバーのID。

リモートユーザーに関するインターフェース

startRemoteView

指定メンバーのリモートビデオ画面を再生します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定メンバーのID。
streamType	int	視聴したいビデオストリームのタイプ。詳細については、TRTC SDKをご参照ください。
viewId	int	TRTCCloudVideoViewが発行するviewId。

stopRemoteView

指定メンバーのリモートビデオ画面の再生を停止します。

```
Future<void> stopRemoteView(String userId, int streamType);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	String	指定メンバーのID。
streamType	int	視聴したいビデオストリームのタイプ。詳細については、TRTC SDKをご参照ください。

setRemoteViewParam

指定メンバーのリモート画像のレンダリングパラメータを設定します。

```
Future<void> setRemoteViewParam(String userId, int streamType,
{int fillMode, int rotation, int mirrorType});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定メンバーのID。
streamType	int	視聴したいビデオストリームのタイプ。詳細については、TRTC SDKをご参照ください。
fillMode	int	画像レンダリングモード。FILL（デフォルト）またはFITモード。詳細については、TRTC SDKをご参照ください。
rotation	int	画像の時計回りの回転角度。詳細については、TRTC SDKをご参照ください。
mirrorType	int	イメージモード。詳細については、 TRTC SDKをご参照ください。

muteRemoteAudio

指定メンバーのリモート音声をミュート/ミュート取り消しにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定メンバーのID。
mute	boolean	true：ミュート、false：ミュート停止。

muteAllRemoteAudio

©2013-2019 Tencent Cloud. All rights reserved.

全メンバーのリモート音声をミュート/ミュート取り消しにします。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート、false：ミュート停止。

muteRemoteVideoStream

指定メンバーのリモートビデオを一時停止/再開します。

```
Future<void> muteRemoteVideoStream(String userId, bool mute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定メンバーのID。
mute	boolean	true：一時停止、false：再開。

muteAllRemoteVideoStream

全メンバーのリモートビデオストリームを一時停止/再開します。

```
Future<void> muteAllRemoteVideoStream(bool mute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：一時停止、false：再開。

ローカルのビデオ操作インターフェース

startCameraPreview

ローカルビデオのプレビュー画面を立ち上げます。

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

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFront	boolean	true：フロントカメラ、false：リアカメラ。
viewId	int	TRTCCloudVideoViewが発行するviewId。

stopCameraPreview

ローカルのビデオキャプチャおよびプレビューを停止します。

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

switchCamera

フロント/リアカメラを切り替えます。

```
Future<void> switchCamera(bool isFront);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isFront	boolean	true：フロントカメラ、false：リアカメラ。

setVideoEncoderParam

ビデオエンコーダ関連パラメータを設定します。

```
Future<void> setVideoEncoderParam({
int videoFps,
int videoBitrate,
int videoResolution,
int videoResolutionMode,
});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
videoFps	int	ビデオキャプチャのフレームレート。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
videoBitrate	int	ビットレート。SDKは目標ビットレートに応じてエンコードを行い、ネットワークの状態が良くない場合のみ、ビデオビットレートを動的に引き下げます。
videoResolution	int	ビデオ解像度。
videoResolutionMode	int	解像度モード。

説明：
詳細については、TRTC SDKをご参照ください。

setLocalViewMirror

ローカル画面のミラーモードのプレビューを設定します。

```
Future<void> setLocalViewMirror(bool isMirror);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMirror	boolean	イメージプレビューモードの起動の有無。true：オン、false：オフ。

ローカル音声操作のインターフェース

startMicrophone

マイクの集音開始。

```
Future<void> startMicrophone({int quality});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	int	オーディオ品質。詳細については、 TRTC SDKをご参照ください。

stopMicrophone

©2013-2019 Tencent Cloud. All rights reserved.

マイクの集音停止。

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

muteLocalAudio

ローカル音声のミュート起動/停止。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート、false：ミュート取り消し。

setSpeaker

スピーカーまたはヘッドホンの起動設定。

```
Future<void> setSpeaker(bool useSpeaker);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
useSpeaker	boolean	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。

```
Future<void> setAudioCaptureVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、値：0 - 100、デフォルト値：100。

setAudioPlayoutVolume

再生音量の設定。

©2013-2019 Tencent Cloud. All rights reserved.

```
Future<void> setAudioPlayoutVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	再生音量、値：0 - 100、デフォルト値：100。

startAudioRecording

録音の開始。

```
Future<int?> startAudioRecording(String filePath);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
filePath	String	録音ファイルの保存パス。このパスはユーザーが自身で指定する必要があります。パスが存在し、記述できることを確認してください。このパスはファイル名およびフォーマットの拡張子まで正確に示す必要があります。フォーマットの拡張子によって録音ファイルの形式が決まります。現在サポートしている形式はPCM、WAV、AACです。

stopAudioRecording

録音の停止。

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

enableAudioVolumeEvaluation

音量レベルリマインダを有効にします。

```
Future<void> enableAudioVolumeEvaluation(int intervalMs);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
intervalMs	int	onUserVoiceVolume コールバックをトリガーするインターバルを決定します。単位はms。最小インターバルは100ms。0以下の場合、コールバックは停止します。設定を300msにすることを推奨します。

©2013-2019 Tencent Cloud. All rights reserved.

スクリーンキャプチャのインターフェース

startScreenCapture

画面共有を開始。

```
Future<void> startScreenCapture({
int videoFps,
int videoBitrate,
int videoResolution,
int videoResolutionMode,
String appGroup,
});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
videoFps	int	ビデオキャプチャのフレームレート。
videoBitrate	int	ビットレート。SDKは目標ビットレートに応じてエンコードを行い、ネットワークの状態が良くない場合のみ、ビデオビットレートを動的に引き下げます。
videoResolution	int	ビデオ解像度。
videoResolutionMode	int	解像度モード。
appGroup	String	このパラメータはiOS端末でのみ有効となり、Android端末の場合、このパラメータを気にする必要はありません。このパラメータはメインAppとBroadcastが共有するApplication Group Identifierです。

> 説明：
> 詳細については、TRTC SDKをご参照ください。

stopScreenCapture

画面キャプチャの停止。

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

pauseScreenCapture

©2013-2019 Tencent Cloud. All rights reserved.

スクリーンキャプチャの一時停止。

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

resumeScreenCapture

スクリーンキャプチャの再開。

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

管理オブジェクト取得に関するインターフェース

getDeviceManager

デバイス管理オブジェクト TXDeviceManagerを取得します。

```
getDeviceManager();
```

getBeautyManager

美顔管理オブジェクトTXBeautyManagerを取得します。

```
getBeautyManager();
```

美顔管理では、次の機能を使用できます。

- 「美顔のスタイル」、「美白」、「肌色補正（血色・つや感）」、「デカ眼」、「顔痩せ」、「V顔」、「下あご」、「面長補正」、「小鼻」、「キラキラ目」、「白い歯」、「目の弛み除去」、「シワ除去」、「ほうれい線除去」などの美容効果を設定します。
- 「髪の生え際」、「眼と眼の距離」、「眼角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
- 人の顔のスタンプ（素材）等のダイナミック効果を設定します。
- メイクアップを追加します。
- ジェスチャー認識を行います。

メッセージ送信関連インターフェース

sendRoomTextMsg

ミーティング中にテキストメッセージをブロードキャストします。通常、チャットに使用します。

©2013-2019 Tencent Cloud. All rights reserved.

```
Future sendRoomTextMsg(String message);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

```
Future sendRoomCustomMsg(String cmd, String message);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	コマンドワード。開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。

TRTCMeetingDelegateイベントコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
errCode	int	エラーコード。
errMsg	String	エラー情報。
extraInfo	String	拡張情報フィールド。個別のエラーコードにより問題の特定に役立つ追加情報をもたせることが可能です。

onWarning

警告のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
warningCode	int	エラーコード。
warningMsg	String	警告情報。
extraInfo	String	拡張情報フィールド。個別のエラーコードにより問題の特定に役立つ追加情報をもたせることが可能です。

onKickedOffline

他のユーザーが同じアカウントでログインすると、キックアウトされてオフラインになります。

ミーティングルームのイベントコールバック

onRoomDestroy

ミーティングルームが破棄された時のコールバック。キャスターの退室時、ルーム内の全ユーザーがこの通知を受信します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	String	ミーティングルームID。

onNetworkQuality

ネットワーク状態のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
localQuality	TRTCCloudDef.TRTCQuality	アップストリームのネットワーク品質。
remoteQuality	List<TRTCCloudDef.TRTCQuality>	ダウンストリームのネットワーク品質。

onUserVolumeUpdate

ユーザーの通話音量のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userVolumes	List	発言中のすべてのルーム参加者の音量。値の範囲：0～100。
totalVolume	int	すべてのリモート参加者の総音量。値の範囲：0～100。

参加者入退室のイベントコールバック

onEnterRoom

ローカルの入室のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
result	int	0を上回るときは入室の所要時間（ms）、0未満のときは入室のエラーコードです。

onLeaveRoom

ローカルの退室のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
reason	int	ミーティング退出の原因。0：自主的にleaveMeetingを呼び出して退出。1：サーバーによる現在のミーティングからのキックアウト。2：現在のミーティング全体の解散。

©2013-2019 Tencent Cloud. All rights reserved.

onUserEnterRoom

新しい参加者の入室のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	新しい参加者のユーザーID。

onUserLeaveRoom

参加者の退出のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	退出した参加者のユーザーID。

参加者の音声・ビデオのイベントコールバック

onUserAudioAvailable

参加者のマイク起動/停止のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
available	boolean	true：ユーザーがマイクをオンにします。false：ユーザーがマイクをオフにします。

onUserVideoAvailable

参加者によるカメラ起動/停止のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
available	boolean	true：ユーザーがカメラをオンにします。false：ユーザーがカメラをオフにします。

onUserSubStreamAvailable

参加者がサブチャンネル画面を起動/停止したときのコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
available	boolean	true：ユーザーがサブチャンネル画面を起動。false：ユーザーがサブチャンネル画面を停止。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージ受信のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
sendId	String	送信者のユーザーID。
userAvatar	String	送信者のユーザープロフィール画像。
userName	String	送信者のユーザーニックネーム。

onRecvRoomCustomMsg

カスタムメッセージ受信のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
command	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
sendId	String	送信者のユーザーID。
userAvatar	String	送信者のユーザープロフィール画像。
userName	String	送信者のユーザーニックネーム。

スクリーンキャプチャのイベントコールバック

onScreenCaptureStarted

スクリーンキャプチャ開始のコールバック。

onScreenCapturePaused

スクリーンキャプチャー時停止のコールバック。

onScreenCaptureResumed

スクリーンキャプチャ再生のコールバック。

onScreenCaptureStoped

スクリーンキャプチャ停止のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
reason	int	停止の理由。0：ユーザーの自発的な停止。1：スクリーンウィンドウが閉じられたことによるによる停止。

©2013-2019 Tencent Cloud. All rights reserved.

TUIRoom(Windows&Mac)

最終更新日：：2022-03-01 12:24:01

TUIRoomは、Tencent CloudのTencent Real-Time Communication（TRTC）およびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- キャスターがルームを作成し、参加者はルームナンバーを入力した後に入室できます。
- 参加者の間で画面共有を行います。
- 各種のテキストメッセージとカスタムメッセージの送信をサポートします。

TUIRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実装プロセスについては、 多人数オーディオビデオルーム(Windows&Mac)をご参照ください。

- TRTC SDK： TRTC SDK を低遅延のビデオミーティングのコンポーネントとして使用します。
- IM SDK：IM SDK を利用してチャットルームの機能（**IM SDKはC++バージョンを使用**）を実装します。

TUIRoom API概要

TUIRoomCore基本関数

API	説明
GetInstance	シングルトンオブジェクトを取得します。
DestroyInstance	シングルトンオブジェクトを破棄します。
SetCallback	イベントコールバックを設定します。

ルーム関連インターフェース関数

API	説明
login	ログイン。
logout	ログアウト。
CreateRoom	ルームの作成（キャスターが呼び出し）。
DestroyRoom	ルームの破棄（キャスターが呼び出し）。
EnterRoom	入室（参加者が呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
LeaveRoom	退室（参加者またはキャスターが呼び出し）。
GetRoomInfo	ルーム情報の取得。
GetRoomUsers	ルーム内全メンバー情報の取得。
GetUserInfo	特定ユーザーの情報の取得。
TransferRoomMaster	キャスター権限の移転（キャスターが呼び出し）。

ローカルのオーディオビデオ操作インターフェース

API	説明
StartCameraPreview	ローカルビデオのプレビュー画面を立ち上げます。
StopCameraPreview	ローカルビデオキャプチャおよびプレビューを停止します。
UpdateCameraPreview	ローカルビデオレンダリングウィンドウを変更します。
StartLocalAudio	マイクキャプチャを起動します。
StopLocalAudio	マイクキャプチャを停止します。
StartSystemAudioLoopback	システム音声のキャプチャを起動/停止します。
StopSystemAudioLoopback	システム音声のキャプチャを起動/停止します。
SetVideoMirror	ローカル画面のミラーモードのプレビューを設定します。

リモートユーザーに関するインターフェース

API	説明
StartRemoteView	指定メンバーのリモートビデオ画面をサブスクリプションし再生します。
StopRemoteView	リモートビデオ画面のサブスクリプションをキャンセルし再生を停止します。
UpdateRemoteView	リモートユーザーのビデオレンダリングウィンドウを変更します。

チャットメッセージ送信インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
SendChatMessage	チャットメッセージを送信します。
SendCustomMessage	カスタムメッセージを送信します。

フィールドコントロール関連インターフェース

API	説明
MuteUserMicrophone	特定ユーザーのマイクを無効化/再有効化します。
MuteAllUsersMicrophone	全ユーザーのマイクを無効化/再有効化し、ステータスをルーム情報に同期させます。
MuteUserCamera	特定ユーザーのカメラを無効化/再有効化します。
MuteAllUsersCamera	全ユーザーのカメラを無効化/再有効化し、ステータスをルーム情報に同期させます。
MuteChatRoom	チャットルームのミュートを開始/停止します（キャスターが呼び出し）。
KickOffUser	ルーム内の特定ユーザーをリムーブします（キャスターが呼び出し）。
StartCallingRoll	キャスターが点呼を開始します。
StopCallingRoll	キャスターが点呼を終了します。
ReplyCallingRoll	メンバーがキャスターの点呼に応答します。
SendSpeechInvitation	キャスターがメンバーに発言するようインビテーションを送信します。
CancelSpeechInvitation	キャスターがメンバーの発言のためのインビテーションをキャンセルします。
ReplySpeechInvitation	メンバーがキャスターの発言申請に同意/を拒否します。
SendSpeechApplication	メンバーが発言を要請します。
CancelSpeechApplication	メンバーが発言申請をキャンセルします。
ReplySpeechApplication	キャスターがメンバーの発言申請に同意/拒否します。
ForbidSpeechApplication	キャスターが発言申請を禁止します。
SendOffSpeaker	キャスターがメンバーに発言を停止するよう命令します。
SendOffAllSpeakers	キャスターが全員に発言を停止するよう命令します。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
ExitSpeechState	メンバーは発言を停止し、視聴者になります。

基本コンポーネントインターフェース関数

API	説明
GetDeviceManager	ローカル設定管理オブジェクトITXDeviceManagerを取得します。
GetScreenShareManager	画面共有管理オブジェクトIScreenShareManagerを取得します。

クラウドレコーディングインターフェース関数

API	説明
StartCloudRecord	クラウドレコーディングを開始します 。
StopCloudRecord	クラウドレコーディングを停止します 。

美顔関連インターフェース関数

API	説明
SetBeautyStyle	美顔を設定します。

関連設定インターフェース

API	説明
SetVideoQosPreference	ネットワークトラフィックコントロール関連パラメータを設定します。

SDKバージョンインターフェース関数の取得

API	説明
GetSDKVersion	SDKバージョンを取得します。

TUIRoomCoreCallback API概要

エラーイベントコールバック

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
OnError	エラーのコールバック。

基本イベントコールバック

API	説明
OnLogin	ログインコールバック。
OnLogout	ログアウトコールバック。
OnCreateRoom	ルーム作成のコールバック。
OnDestroyRoom	ルーム解散のコールバック。
OnEnterRoom	入室のコールバック。
OnExitRoom	退室のコールバック。
OnFirstVideoFrame	最初のフレーム画面のコールバック。
OnUserVoiceVolume	音量の大きさのコールバック。
OnRoomMasterChanged	キャスター変更のコールバック。

リモートユーザーイベントコールバック

API	説明
OnRemoteUserEnter	リモートユーザー入室コールバック。
OnRemoteUserLeave	リモートユーザー退室コールバック。
OnRemoteUserCameraAvailable	リモートユーザーがカメラビデオを起動するかどうかのコールバック。
OnRemoteUserScreenAvailable	リモートユーザーが画面共有を開始するかどうかのコールバック。
OnRemoteUserAudioAvailable	リモートユーザーがマイクをオンにしているかどうかのコールバック。
OnRemoteUserEnterSpeechState	リモートユーザーの発言開始のコールバック。
OnRemoteUserExitSpeechState	リモートユーザーの発言終了のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

メッセージイベントのコールバック

API	説明
OnReceiveChatMessage	テキストメッセージ受信のコールバック。
OnReceiveCustomMessage	テキストメッセージ受信のコールバック。

フィールドコントロールイベントコールバック

API	説明
OnReceiveSpeechInvitation	ユーザーがキャスターの発言要請を受信した場合のコールバック。
OnReceiveInvitationCancelled	ユーザーがキャスターの発言要請キャンセルを受信する場合のコールバック。
OnReceiveReplyToSpeechInvitation	キャスターがユーザーの発言要請への同意を受信する場合のコールバック。
OnReceiveSpeechApplication	キャスターがユーザーの発言要請を受信する場合のコールバック。
OnSpeechApplicationCancelled	ユーザーが発言申請をキャンセルする場合のコールバック。
OnReceiveReplyToSpeechApplication	キャスターが発言要請に同意する場合のコールバック。
OnSpeechApplicationForbidden	キャスターが発言申請を禁止する場合のコールバック。
OnOrderedToExitSpeechState	メンバーが発言の停止をリクエストされる場合のコールバック。
OnCallingRollStarted	キャスターが点呼を開始し、メンバーが受信する場合のコールバック。
OnCallingRollStopped	キャスターが点呼を終了し、メンバーが受信する場合のコールバック。
OnMemberReplyCallingRoll	メンバーが点呼に応答し、キャスターが受信する場合のコールバック。
OnChatRoomMuted	キャスターがチャットルームのミュートを変更する場合のコールバック。
OnMicrophoneMuted	キャスターがマイクの無効化を設定する場合のコールバック。
OnCameraMuted	キャスターがカメラの無効化を設定する場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

統計および品質コールバック

API	説明
OnStatistics	技術指標統計のコールバック。
OnNetworkQuality	ネットワーク品質のコールバック。

画面共有関連コールバック

API	説明
OnScreenCaptureStarted	画面共有開始のコールバック。
OnScreenCaptureStopped	画面共有停止のコールバック。

ビデオレコーディングコールバック

API	説明
OnRecordError	レコーディングエラーのコールバック。
OnRecordComplete	レコーディング完了のコールバック。
OnRecordProgress	レコーディング進捗のコールバック。

ローカルデバイステストコールバック

API	説明
OnTestSpeakerVolume	スピーカー音量のコールバック。
OnTestMicrophoneVolume	マイク音量のコールバック。
OnAudioDeviceCaptureVolumeChanged	システムキャプチャ音量調節のコールバック。
OnAudioDevicePlayoutVolumeChanged	システム再生音量調節のコールバック。

TUIRoomCore基本関数

GetInstance

TUIRoomCore シングルトンオブジェクトを取得します。

©2013-2019 Tencent Cloud. All rights reserved.

```
static TUIRoomCore* GetInstance();
```

DestroyInstance

```
static void DestroyInstance();
```

SetCallback

TUIRoomCoreイベントコールバック。TUIRoomCoreCallbackを介してTUIRoomCoreの各種ステータス通知を取得できます。

```
virtual void SetCallback(const TUIRoomCoreCallback* callback) = 0;
```

Login

ログイン

```
virtual int Login(int sdk_appid, const std::string& user_id, const std::string& user_sig) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdk_appid	int	TRTCコンソール > **アプリケーション管理**> アプリケーション情報の中でSDKAppIDを確認できます。
user_id	string	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
user_sig	string	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、 UserSigの計算、使用方法をご参照ください。

Logout

ログアウト。

```
virtual int Logout() = 0;
```

CreateRoom

ルームの作成（キャスターが呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

```
virtual int CreateRoom(const std::string& room_id, TUISpeechMode speech_mode) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
room_id	string	ルームIDは、ご自身でアサインし、一元管理する必要があります。
speech_mode	TUISpeechMode	発言モード。

キャスターの通常の呼び出しフローは以下のとおりです。

1. **キャスター**が `CreateRoom()` を呼び出し、ルームを作成します。ルームの作成に成功したかどうかがOnCreateRoomを介してキャスターに通知されます。
2. **キャスター**が `EnterRoom()` を呼び出し、入室します。
3. **キャスター**が `StartCameraPreview()` を呼び出し 、カメラキャプチャとプレビューを起動します。
4. **キャスター**が `StartLocalAudio()` を呼び出し、ローカルマイクを起動します。

DestroyRoom

ルームの破棄（キャスターが呼び出し）。キャスターは、ルームの作成後、この関数を呼び出して、ルームを破棄できます。

```
virtual int DestroyRoom() = 0;
```

EnterRoom

入室（参加者が呼び出し）。

```
virtual int EnterRoom(const std::string& room_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
room_id	string	ルームID。

参加者が入室する場合の通常の呼び出し手順は次のとおりです。

1. **参加者**が `EnterRoom` を呼び出し、room_idを渡せば、入室できます。
2. **参加者**が `startCameraPreview()` を呼び出して、カメラプレビューを起動し、 `StartLocalAudio()` を呼び出して、マイクキャプチャを起動します。

©2013-2019 Tencent Cloud. All rights reserved.

3. **参加者**が `OnRemoteUserCameraAvailable` のイベントを受信し、 `StartRemoteView()` を呼び出して、ビデオ再生を開始します。

LeaveRoom

退室（参加者が呼び出し）。

```
virtual int LeaveRoom() = 0;
```

GetRoomInfo

ルーム情報を取得します。

```
virtual TUIRoomInfo GetRoomInfo() = 0;
```

GetRoomUsers

ルームの全メンバー情報を取得します。

```
virtual std::vector GetRoomUsers() = 0;
```

GetUserInfo

ルームのメンバー情報を取得します。

```
virtual const TUIUserInfo* GetUserInfo(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザーID。

SetSelfProfile

ユーザーの属性を設定します。

```
virtual int SetSelfProfile(const std::string& user_name, const std::string& avatar_url) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
user_name	string	ユーザー氏名。
avatar_url	string	ユーザーのプロフィール画像URL。

TransferRoomMaster

グループを他のユーザーに引き渡します。

```
virtual int TransferRoomMaster(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザーID。

ローカルプッシュインターフェース

StartCameraPreview

ローカルカメラプレビューを起動します。

```
virtual int StartCameraPreview(const liteav::TXView& view) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
view	liteav::TXView	ウィンドウハンドル。

StopCameraPreview

ローカルカメラプレビューを停止します。

```
virtual int StopCameraPreview() = 0;
```

UpdateCameraPreview

ローカルビデオプレビュー画面のウィンドウを更新します。

```
virtual int UpdateCameraPreview(const liteav::TXView& view) = 0;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
view	liteav::TXView	ウィンドウハンドル。

StartLocalAudio

ローカルオーディオデバイスを起動します。

```
virtual int StartLocalAudio(const liteav::TRTCAudioQuality& quality) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
view	liteav::TXView	ウィンドウハンドル。

StopLocalAudio

ローカルオーディオデバイスを停止します。

```
virtual int StopLocalAudio() = 0;
```

StartSystemAudioLoopback

システム音声のキャプチャを開始します。

```
virtual int StartSystemAudioLoopback() = 0;
```

StopSystemAudioLoopback

システム音声のキャプチャを停止します。

```
virtual int StopSystemAudioLoopback() = 0;
```

SetVideoMirror

イメージを設定します。

```
virtual int SetVideoMirror(bool mirror) = 0;
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
mirror	bool	ミラーオン/オフ。

リモートユーザーに関するインターフェース

StartRemoteView

リモートユーザーのビデオストリームのサブスクリプション。

```
virtual int StartRemoteView(const std::string& user_id, const liteav::TXView& view,
TUIStreamType type = TUIStreamType::kStreamTypeCamera) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	再生が必要なユーザー ID。
liteav::TXView	TXView	ビデオ画像をロードするviewウィジェット。
type	TUIStreamType	ストリームのタイプ。

StopRemoteView

サブスクリプションをキャンセルし、リモートビデオ画面の再生を停止します。

```
virtual int StopRemoteView(const std::string& user_id,
TUIStreamType type = TUIStreamType::kStreamTypeCamera) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	再生の停止が必要なユーザー ID。
type	TUIStreamType	ストリームのタイプ。

UpdateRemoteView

リモートビデオレンダリングウィンドウを更新します。

```
virtual int UpdateRemoteView(const std::string& user_id, TUIStreamType type, liteav::TXView& vie
w) = 0;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
type	TUIStreamType	ストリームのタイプ。
view	liteav::TXView	レンダリングウィンドウハンドル。

メッセージ送信インターフェース

SendChatMessage

テキストメッセージを送信します。

```
virtual int SendChatMessage(const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	string	メッセージの内容。

SendCustomMessage

カスタムメッセージを送信します。

```
virtual int SendCustomMessage(const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	string	メッセージの内容。

フィールドコントロール関連インターフェース

MuteUserMicrophone

特定ユーザーのマイクを無効化/再有効化します。

©2013-2019 Tencent Cloud. All rights reserved.

```
virtual int MuteUserMicrophone(const std::string& user_id, bool mute, Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
mute	bool	無効にするかどうか。
callback	Callback	インターフェースコールバック。

MuteAllUsersMicrophone

全ユーザーのマイクを無効化/再有効化します。

```
virtual int MuteAllUsersMicrophone(bool mute) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	bool	無効にするかどうか。

MuteUserCamera

特定ユーザーのカメラを無効化/再有効化します。

```
virtual int MuteUserCamera(const std::string& user_id, bool mute, Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
mute	bool	無効にするかどうか。
callback	Callback	インターフェースコールバック。

MuteAllUsersCamera

全ユーザーのカメラを無効化/再有効化します。

```
virtual int MuteAllUsersCamera(bool mute) = 0;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	bool	無効にするかどうか。

MuteChatRoom

チャットルームのミュートを開始/停止します。

```
virtual int MuteChatRoom(bool mute) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	bool	無効にするかどうか。

KickOffUser

キャスターがキックアウトします。

```
virtual int KickOffUser(const std::string& user_id, Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
callback	Callback	インターフェースコールバック。

StartCallingRoll

キャスターが点呼を開始します。

```
virtual int StartCallingRoll() = 0;
```

StopCallingRoll

キャスターが点呼を終了します。

```
virtual int StopCallingRoll() = 0;
```

©2013-2019 Tencent Cloud. All rights reserved.

ReplyCallingRoll

メンバーがキャスターの点呼に応答します。

```
virtual int ReplyCallingRoll(Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	Callback	インターフェースコールバック。

SendSpeechInvitation

キャスターがメンバーの発言を要請します。

```
virtual int SendSpeechInvitation(const std::string& user_id, Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
callback	Callback	インターフェースコールバック。

CancelSpeechInvitation

キャスターがメンバーへの発言要請をキャンセルします。

```
virtual int CancelSpeechInvitation(const std::string& user_id, Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
callback	Callback	インターフェースコールバック。

ReplySpeechInvitation

メンバーがキャスターの発言要請に同意/を拒否します。

```
virtual int ReplySpeechInvitation(bool agree, Callback callback) = 0;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
agree	bool	同意するかどうか。
callback	Callback	インターフェースコールバック。

SendSpeechApplication

メンバーが発言を要請します。

```
virtual int SendSpeechApplication(Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	Callback	インターフェースコールバック。

CancelSpeechApplication

メンバーが発言申請をキャンセルします。

```
virtual int CancelSpeechApplication(Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	Callback	インターフェースコールバック。

ReplySpeechApplication

キャスターがメンバーの発言申請に同意/を拒否します。

```
virtual int ReplySpeechApplication(const std::string& user_id, bool agree, Callback callback) = 0
;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	Callback	インターフェースコールバック。

ForbidSpeechApplication

キャスターが発言申請を禁止します。

```
virtual int ForbidSpeechApplication(bool forbid) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
forbid	bool	禁止するかどうか。

SendOffSpeaker

キャスターがメンバーに発言の停止を命令します。

```
virtual int SendOffSpeaker(const std::string& user_id, Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
callback	Callback	インターフェースコールバック。

SendOffAllSpeakers

キャスターが全メンバーに発言の停止を命令します。

```
virtual int SendOffAllSpeakers(Callback callback) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	Callback	インターフェースコールバック。

ExitSpeechState

メンバーが発言を停止し、視聴者になります。

©2013-2019 Tencent Cloud. All rights reserved.

```
virtual int ExitSpeechState() = 0;
```

基本コンポーネントインターフェース

GetDeviceManager

デバイス管理のオブジェクトポインタを取得します。

```
virtual liteav::ITXDeviceManager* GetDeviceManager() = 0;
```

GetScreenShareManager

画面共有管理のオブジェクトポインタを取得します。

```
virtual IScreenShareManager* GetScreenShareManager() = 0;
```

クラウドレコーディングインターフェース

StartCloudRecord

クラウドレコーディングを開始します。

```
virtual int StartCloudRecord() = 0;
```

StopCloudRecord

クラウドレコーディングを停止します。

```
virtual int StopCloudRecord() = 0;
```

美顔関連インターフェース関数

SetBeautyStyle

美顔、美白、肌の色調補正効果のランクを設定します。

©2013-2019 Tencent Cloud. All rights reserved.

```
virtual int SetBeautyStyle(liteav::TRTCBeautyStyle style, uint32_t beauty_level,
uint32_t whiteness_level, uint32_t ruddiness_level) = 0;
```

美顔管理では、次の機能を使用できます。

- 「美顔スタイル」を「スムース」または「ナチュラル」に設定します。「スムース」では、より強力な美肌補正効果が得られます。
- 「美顔レベル」を設定します。数値の範囲は0～9で、0はオフ、1～9までは数値が大きくなるほど効果が高くなります。
- 「美白レベル」を設定します。数値の範囲は0～9で、0はオフ、1～9までは数値が大きくなるほど効果が高くなります。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
style	liteav::TRTCBeautyStyle	美顔スタイル。
beauty_level	uint32_t	美顔レベル。
whiteness_level	uint32_t	美白レベル。
ruddiness_level	uint32_t	肌色補正レベル。

関連設定インターフェース

SetVideoQosPreference

ネットワークトラフィックコントロール関連パラメータを設定します。

```
virtual int SetVideoQosPreference(TUIVideoQosPreference preference) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
preference	TUIVideoQosPreference	ネットワークトラフィックコントロールポリシー。

SDKバージョンインターフェースの取得

GetSDKVersion

©2013-2019 Tencent Cloud. All rights reserved.

SDKバージョン情報を取得します。

```
virtual const char* GetSDKVersion() = 0;
```

エラーイベントコールバック

OnError

```
void OnError(int code, const std::string& message);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	string	エラー情報。

基本イベントコールバック

OnLogin

```
virtual void OnLogin(int code, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	string	ログイン情報またはログイン失敗のエラー情報。

OnLogout

```
virtual void OnLogout(int code, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
message	string	エラー情報。

OnCreateRoom

ルーム作成のコールバック。

```
virtual void OnCreateRoom(int code, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	string	エラー情報。

OnDestroyRoom

ルーム解散のコールバック。

```
virtual void OnDestroyRoom(int code, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	string	エラー情報。

OnEnterRoom

入室コールバック。

```
virtual void OnEnterRoom(int code, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	string	エラー情報。

©2013-2019 Tencent Cloud. All rights reserved.

OnExitRoom

退室コールバック。

```
virtual void OnExitRoom(TUIExitRoomType type, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
type	TUIExitRoomType	退室のタイプ。
message	string	エラー情報。

OnFirstVideoFrame

自身のローカルまたはリモートユーザーの最初のフレーム画面のレンダリングを開始します。

```
virtual void OnFirstVideoFrame(const std::string& user_id, const TUIStreamType stream_type) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
stream_type	TUIStreamType	ストリームのタイプ。

OnUserVoiceVolume

ユーザー音量の大きさのコールバック。

```
virtual void OnUserVoiceVolume(const std::string& user_id, int volume)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
volume	int	ユーザーの音量の大きさ、値の範囲0～100。

OnRoomMasterChanged

キャスター変更のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

```
virtual void OnRoomMasterChanged(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

リモートユーザーコールバックイベント

OnRemoteUserEnter

リモートユーザー入室コールバック。

```
virtual void OnRemoteUserEnter(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

OnRemoteUserLeave

リモートユーザー退室コールバック。

```
virtual void OnRemoteUserLeave(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

OnRemoteUserCameraAvailable

リモートユーザーが、カメラ、ビデオを起動しているかどうか。

```
virtual void OnRemoteUserCameraAvailable(const std::string& user_id, bool available) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
available	bool	true：ビデオストリームデータあり；false：ビデオストリームデータなし。

OnRemoteUserScreenAvailable

リモートユーザーが画面共有を開始しているかどうか。

```
virtual void OnRemoteUserScreenAvailable(const std::string& user_id, bool available) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
available	bool	true：ビデオストリームデータあり；false：ビデオストリームデータなし。

OnRemoteUserAudioAvailable

リモートユーザーがマイクをオンにしているかどうか。

```
virtual void OnRemoteUserAudioAvailable(const std::string& user_id, bool available) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
available	bool	true：オーディオストリームデータあり、false：オーディオストリームデータなし。

OnRemoteUserEnterSpeechState

リモートユーザーが発言を開始します。

```
virtual void OnRemoteUserEnterSpeechState(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

OnRemoteUserExitSpeechState

リモートユーザーが発言を終了します。

```
virtual void OnRemoteUserExitSpeechState(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

チャットルームメッセージイベントコールバック

OnReceiveChatMessage

テキストメッセージの受信。

```
virtual void OnReceiveChatMessage(const std::string& user_id, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
message	string	テキストメッセージ。

OnReceiveCustomMessage

カスタムメッセージの受信。

```
virtual void OnReceiveCustomMessage(const std::string& user_id, const std::string& message) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
message	string	カスタムメッセージ。

フィールドコントロールメッセージコールバック

OnReceiveSpeechInvitation

ユーザーがキャスターの発言要請を受信する場合のコールバック。

```
virtual void OnReceiveSpeechInvitation() = 0;
```

OnReceiveInvitationCancelled

ユーザーがキャスターの発言要請キャンセルを受信する場合のコールバック。

```
virtual void OnReceiveInvitationCancelled() = 0;
```

OnReceiveReplyToSpeechInvitation

キャスターがユーザーの発言要請への同意を受信する場合のコールバック。

```
virtual void OnReceiveReplyToSpeechInvitation(const std::string& user_id, bool agree) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。
agree	bool	同意するかどうか。

OnReceiveSpeechApplication

キャスターがユーザーの発言要請を受信する場合のコールバック。

```
virtual void OnReceiveSpeechApplication(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

OnSpeechApplicationCancelled

ユーザーが発言申請をキャンセルする場合のコールバック。

```
virtual void OnSpeechApplicationCancelled(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

OnReceiveReplyToSpeechApplication

キャスターが発言申請に同意する場合のコールバック。

```
virtual void OnReceiveReplyToSpeechApplication(bool agree) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
agree	bool	同意するかどうか。

OnSpeechApplicationForbidden

キャスターが発言申請を禁止する場合のコールバック。

```
virtual void OnSpeechApplicationForbidden(bool forbidden) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
forbidden	bool	禁止するかどうか。

OnOrderedToExitSpeechState

メンバーが発言を停止するようリクエストされる場合のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

```
virtual void OnOrderedToExitSpeechState() = 0;
```

OnCallingRollStarted

キャスターが点呼を開始し、メンバーが受信する場合のコールバック。

```
virtual void OnCallingRollStarted() = 0;
```

OnCallingRollStopped

キャスターが点呼を終了し、メンバーが受信する場合のコールバック。

```
virtual void OnCallingRollStopped() = 0;
```

OnMemberReplyCallingRoll

メンバーが点呼に応答し、キャスターが受信する場合のコールバック。

```
virtual void OnMemberReplyCallingRoll(const std::string& user_id) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user_id	string	ユーザー ID。

OnChatRoomMuted

キャスターがチャットルームのミュートを変更する場合のコールバック。

```
virtual void OnChatRoomMuted(bool muted) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	bool	無効にするかどうか。

OnMicrophoneMuted

キャスターがマイクの無効化を設定する場合のコールバック。

```
virtual void OnMicrophoneMuted(bool muted) = 0;
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	bool	無効にするかどうか。

OnCameraMuted

キャスターがカメラの無効化を設定する場合のコールバック。

```
virtual void OnCameraMuted(bool muted) = 0;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
muted	bool	無効にするかどうか。

統計および品質コールバック

OnStatistics

技術指標統計のコールバック。

```
virtual void OnStatistics(const liteav::TRTCStatistics& statis) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
statis	liteav::TRTCStatistics	統計データ。

OnNetworkQuality

ネットワーク状況のコールバック。

```
virtual void OnNetworkQuality(const liteav::TRTCQualityInfo& local_quality, liteav::TRTCQualityIn
fo* remote_quality,
uint32_t remote_quality_count) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
local_quality	liteav::TRTCQualityInfo	ローカルユーザー品質情報。
remote_quality	liteav::TRTCQualityInfo*	リモートユーザー品質情報ポインタ。
remote_quality_count	uint32_t	リモートユーザー数。

スクリーンキャプチャのイベントコールバック

OnScreenCaptureStarted

画面共有開始のコールバック。

```
virtual void OnScreenCaptureStarted() {}
```

OnScreenCaptureStopped

画面共有停止のコールバック。

```
void OnScreenCaptureStopped(int reason) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
reason	int	停止の理由。0：ユーザーの自発的な停止。1：その他アプリケーションに占有されたことによる停止。

ビデオレコーディングコールバック

OnRecordError

レコーディングエラーのコールバック。

```
virtual void OnRecordError(TXLiteAVLocalRecordError error, const std::string& messgae) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
error	TXLiteAVLocalRecordError	エラー情報。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
messgae	string	エラー説明。

OnRecordComplete

レコーディング完了のコールバック。

```
virtual void OnRecordComplete(const std::string& path) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
path	string	エラーの説明。

OnRecordProgress

レコーディング進捗のコールバック。

```
virtual void OnRecordProgress(int duration, int file_size) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
duration	int	ファイルの長さ。
file_size	int	ファイルのサイズ。

ローカルデバイステストコールバック

OnTestSpeakerVolume

スピーカー音量の大きさのコールバック。

```
virtual void OnTestSpeakerVolume(uint32_t volume) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	uint32_t	音量の大きさ。

©2013-2019 Tencent Cloud. All rights reserved.

OnTestMicrophoneVolume

マイク音量の大きさのコールバック。

```
virtual void OnTestMicrophoneVolume(uint32_t volume) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	uint32_t	音量の大きさ。

OnAudioDeviceCaptureVolumeChanged

システムキャプチャ音量調節のコールバック。

```
virtual void OnAudioDeviceCaptureVolumeChanged(uint32_t volume, bool muted) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	uint32_t	音量の大きさ。
muted	bool	無効にされるかどうか

OnAudioDevicePlayoutVolumeChanged

システム再生音量調節のコールバック。

```
virtual void OnAudioDevicePlayoutVolumeChanged(uint32_t volume, bool muted) {}
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	uint32_t	音量の大きさ。
muted	bool	無効にされるかどうか

©2013-2019 Tencent Cloud. All rights reserved.

ボイスサロン
ボイスサロン(Android)

最終更新日：：2022-03-09 17:55:35

デモンストレーション

ダウンロードしてAppをインストールすると、ボイスサロンの機能をご体験いただけます。これには、ボイスチャット、マイクのオン・オフ、低遅延音声インタラクションなどのボイスチャットシナリオにおけるTRTCの関連機能が含まれています。

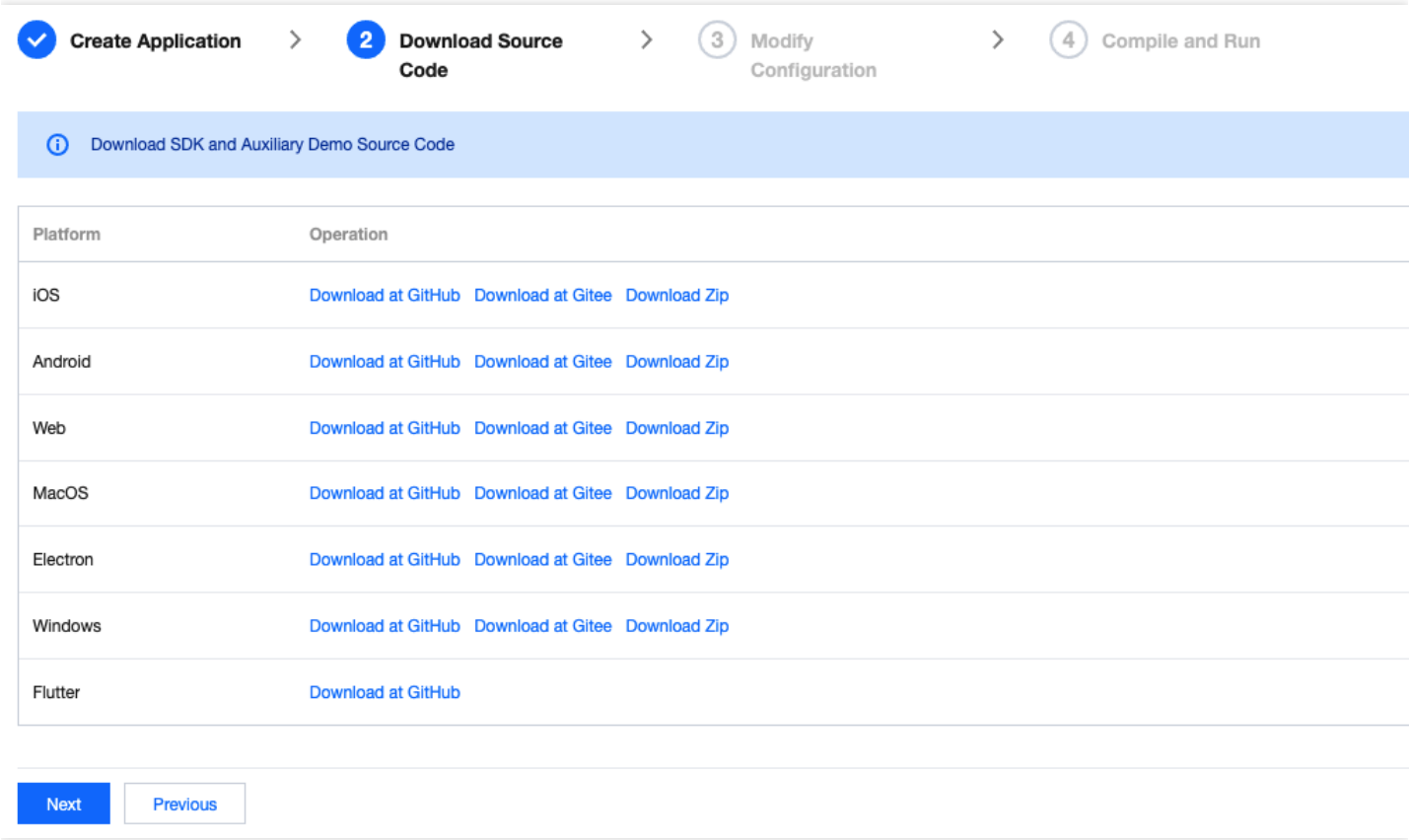
ボイスサロン機能をすばやくアクセスする必要がある場合、当社が提供するAppをもとに直接変更を加えて適応させるか、または当社が提供するTRTCChatSalonコンポーネントでカスタマイズしたUIを実装することができます。

AppのUIの再利用

手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、 **開発支援** > **Demoクイックスタート**を選択します。
2. アプリケーション名（例： `TestChatSalon` ）を入力し、**作成**をクリックします。
3. **ダウンロードしました。次のステップ**をクリックすると、この手順をスキップします。

©2013-2019 Tencent Cloud. All rights reserved.



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUIChatSalonに進み、ソースコードをCloneまたはダウンロードします。

手順3：Appプロジェクトファイルの設定

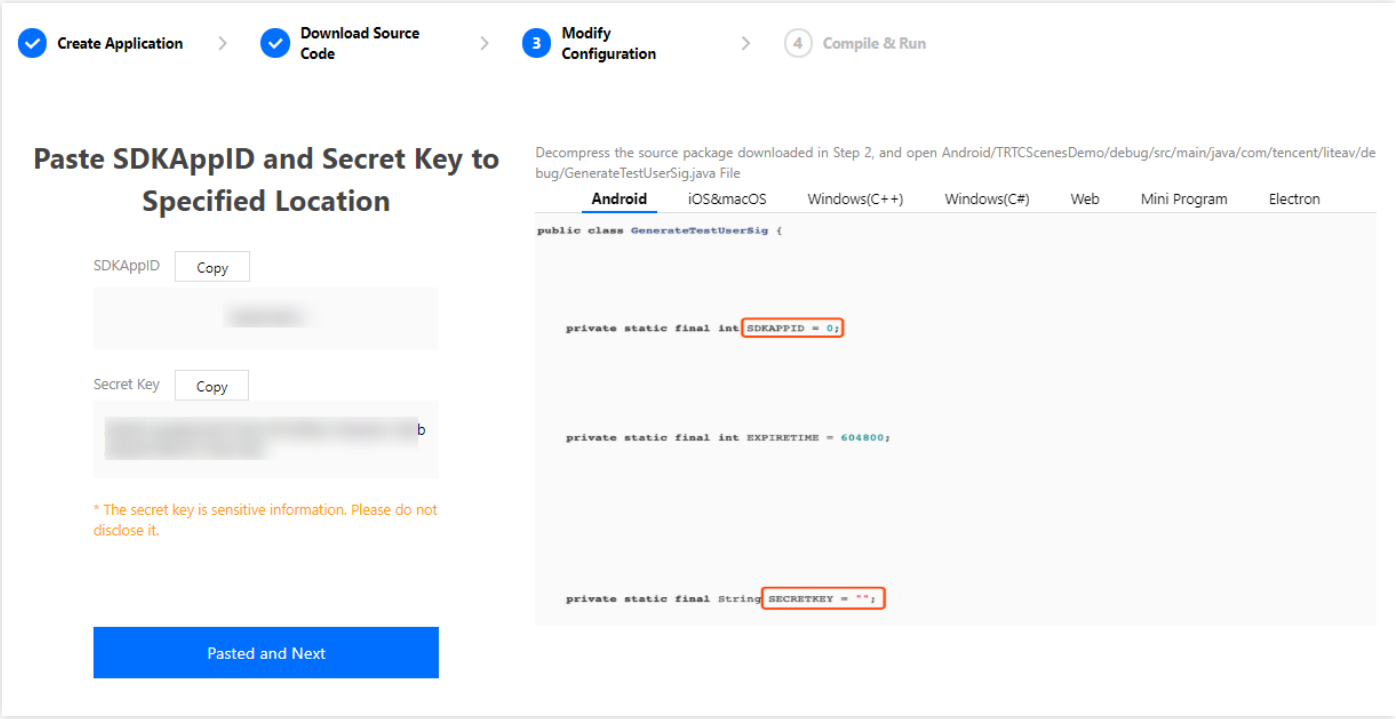
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `Android/Debug/src/main/java/com/tencent/liteav/debug/GenerateTestUserSig.java` のファイルを見つけて開きます。

3. `GenerateTestUserSig.java` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはプレースホルダー(PLACEHOLDER)。実際のSDKAppIDを設定してください。

©2013-2019 Tencent Cloud. All rights reserved.

- SECRETKEY：デフォルトはプレースホルダー(PLACEHOLDER)。実際のキー情報を設定してください。



4. 貼り付け完了後、**貼り付けました。次のステップ**をクリックすれば、作成が完了します。

5. コンパイル完了後、 **コンソール概要に戻る** をクリックすれば終了です。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Android Studio（バージョン3.5以降）を使用してソースコード `TUIChatSalon` プロジェクトを開き、**実行**をクリックすれば、このAppのデバッグが開始されます。

手順5：Appソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードのSourceフォルダは、uiとmodelという2つのサブフォルダを含んでいます。uiフォルダにはインターフェースコードが含まれています。以下の表にファイルまたはフォルダおよび対応するUIをリストアップして、二次調整を行いやすくしています。

ファイルまたはフォルダ	機能の説明
base	UIに使用される基礎となるクラス。
list	ルームページの作成。
room	メインルームページは、管理者とリスナーという2種類のインターフェースがあります。
widget	汎用ウィジェット。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ログイン後、**ルームの作成**をクリックします。
3. ルームトピックを入力し、**チャット開始**をクリックします。

ユーザーB

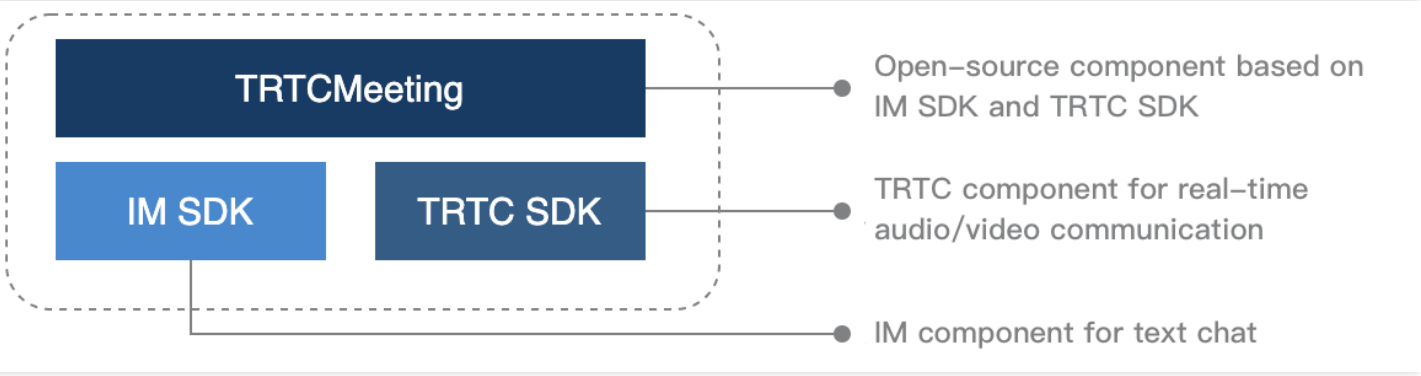
1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ユーザーAが作成したルームナンバーを入力し、**入室する**をクリックします。

> 注意：
> ルームナンバーはユーザーAのルーム上部に表示されます。

UIカスタマイズの実装

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードのSourceフォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTRTCChatSalonがあります。`TRTCChatSalon.java` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタムUIを実装することができます。



手順1：SDKの統合

ボイスサロンコンポーネントSourceは、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

方法1：Mavenリポジトリを介する依存

1. dependenciesにTRTCSDKとIMSDKの依存を追加します。

```
dependencies {
complie ”com.tencent.liteav:LiteAVSDK_TRTC:latest.release”
complie ’com.tencent.imsdk:imsdk:latest.release’
compile ’com.google.code.gson:gson:2.3.1’
}
```

2. defaultConfigでAppが使用するCPUアーキテクチャを指定します。

```
defaultConfig {
ndk {
abiFilters ”armeabi-v7a”
}
}
```

3. **Sync Now**をクリックし、自動でSDKをダウンロードし、プロジェクトに統合します。

方法2：ローカルAARを介する依存

©2013-2019 Tencent Cloud. All rights reserved.

開発環境でのMavenリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定および難読化ルール

AndroidManifest.xmlにAppの権限を設定します。SDKには次の権限が必要です（6.0以降のAndroidシステムには、動的な読み取りストレージの権限が必要です）。

```
<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.READ_PHONE_STATE">
```

proguard-rules.proファイルでは、SDK関連を非難読化リストに追加します。

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

手順3：TRTCChatSalonコンポーネントのインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
Source/src/main/java/com/tencent/liteav/trtcchatsalon/model
```

手順4：コンポーネントの作成およびログイン

1. `sharedInstance` インターフェースを呼び出すと、TRTCChatSalonコンポーネントのインスタンスオブジェクトを作成できます。
2. `setDelegate` 関数を呼び出してコンポーネントのイベント通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	作用

©2013-2019 Tencent Cloud. All rights reserved.

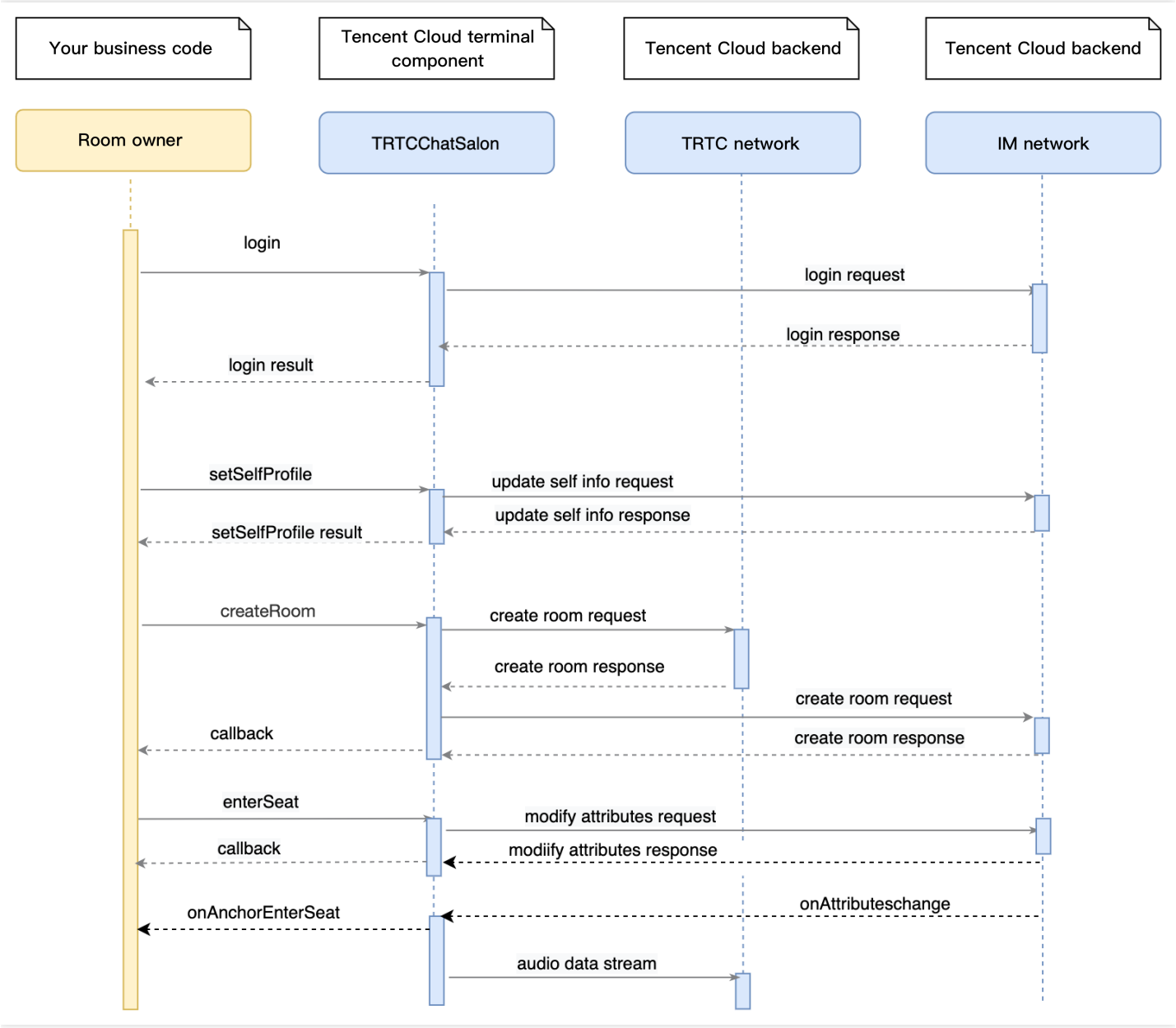
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
callback	ログインのコールバック。成功時にcodeは0になります。

```
TRTCChatSalon mTRTCChatSalon = TRTCChatSalon.sharedInstance(this);
mTRTCChatSalon.setDelegate(this);
mTRTCChatSalon.login(SDKAPPID, userId, userSig, new TRTCChatSalonCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//ログイン成功
}
}
});
```

手順5：管理者側での配信開始

1. 管理者は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 管理者は、 `createRoom` を呼び出して新しいボイスサロンを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否、ルームタイプなどルームの属性情報を渡します。
3. 管理者は、メンバーが参加した `onAnchorEnterSeat` というイベント通知を受信します。この時、マイク集音は自動的に開始されます。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.管理者は、ニックネームおよびプロフィール画像を設定します
mTRTCChatSalon.setSelfProfile("my_name", "my_face_url", null);

// 2.管理者は、createRoomを呼び出してルームを作成します
final TRTCChatSalonDef.RoomParam roomParam = new TRTCChatSalonDef.RoomParam();
roomParam.roomName = "ルーム名";
roomParam.needRequest = true; // マイク・オンの管理者による確認の要否
roomParam.coverUrl = "ルームカバー図のURL ";
mTRTCChatSalon.createRoom(mRoomId, roomParam, new TRTCChatSalonCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
// 3.座席を占有します
```

©2013-2019 Tencent Cloud. All rights reserved.

```
mTRTCChatSalon.enterSeat(new TRTCChatSalonCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
}
}
});
}
}
});

// 4.席の占有成功後、onAnchorEnterSeatイベント通知を受信します
@Override
public void onAnchorEnterSeat(TRTCChatSalonDef.UserInfo userInfo) {
}
```

手順6：リスナー側の視聴

1. リスナー側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. リスナー側は、業務バックエンドから最新のボイスサロンのルームリストを取得します。

> 説明：
> Appのボイスサロンリストは、デモンストレーション用のためだけのものです。ボイスサロンリストのビジネスロジックは様々です。Tencent Cloudでは現在ボイスサロンリストの管理サービスを提供していません。ご自身でボイスサロンリストを管理してください。

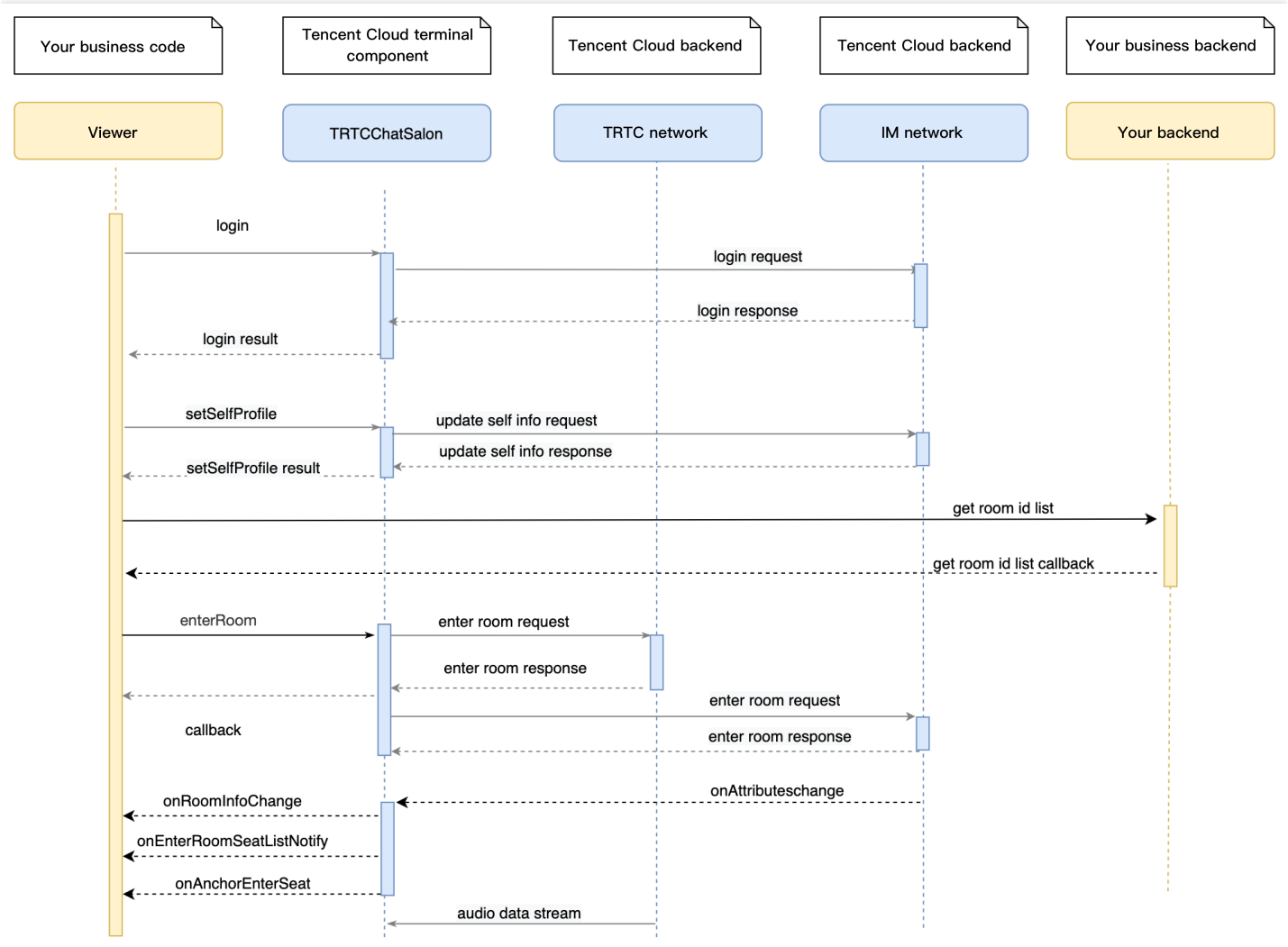
3. リスナー側は `getRoomInfoList` を呼び出して、ルームの詳細情報を取得します。この情報は、管理者側が `createRoom` を呼び出してボイスサロンを作成するときに設定する簡単な説明情報です。

> 注意：
> ボイスサロンリストに十分に包括的な情報がある場合は、 `getRoomInfoList` の呼び出しに関する手順をスキップできます。

4. リスナーはボイスサロンの1つを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
5. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。

©2013-2019 Tencent Cloud. All rights reserved.

6. 入室後にマイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。




```
// 1. リスナーは、ニックネームおよびプロフィール画像を設定します
mTRTCChatSalon.setSelfProfile("my_name", "my_face_url", null);

// 2.業務バックエンドから取得したルームリストをroomListと仮定します
List<integer> roomList = GetRoomList();

// 3. getRoomInfoListを呼び出すことによって、ルームの詳細情報を取得します
mTRTCChatSalon.getRoomInfoList(roomList, new TRTCChatSalonCallback.RoomInfoCallback() {

@Override
public void onCallback(int code, String msg, List<trtcchatsalondef.roominfo> list) {
if (code == 0) {
// この時、自分のUIルームリストを更新することができます
}
}
```

©2013-2019 Tencent Cloud. All rights reserved.

```
});

// 4.roomIdを渡してルームに参加します
mTRTCChatSalon.enterRoom(roomId, new TRTCChatSalonCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//入室に成功
}
}
});

// 5.入室に成功後、onRoomInfoChangeイベント通知を受信します
@Override
public void onRoomInfoChange(TRTCChatSalonDef.RoomInfo roomInfo) {
mNeedRequest = roomInfo.needRequest;
mRoomName = roomInfo.roomName;
// UIはタイトルなどを表示することができます
}

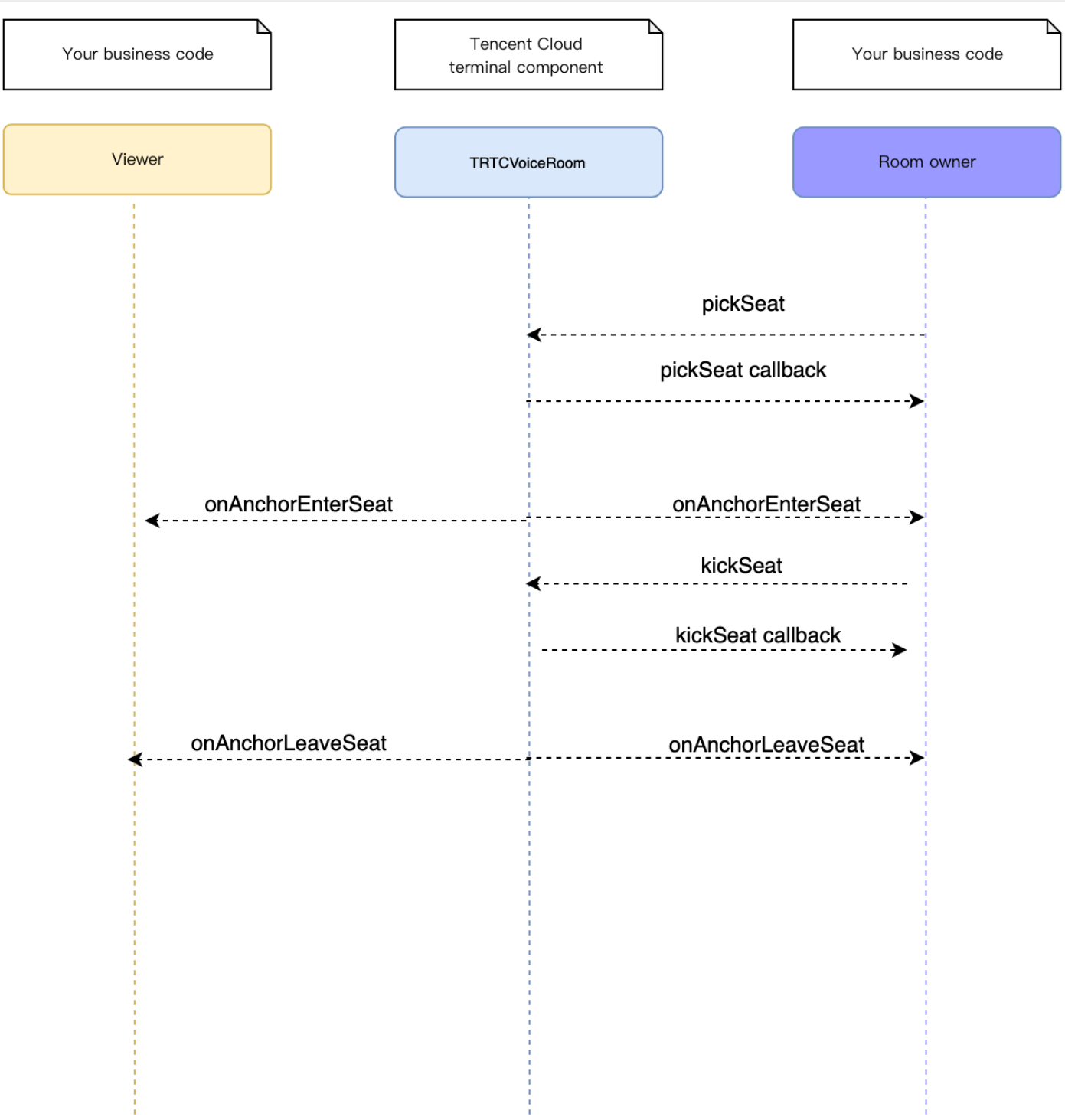
// 6. onAnchorEnterSeatイベント通知を受信します
@Override
public void onAnchorEnterSeat(TRTCChatSalonDef.UserInfo userInfo) {
}
```

手順7：マイクのオン・オフ

- 管理者側
- リスナー側

1. `pickSeat` は、リスナーのuserIdを渡し、発言できるように視聴者を招待できます。ルーム内の全メンバーは `onAnchorEnterSeat` というイベント通知を受信します。
2. `kickSeat` で該当するユーザーのuserIdを渡すと、その人をキックアウトしてマイク・オフにできます。ルーム内の全メンバーが `onAnchorLeaveSeat` のイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.



マイク操作後のイベント通知の順番は次のとおりです。callback > onAnchorEnterSeatなど独立したイベント。

```
// 1.管理者が、視聴者を発言できるように招待できます
mTRTCChatSalon.pickSeat("123", new TRTCChatSalonCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// 2. callbackコールバックを受信します
if (code == 0) {
}
}
});

// 3.リスナーがキャスターとしてマイク通知に進むと、リスナーが実際にマイク・オンに成功したかどうかを判断できます
public void onAnchorEnterSeat(TRTCChatSalonDef.UserInfo user) {
}
```

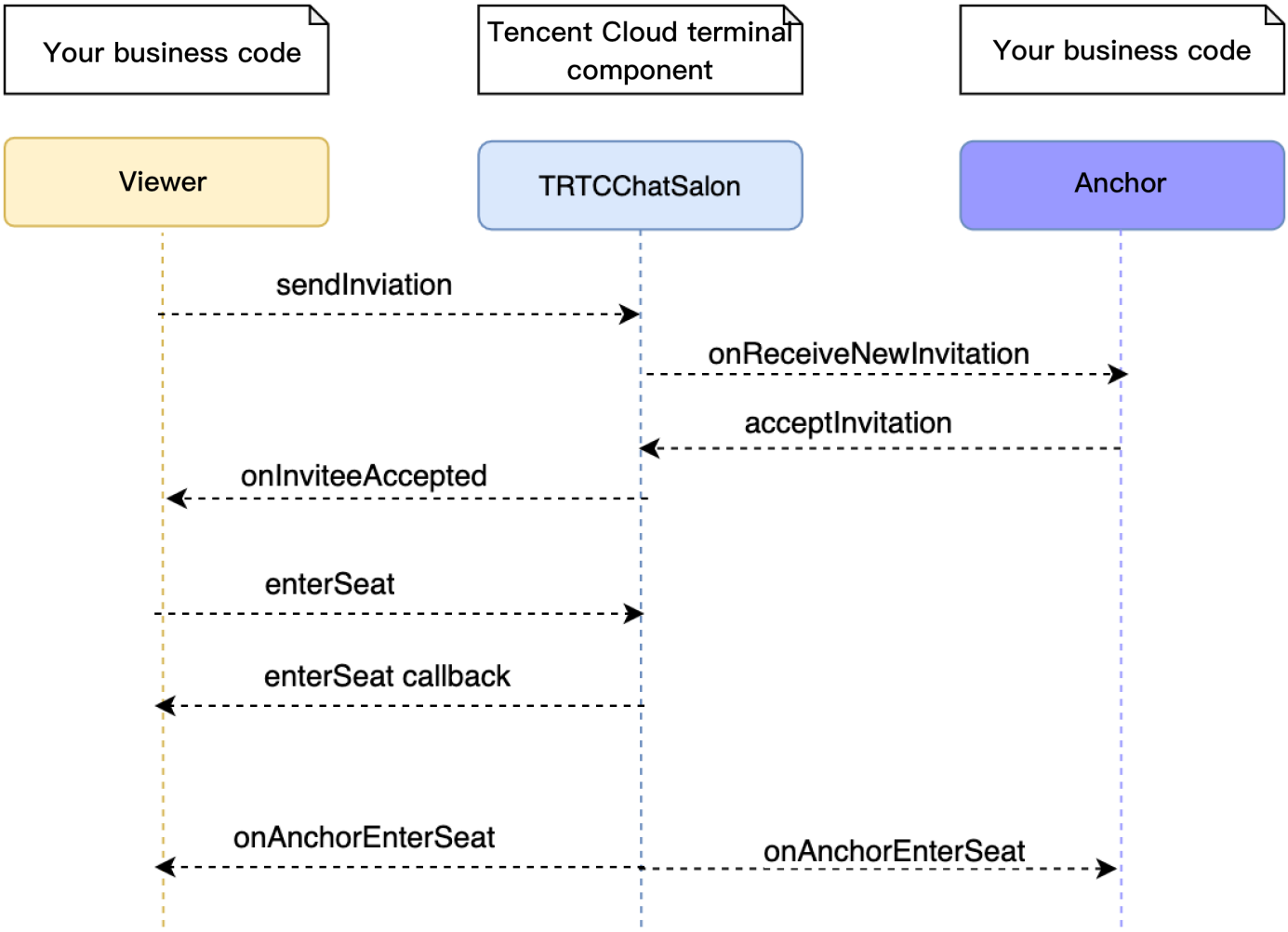
手順8：招待シグナリングの使用

Appにおいて相手の同意がなければ、次の操作の業務フローを実施できない場合は、招待シグナリングによって相応のサポートを行うことができます。

- リスナーからの自主的なマイク・オン申請
- 管理者からの招待によるリスナーのマイク・オン

1. リスナー側が `sendInvitation` を呼び出し、管理者のuserIdおよび業務のカスタムコマンドワードなどを渡します。この時、関数が1つのinviteIdを返しますので、このinviteIdを記録します。
2. 管理者側は、 `onReceiveNewInvitation` というイベント通知を受信します。この時UIでウィンドウをポップアップさせ、管理者に同意の有無をたずねることができます。
3. 管理者が同意を選択後、 `acceptInvitation` を呼び出してinviteIdを渡します。
4. リスナー側は、 `onInviteeAccepted` というイベント通知を受信し、 `enterSeat` を呼び出してマイク・オンにします。

©2013-2019 Tencent Cloud. All rights reserved.




```
// リスナー側の視点
// 1.sendInvitationを呼び出し、マイク・オンをリクエストします
String inviteId = mTRTCChatSalon.sendInvitation("ENTER_SEAT", ownerUserId, "123", null);

// 2.招待のリクエスト同意を受信し、正式にマイク・オンになります
@Override
public void onInviteeAccepted(String id, String invitee) {
if(id.equals(inviteId)) {
mTRTCChatSalon.enterSeat(null);
}
}

// 管理者側の視点
// 1.管理者がリクエストを受信します
@Override
public void onReceiveNewInvitation(final String id, String inviter, String cmd, final String cont
ent) {
if (cmd.equals("ENTER_SEAT")) {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// 2.管理者がリスナーのリクエストに同意します
mTRTCChatSalon.acceptInvitation(id, null);
}
}
```

手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、このルーム内のすべてのキャスターおよびリスナーが `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

```
// 発信側：テキストメッセージの発信
mTRTCChatSalon.sendRoomTextMsg("Hello Word!", null);
// 受信側：テキストメッセージのモニタリング
mTRTCChatSalon.setDelegate(new TRTCChatSalonDelegate() {
@Override
public void onRecvRoomTextMsg(String message, TRTCChatSalonDef.UserInfo userInfo) {
Log.d(TAG,"が" + userInfo.userName + "から受信したメッセージ:" + message);
}
});
```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターとリスナーが `onRecvRoomCustomMsg` コールバックを受信できます。
  カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

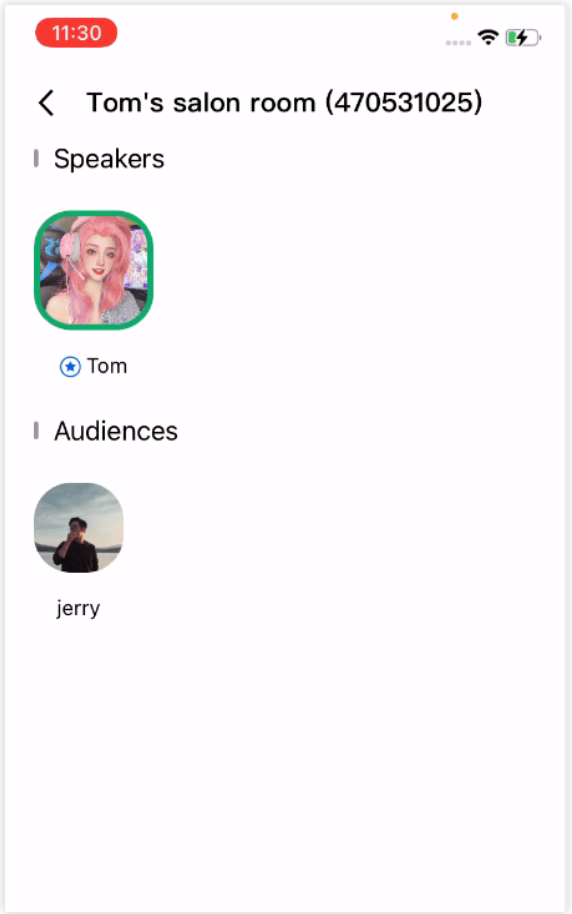
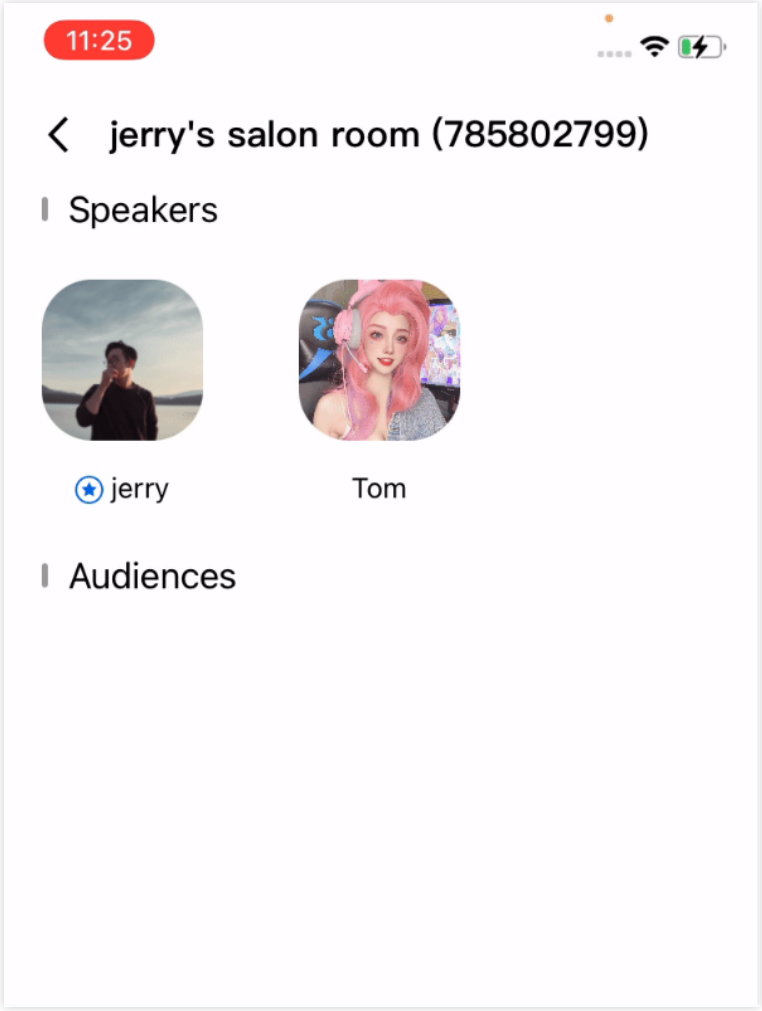
```
// 発信側：カスタマイズCmdによって弾幕と「いいね」情報を区別することができます
// eg:「CMD_DANMU」は弾幕コメントを表し、「CMD_LIKE」は「いいね」情報を表します
mTRTCChatSalon.sendRoomCustomMsg("CMD_DANMU", "Hello world", null);
mTRTCChatSalon.sendRoomCustomMsg("CMD_LIKE", "", null);
// 受信側：カスタムメッセージのモニタリング
mTRTCChatSalon.setDelegate(new TRTCChatSalonDelegate() {
@Override
public void onRecvRoomCustomMsg(String cmd, String message, TRTCChatSalonDef.UserInfo userInf
o) {
if ("CMD_DANMU".equals(cmd)) {
// 弾幕コメントの受信
Log.d(TAG, "が" + userInfo.userName + "から受信した弾幕コメント:" + message);
} else if ("CMD_LIKE".equals(cmd)) {
// 「いいね」情報の受信
Log.d(TAG, userInfo.userName + "いいねを付けました！");
}
}
});
```

©2013-2019 Tencent Cloud. All rights reserved.

ボイスサロン(Flutter)

最終更新日：：2022-02-18 14:16:06

デモンストレーション

管理者によるマイク操作	リスナーによるマイク操作
	

ボイスサロン機能をすばやく実装する必要がある場合、当社が提供するDemoをもとにアダプタを直接修正するか、または当社が提供するTRTCChatSalonコンポーネントでカスタマイズしたUIを実装することができます。

DemoのUIの再利用

©2013-2019 Tencent Cloud. All rights reserved.

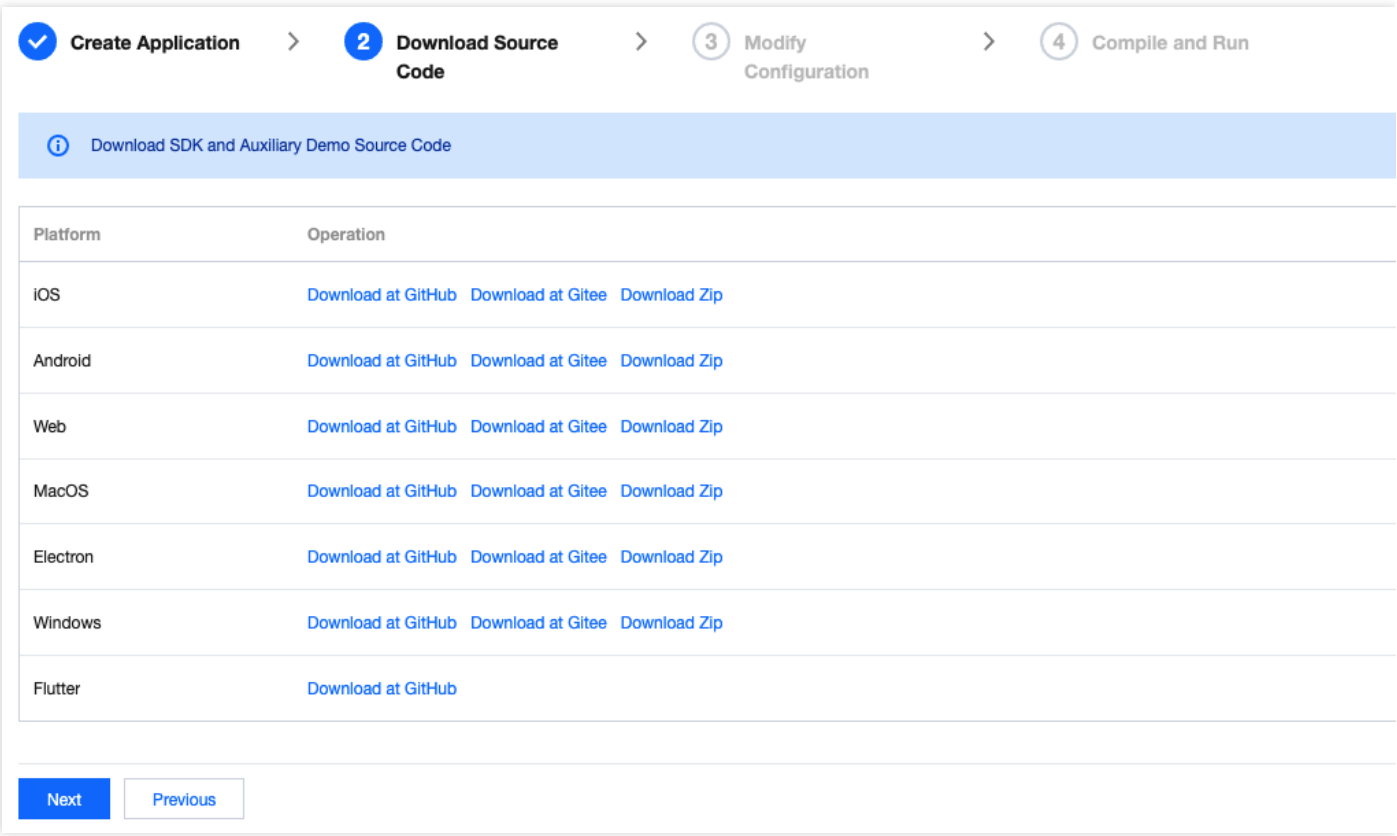
手順1：アプリケーションの新規作成

1. TRTCコンソールにログインし、【開発支援】>【快速跑通Demo】を選択します。
2. アプリケーション名（例：`TestChatSalon`）を入力し、【作成】をクリックします。

> 注意：
> 本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

ステップ2：SDKおよびDemoソースコードをダウンロード

1. 実際のビジネスニーズに応じて、SDKと付属のDemoソースコードをダウンロードします。
2. ダウンロード完了後、【ダウンロード済み。次へ】をクリックします。



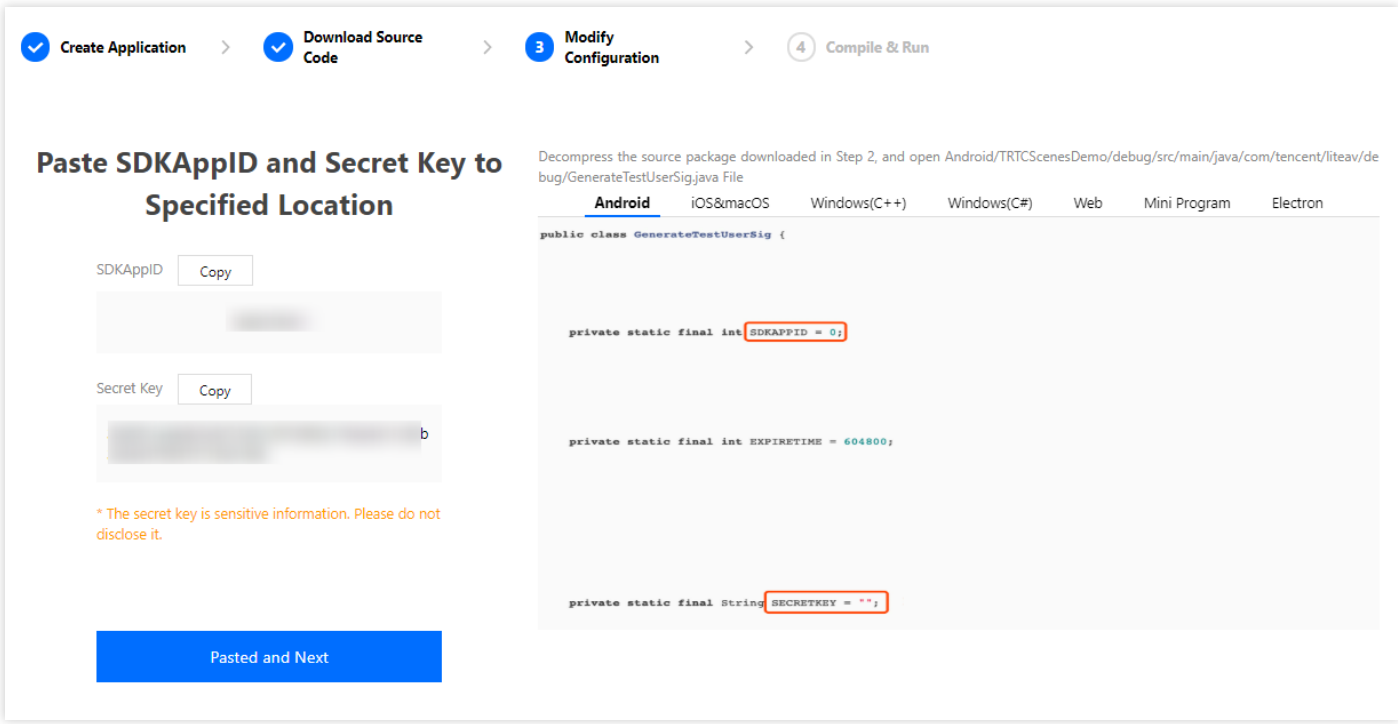
ステップ3：Demo プログラムファイルの設定

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `/example/lib/debug/GenerateTestUserSig.dart` ファイルを見つけて開きます。

©2013-2019 Tencent Cloud. All rights reserved.

3. `GenerateTestUserSig.dart` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトはPLACEHOLDER、実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトはPLACEHOLDER、実際のキー情報を設定してください。



4. 貼り付け完了後、【貼り付けました。次へ】をクリックすれば、作成が完了します。

5. コンパイル完了後、【コンソールの概要ページに戻る】をクリックすればOKです。

注意：

- ここで言及したUserSigの新規作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：コンパイル実行

©2013-2019 Tencent Cloud. All rights reserved.

注意：
Androidは実際のマシンで実行する必要があり、エミュレーターのデバッグはサポートされていません。

1. `flutter pub get` を実行します。
2. コンパイルを実行し、デバッグを行います。
   - Android 端末
   - iOS 端末
   i. `flutter run` を実行します。
   ii. Android Studio（3.5およびそれ以降のバージョン）を使用して、ソースプロジェクトを開き、【実行】をクリックすれば完了です。

手順5：Demoソースコードの修正

ソースコード のtrtcchatsalondemoフォルダには2つのサブフォルダuiとmodelが含まれ、uiフォルダにはインターフェースコードが含まれています。下表に各ファイルまたはフォルダおよび対応するUIをリストアップし、二次調整に役立つようにしています。

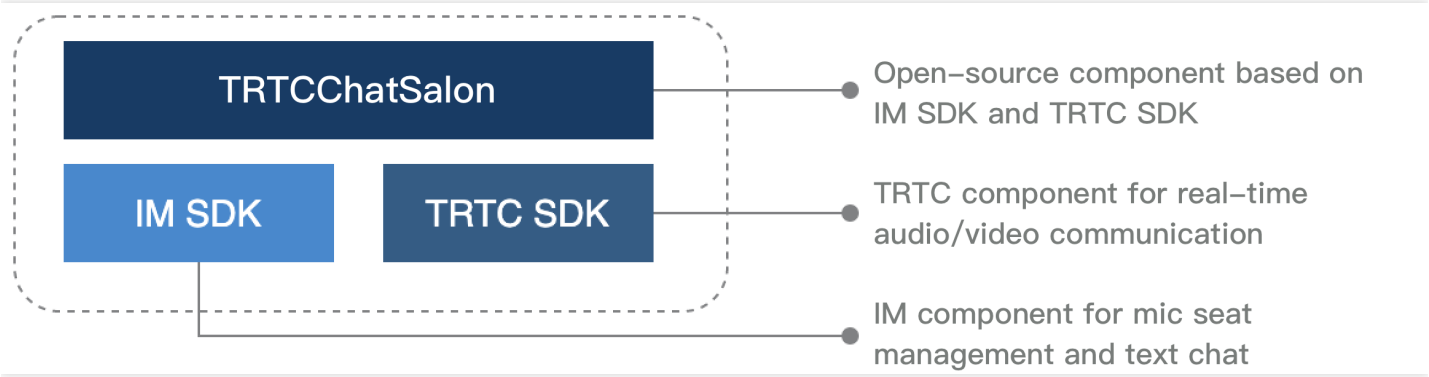
ファイルまたはフォルダ	機能の説明
base	UIに使用される基礎となるクラス。
list	リストページおよびルームページの作成。
room	メインルームページは、管理者とリスナーという2種類のインターフェースがあります。
widget	汎用ウィジェット。

UIカスタマイズの実装

ソースコード のtrtcchatsalondemoフォルダには2つのサブフォルダuiとmodelが含まれ、modelフォルダには再利用できるオープンソースコンポーネントTRTCChatSalonが含まれています。 `TRTCChatSalon.dart` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタ

©2013-2019 Tencent Cloud. All rights reserved.

マイズしたUIを実装することができます。



手順1：SDKの統合

ボイスサロンコンポーネントtrtcchatsalondemoは、TRTC SDKとIM SDKに依存しています。 `pubspec.yaml` を設定することで、更新を自動的にダウンロードできます。

プロジェクトの `pubspec.yaml` に次の依存関係を記述します。

```
dependencies:
tencent_trtc_cloud: 最新バージョン番号
tencent_im_sdk_plugin: 最新バージョン番号
```

手順2：権限の設定および難読化ルール

- iOS 端末
- Android 端末

`Info.plist` にマイクの権限申請を追加する必要があります。

```
<key>NSMicrophoneUsageDescription</key>
<string>通常の音声通話が行えるようにマイクの権限を承認します</string>
```

手順3：TRTCChatSalonコンポーネントのインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
lib/TRTCChatSalonDemo/model/
```

手順4：コンポーネントの作成およびログイン

1. `sharedInstance` インターフェースを呼び出すと、TRTCChatSalonコンポーネントのインスタンスオブジェクトを作成できます。

©2013-2019 Tencent Cloud. All rights reserved.

2. `registerListener` 関数を呼び出して、コンポーネントのイベント通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

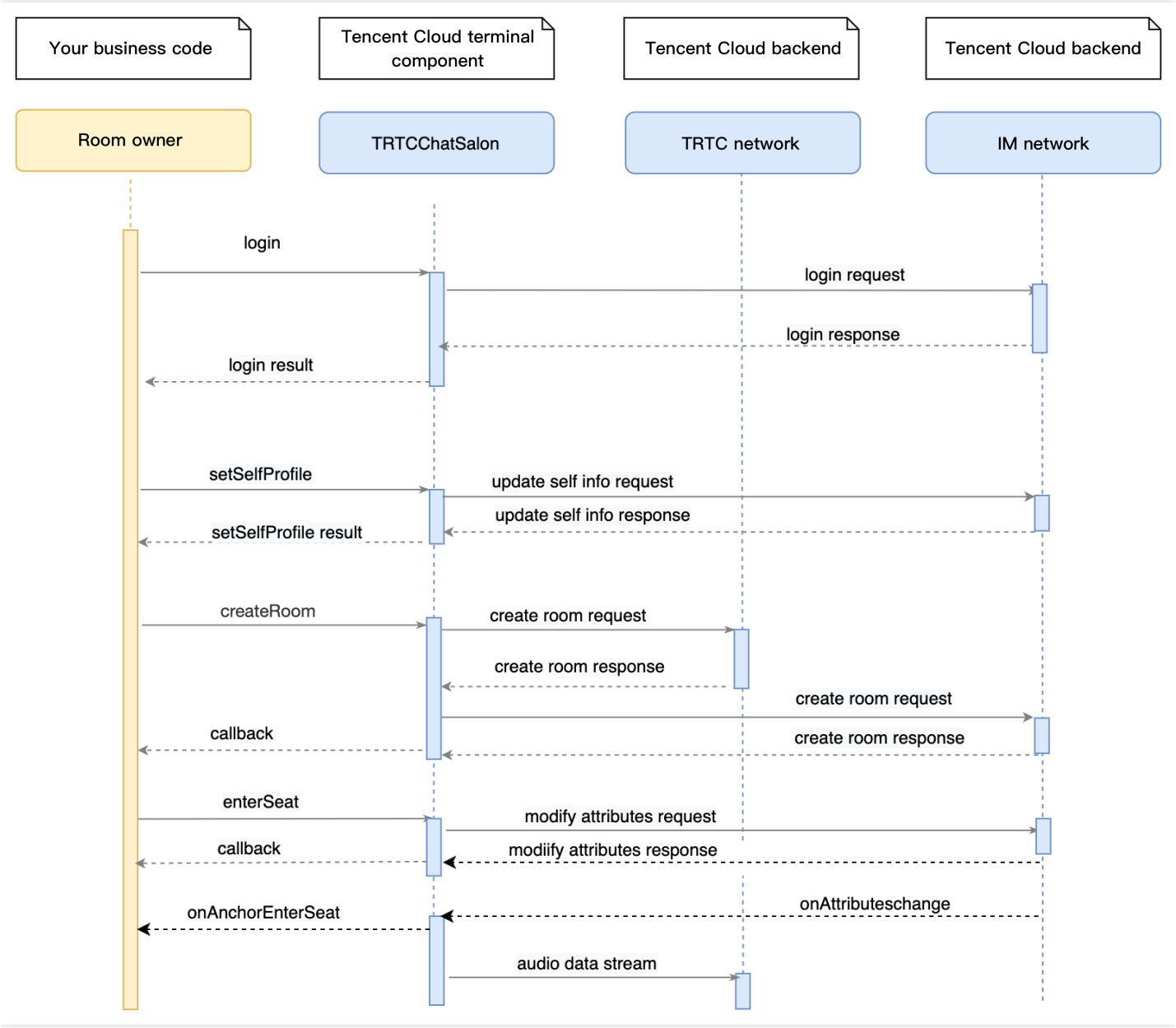
パラメータ名	機能
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。

```
TRTCChatSalon trtcVoiceRoom = await TRTCChatSalon.sharedInstance();
trtcVoiceRoom.registerListener(onVoiceListener);
ActionCallback resValue = await trtcVoiceRoom.login(
GenerateTestUserSig.sdkAppId,
userId,
GenerateTestUserSig.genTestSig(userId),
);
if (resValue.code == 0) {
//ログイン成功
}
```

手順5：管理者側での配信開始

1. 管理者は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 管理者は、 `createRoom` を呼び出して新しいボイスサロンを作成します。このとき、ルームID、ルーム名などルームの属性情報を渡します。
3. 管理者は、メンバーが参加した `TRTCChatSalonDelegate.onAudienceEnter` のイベント通知を受信します。このとき、マイク集音は自動的に開始されます。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.管理者は、ニックネームおよびプロフィール画像を設定します
trtcVoiceRoom.setSelfProfile("my_name", "my_face_url", null);

// 2.管理者は、createRoomを呼び出してルームを作成します
ActionCallback resp = await trtcVoiceRoom.createRoom(
roomId,
RoomParam(
coverUrl: 'ルームカバー図のURL',
roomName: 'ルーム名',
),
);
if (resp.code == 0) {
//3.席の占有成功
```

©2013-2019 Tencent Cloud. All rights reserved.

```
}

// 4.席の占有に成功した後、 TRTCChatSalonDelegate.onAudienceEnterのイベント通知を受信します
onVoiceListener(type, param) async {
switch (type) {
case TRTCChatSalonDelegate.onAudienceEnter:
//リスナーの入室
break;
}
}
```

手順6：リスナー側の視聴

1. リスナー側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. リスナー側は、業務バックエンドから最新のボイスサロンのルームリストを取得します。

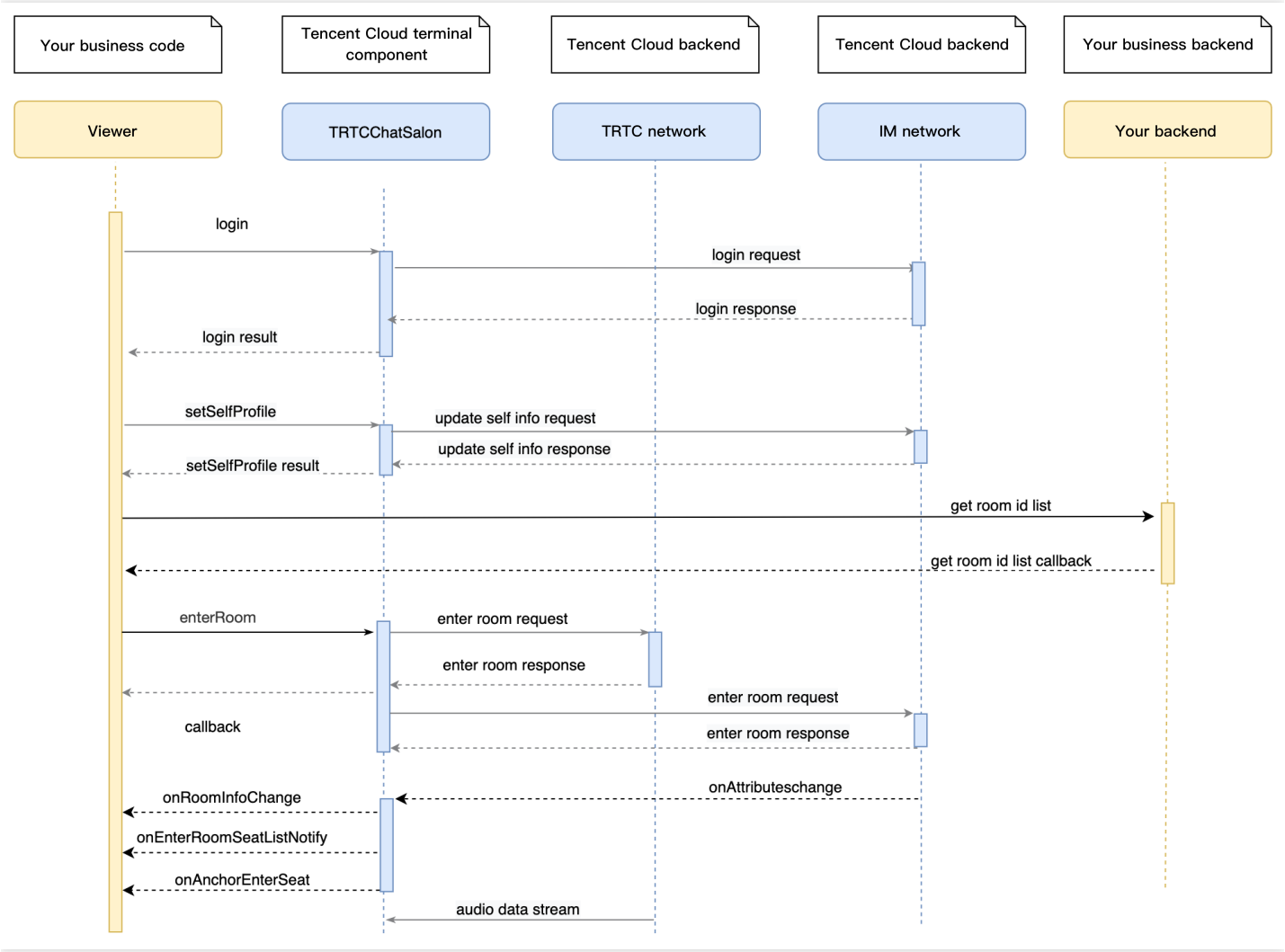
> 説明：
> Demoのボイスサロンリストは、デモンストレーション用のためだけのものです。ボイスサロンリストのビジネスロジックは様々です。Tencent Cloudでは現在ボイスサロンリストの管理サービスを提供していません。ご自身でボイスサロンリストを管理してください。

3. リスナー側は、 `getRoomList` を呼び出してルームの詳細情報を取得します。この情報は、管理者側が `createRoom` を呼び出してボイスサロンを作成するときに設定する簡単な説明情報です。

> 注意：
> ボイスサロンリストに十分に包括的な情報がある場合は、 `getRoomList` の呼び出しに関する手順をスキップできます。また、ルーム番号を入力すると、そのルームに入室できます。

4. 入室後、コンポーネントの `TRTCChatSalonDelegate.onAudienceEnter` と `TRTCChatSalonDelegate.onAudienceExit` という視聴者の入退室通知を受信します。イベントコールバックを監視した後、変更をUI上に更新できます。
5. 入室後、マイクリストのキャスターが参加した `TRTCChatSalonDelegate.onAnchorEnterMic` と `TRTCChatSalonDelegate.onAnchorLeaveMic` のイベントの通知も受信します。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.リスナーは、ニックネームおよびプロフィール画像を設定します
trtcVoiceRoom.setSelfProfile("my_name", "my_face_url");

// 2.業務バックエンドから取得したルームリストをroomListと仮定します
List<integer> roomList = GetRoomList();

// 3. getRoomInfoListを呼び出すことによって、ルームの詳細情報を取得します
RoomInfoCallback resp = await trtcVoiceRoom.getRoomInfoList(roomList);
if (resp.code == 0) {
//この時、自分のUIルームリストを更新することができます
}

// 4.roomIdを渡してルームに参加します
ActionCallback enterRoomResp =
await trtcVoiceRoom.enterRoom(_currentRoomId);
if (enterRoomResp.code == 0) {
//入室に成功
}
```

©2013-2019 Tencent Cloud. All rights reserved.

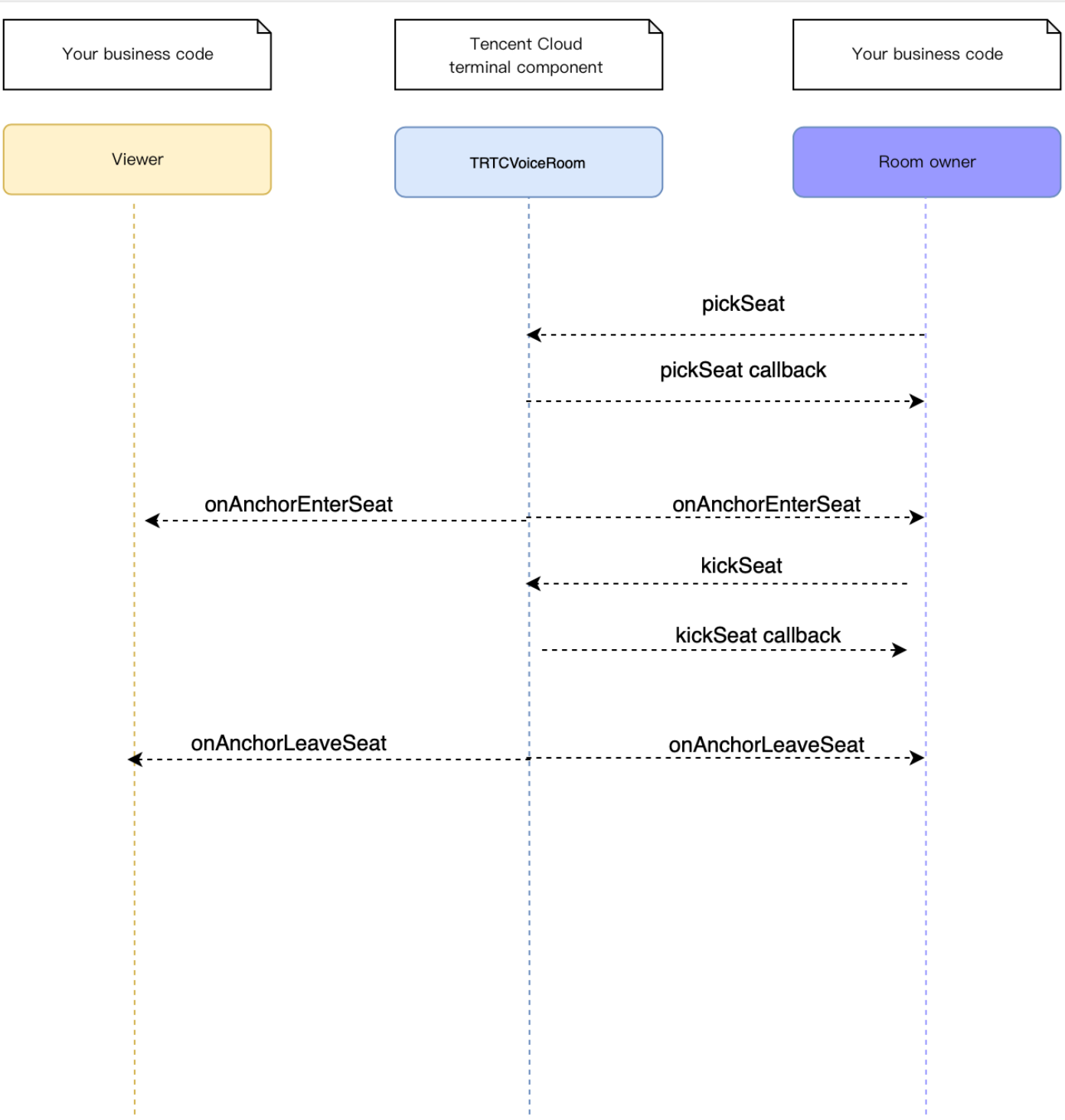
```
// 5.入室に成功すると、処理イベント通知を受信します
onVoiceListener(type, param) async {
switch (type) {
case TRTCChatSalonDelegate.onAudienceEnter:
//リスナーの入室
break;
case TRTCChatSalonDelegate.onAudienceExit:
//リスナーの退室
break;
case TRTCChatSalonDelegate.onAnchorLeaveMic:
//管理者の退室
break;
case TRTCChatSalonDelegate.onAnchorEnterMic:
//管理者の入室
break;
}
}
```

手順7：マイクのオン・オフ

- 管理者側
- リスナー側

1. `leaveMic` は自主的にマイク・オフにし、ルーム内の全メンバーは `onAnchorLeaveMic` のイベント通知を受信します。
2. `kickMic` は、対応するユーザーのuserIdを渡し、キックアウトしてマイク・オフできます。管理者は、キックアウトしてマイク・オフにし、ルーム内の全メンバーは `onAnchorLeaveMic` のイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.自主的なマイク・オフ
trtcVoiceRoom.leaveMic();

//2.キックアウトしてマイク・オフ
trtcVoiceRoom.kickMic(userId);
```

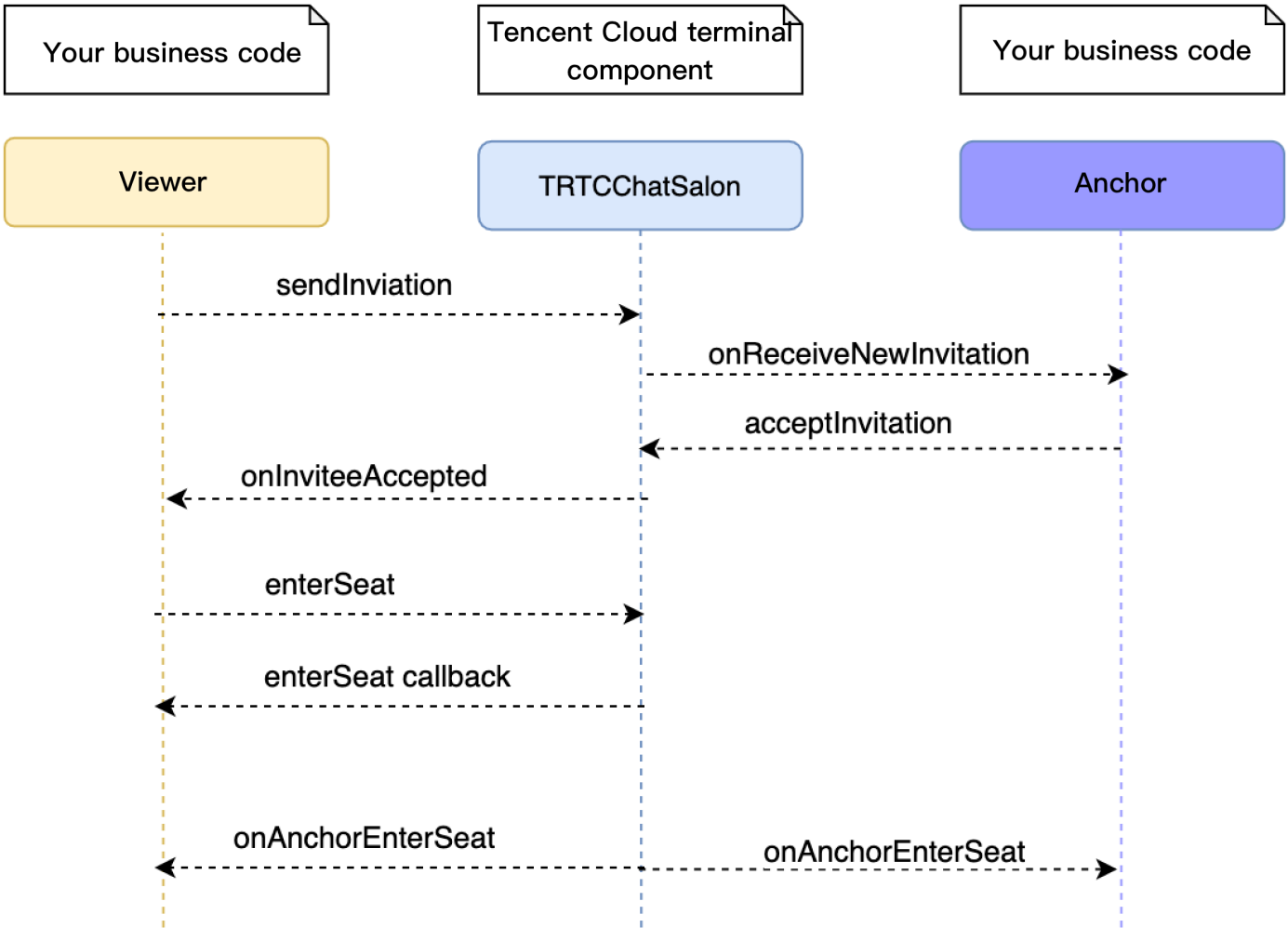
©2013-2019 Tencent Cloud. All rights reserved.

手順8：挙手によるマイク・オンシグナリングの使用リクエスト

Appにおいて相手の同意がなければ、次の操作の業務フローを実施できない場合は、招待シグナリングによって相応のサポートを行うことができます。

リスナーからのからの自主的なマイク・オン申請

1. リスナー側が `raiseHand` を呼び出し、挙手を申請します。
2. 管理者側は、 `onRaiseHand` のイベントを受信します。このとき、UIはウィンドウをポップアップし、管理者に同意するかどうか尋ねることができます。
3. 管理者が同意を選択した場合、 `agreeToSpeak` を呼び出し、userIdを渡します。
4. リスナー側は、 `onAgreeToSpeak` のイベント通知を受信し、 `enterMic` を呼び出して、マイク・オンします。




```
// リスナー側の視点
// 1.sendInvitationを呼び出し、マイク・オンをリクエストします
trtcVoiceRoom.raiseHand();
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// 2.招待のリクエスト同意を受信し、正式にマイク・オンになります
onVoiceListener(type, param) async {
switch (type) {
case TRTCChatSalonDelegate.onRaiseHand:
trtcVoiceRoom.enterMic();
break;
}
}

// 管理者側の視点
// 1.管理者がリクエストを受信します
onVoiceListener(type, param) async {
switch (type) {
case TRTCChatSalonDelegate.onAgreeToSpeak:
trtcVoiceRoom.agreeToSpeak();
break;
}
}
```

手順9：文字チャットおよび弾幕コメントの実装

`sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、このルーム内のすべてのキャスターおよびリスナーが `onRecvRoomTextMsg` のコールバックを受信することができます。
IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

```
// 発信側：テキストメッセージの発信
trtcVoiceRoom.sendRoomTextMsg("Hello Word!");
// 受信側：テキストメッセージのモニタリング
onVoiceListener(type, param) async {
switch (type) {
case TRTCChatSalonDelegate.onRecvRoomTextMsg:
//グループテキストメッセージを受信すると、テキストチャットルームを使用できます
break;
}
}
```

©2013-2019 Tencent Cloud. All rights reserved.

TRTCChatSalon(iOS)

最終更新日：：2022-04-21 16:20:17

TRTCChatSalonは、Tencent CloudのTRTCおよびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- 管理者は新しいボイスサロンを作成して配信を開始し、リスナーはボイスサロンに参加して視聴/インタラクションを行います。
- 管理者は、リスナーをマイク・オンに招待したり、マイク・オンのキャスターをキックアウトしたりすることもできます。
- リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、他者と音声インタラクションを行うことができます。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
- 各種のテキストメッセージやカスタムメッセージの送信をサポートします。カスタムメッセージは弾幕、「いいね」、ギフトなどを実装するために使用することができます。

TRTCChatSalonは、オープンソースのClassであり、Tencent Cloudの2つのクローズドソースであるSDKに依存しています。

- TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用します。
- IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCChatSalon API概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。
setDelegate	イベントコールバックを設定します。
delegateQueue	イベントコールバックが配置されているスレッドを設定します。
login	ログイン。
logout	ログアウト。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。nilの場合は、ルーム内全員の情報を取得します。

マイクのオン・オフインターフェース

API	説明
enterSeat	ユーザーが発言者になる（リスナー側/管理者ともに呼び出し可）。
leaveSeat	ユーザーが視聴者になる（キャスターが呼び出し）。
pickSeat	視聴者が発言できるように招待（管理者が呼び出し）。
kickSeat	キックアウトしてマイク・オフ（管理者が呼び出し）。

ローカルのオーディオ操作インターフェース

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
setAudioQuality	音質の設定。
muteLocalAudio	ローカルオーディオミュートの開始/停止。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	バックグラウンド・サウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

メッセージ送信関連インターフェース

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

招待シグナリング関連インターフェース

API	説明
sendInvitation	ユーザーに招待を送信。
acceptInvitation	招待の同意。
rejectInvitation	招待の辞退。
cancelInvitation	招待の取り消し。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCChatSalonDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが破棄された時のコールバック。
onRoomInfoChange	ボイスサロン情報変更のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

API	説明
onAnchorEnterSeat	発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。
onAnchorLeaveSeat	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。
onSeatMute	管理者のマイクミュート。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

シグナリングイベントのコールバック

API	説明
onReceiveNewInvitation	新規招待リクエストの受信。
onInviteeAccepted	被招待者が招待に同意。
onInviteeRejected	被招待者による招待の辞退。
onInvitationCancelled	招待者が招待を取り消し。

SDK基本関数

sharedInstance

TRTCChatSalonシングルトンオブジェクトを取得します。

```
/**
* TRTCChatSalonシングルトンオブジェクトを取得します
*
* - returns: TRTCChatSalonインスタンス
* - note: {@link TRTCChatSalon#destroySharedInstance()}を呼び出してシングルトンオブジェクトを廃棄することができます
*/
+ (instancetype)sharedInstance NS_SWIFT_NAME(shared());
```

destroySharedInstance

TRTCChatSalonシングルトンオブジェクトを廃棄します。

> 説明：
> インスタンスを破棄すると、外部キャッシュのTRTCChatSalonインスタンスは再利用できなくなります。
> あらためてsharedInstanceを呼び出し、新しいインスタンスを取得する必要があります。

©2013-2019 Tencent Cloud. All rights reserved.

```
/**
* TRTCChatSalonシングルトンオブジェクトを廃棄します
*
* - note: インスタンスの廃棄後、外部にキャッシュされた TRTCChatSalonインスタンスは再利用できなくなります。あらためて{@link TRTCChatSalon#sharedInstance()}を呼び出し、新しいインスタンスを取得する必要があります
*/
+ (void)destroySharedInstance NS_SWIFT_NAME(destroyShared());
```

setDelegate

TRTCChatSalonイベントコールバック。TRTCChatSalonDelegateを介してTRTCChatSalonの各種ステータス通知を受け取ることができます。

```
/**
* コンポーネントコールバックインターフェースの設定
*
* TRTCChatSalonDelegateを介してTRTCChatSalonの各種ステータス通知を受け取ることができます
*
* - parameter delegateコールバックインターフェース
* - note: TRTCChatSalonのコールバックイベントは、デフォルトではMain Queueでコールバックされます。イベントのコールバックが配置されているキューを指定する必要がある場合は、 {@link TRTCChatSalon#setDelegateQueue(queue)}を使用することができます
*/
- (void)setDelegate:(id<TRTCChatSalonDelegate>)delegate NS_SWIFT_NAME(setDelegate(delegate:));
```

説明：
setDelegateはTRTCChatSalonのプロキシコールバックです。

setDelegateQueue

イベントコールバックが配置されているスレッドキューを設定します。デフォルトでは、メインスレッドMainQueueに送信されます。

```
/**
* イベントコールバックが配置されているキューの設定
*
* - parameter queueキュー、TRTCChatSalonの各種ステータス通知コールバックは、指定したqueueに発信されます。
*/
- (void)setDelegateQueue:(dispatch_queue_t)queue NS_SWIFT_NAME(setDelegateQueue(queue:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
queue	dispatch_queue_t	TRTCChatSalonの各種ステータス通知は、指定したスレッドキューに発信されます。

login

ログイン。

```
- (void)login:(int)sdkAppID
userID:(NSString *)userID
userSig:(NSString *)userSig
callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(login(sdkAppID:userID:userSig:callback
:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	TRTCコンソール> 【[アプリケーション管理]（https://console.cloud.tencent.com/trtc/app）】>アプリケーション情報の中でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法についてはUserSigの計算方法をご参照ください。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

logout

ログアウト。

```
- (void)logout:(ActionCallback _Nullable)callback NS_SWIFT_NAME(logout(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ログアウトのコールバック。成功時にcodeは0になります。

©2013-2019 Tencent Cloud. All rights reserved.

setSelfProfile

個人情報の修正。

```
- (void)setSelfProfile:(NSString *)userName avatarURL:(NSString *)avatarURL callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(setSelfProfile(userName:avatarURL:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像のアドレス。
callback	ActionCallback	個人情報設定のコールバック。成功時にcodeは0になります。

ルーム関連インターフェース関数

createRoom

ルームの作成（管理者が呼び出し）。

```
- (void)createRoom:(int)roomID roomParam:(ChatSalonParam *)roomParam callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(createRoom(roomID:roomParam:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomIDを、一つのボイスサロンのルームリストにまとめることができます。Tencent Cloudでは現在、ボイスサロンのルームリストの管理サービスを行っていませんので、ご自身でボイスサロンのルームリストを管理してください。
roomParam	ChatSalonParam	ルーム情報は、ルームについて説明するための情報です。例えば、ルーム名、マイク情報、カバー情報などです。
callback	ActionCallback	ルームの作成結果のコールバック。成功時にcodeは0になります。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです。

1. 管理者は、 `createRoom` を呼び出して新しいボイスサロンを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否などルームの属性情報を渡します。

©2013-2019 Tencent Cloud. All rights reserved.

2. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
3. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。

```
- (void)destroyRoom:(ActionCallback _Nullable)callback NS_SWIFT_NAME(destroyRoom(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

enterRoom

入室（リスナーが呼び出し）

```
- (void)enterRoom:(NSInteger)roomID callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(enterRoom(roomID:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

リスナーが入室し視聴する際の通常の呼び出しプロセスは次のとおりです。

1. リスナーがサーバーから取得する最新のボイスサロンリストには、多くのボイスサロンのroomIdおよびルーム情報が含まれる場合があります。
2. リスナーはボイスサロンの1つを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
3. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
4. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` というイベント通知も受信します。

exitRoom

©2013-2019 Tencent Cloud. All rights reserved.

ルームから退出します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

```
- (void)getRoomInfoList:(NSArray<NSNumber *> *)roomIdList callback:(ChatSalonInfoCallback _Nullable)callback NS_SWIFT_NAME(getRoomInfoList(roomIdList:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIdList	List<Integer>	ルームナンバーリスト。
callback	ChatSalonInfoCallback	ルーム詳細情報のコールバック。

getUserInfoList

指定されたuserIdのユーザー情報を取得します。

```
- (void)getUserInfoList:(NSArray<NSString *> * _Nullable)userIDList callback:(ChatSalonUserListCallback _Nullable)callback NS_SWIFT_NAME(getUserInfoList(userIDList:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIdList	List<String>	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	ChatSalonUserListCallback	ユーザーの詳細情報のコールバック。

マイクのオン・オフインターフェース

enterSeat

ユーザーが発言者になる（リスナー側/管理者ともに呼び出し可）します。

> 説明：
> マイク・オンが成功した後、ルーム内の全メンバーは `onAnchorEnterSeat` というイベント通知を受信します。

```
- (void)enterSeat:(ActionCallback _Nullable)callback
NS_SWIFT_NAME(enterSeat(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

そのインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーが管理者に同意を申請しなければマイク・オンできないユースケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、 `onInvitationAccept` を受信するとその関数を呼び出せるようになります。

leaveSeat

ユーザーが視聴者になる（キャスターが呼び出し）。

> 説明：
> マイク・オフが成功した後、ルーム内の全メンバーは `onAnchorLeaveSeat` というイベント通知を受信します。

```
- (void)leaveSeat:(ActionCallback _Nullable)callback NS_SWIFT_NAME(leaveSeat(callback:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

pickSeat

視聴者が発言できるように招待（管理者が呼び出し）。

> 説明：
> 管理者が視聴者を発言できるように招待すると、ルーム内の全メンバーは `onAnchorEnterSeat` というイベント通知を受信します。

```
- (void)pickSeat:(NSString *)userID callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(pickSeat(userID:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	ユーザーID 。
callback	ActionCallback	操作コールバック。

そのインターフェースを呼び出すと、すぐにマイクリストが修正されます。管理者がリスナーの同意がなければマイク・オンできないユースケースの場合は、まず `sendInvitation` を呼び出してからリスナーに申請し、 `onInvitationAccept` を受信すると、その関数をコールできるようになります。

kickSeat

キックアウトしてマイク・オフ（管理者が呼び出し）。

> 説明：
> 管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

```
- (void)kickSeat:(NSString *)userID callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(kickSeat(userID:callback:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	キックアウトしてマイク・オフにする必要のあるユーザーID。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

ローカルのオーディオ操作インターフェース

startMicrophone

マイクの集音開始。

```
- (void)startMicrophone;
```

stopMicrophone

マイクの集音停止。

```
- (void)stopMicrophone;
```

setAudioQuality

音質の設定。

```
- (void)setAuidoQuality:(NSInteger)quality NS_SWIFT_NAME(setAuidoQuality(quality:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	int	オーディオ品質。詳細については、TRTC SDKをご参照ください。

muteLocalAudio

ローカルの音声のミュート/ミュート取り消し。

```
- (void)muteLocalAudio:(BOOL)mute NS_SWIFT_NAME(muteLocalAudio(mute:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	ミュート/ミュート取り消し。詳細は TRTC SDKをご参照ください。

setSpeaker

スピーカーの起動設定。

```
- (void)setSpeaker:(BOOL)userSpeaker NS_SWIFT_NAME(setSpeaker(userSpeaker:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userSpeaker	boolean	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。

```
- (void)setAudioCaptureVolume:(NSInteger)voluem NS_SWIFT_NAME(setAudioCaptureVolume(volume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、0 - 100、 デフォルト100。

setAudioPlayoutVolume

再生音量の設定。

```
- (void)setAudioPlayoutVolume:(NSInteger)volume NS_SWIFT_NAME(setAudioPlayoutVolume(volume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	再生音量、0 - 100、 デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)muteRemoteAudio:(NSString *)userID mute:(BOOL)mute NS_SWIFT_NAME(muteRemoteAudio(userId:mute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	指定ユーザーID。
mute	boolean	true：ミュート起動；false：ミュート停止。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。

```
- (void)muteAllRemoteAudio:(BOOL)isMute NS_SWIFT_NAME(muteAllRemoteAudio(isMute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMute	boolean	true：ミュートを起動、false：ミュートを停止。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

BGMサウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

```
- (TXAudioEffectManager * _Nullable)getAudioEffectManager;
```

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

```
- (void)sendRoomTextMsg:(NSString *)message callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendRoomTextMsg(message:callback:));
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

```
- (void)sendRoomCustomMsg:(NSString *)cmd message:(NSString *)message callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendRoomCustomMsg(cmd:message:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

招待シグナリング関連インターフェース

sendInvitation

ユーザーに招待を送信。

```
- (NSString *)sendInvitation:(NSString *)cmd
userID:(NSString *)userID
content:(NSString *)content
callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendInvitation(cmd:userId:content:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	業務カスタマイズコマンド。
userID	String	招待ユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
content	String	招待コンテンツ。
callback	ActionCallback	送信結果のコールバック。

戻り値：

戻り値	タイプ	意味
inviteId	String	今回の招待IDの識別に使用。

acceptInvitation

招待の同意。

```
- (void)acceptInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(acceptInvitation(identifier:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
identifier	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

rejectInvitation

招待の拒否。

```
- (void)rejectInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(rejectInvitation(identifier:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
identifier	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

cancelInvitation

招待の取り消し。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)cancelInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(cancelInvitation(identifier:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
identifier	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

TRTCChatSalonDelegateイベントコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

```
- (void)onError:(int)code
message:(NSString*)message
NS_SWIFT_NAME(onError(code:message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラー情報。

onWarning

警告のコールバック。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onWarning:(int)code
message:(NSString *)message
NS_SWIFT_NAME(onWarning(code:message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	警告メッセージ。

onDebugLog

Logコールバック。

```
- (void)onDebugLog:(NSString *)message
NS_SWIFT_NAME(onDebugLog(message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	ログ情報。

ルームイベントのコールバック

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。

```
- (void)onRoomDestroy:(NSString *)message
NS_SWIFT_NAME(onRoomDestroy(message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	String	ルームID。

onRoomInfoChange

©2013-2019 Tencent Cloud. All rights reserved.

入室に成功後、このインターフェースをコールバックします。roomInfoの情報は、管理者がルームを作成するときに渡されます。

```
- (void)onRoomInfoChange:(ChatSalonInfo *)roomInfo
NS_SWIFT_NAME(onRoomInfoChange(roomInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomInfo	ChatSalonInfo	ルーム情報。

onUserVolumeUpdate

音量レベルリマインダを有効にして、各メンバーの音量を通知します。

```
- (void)onUserVolumeUpdate:(NSArray<TRTCVolumeInfo *> *)userVolumes totalVolume:(NSInteger)totalVolume
NS_SWIFT_NAME(onUserVolumeUpdate(userVolumes:totalVolume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userVolumes	NSArray<TRTCVolumeInfo *>	各ユーザーの音量情報。
totalVolume	int	全体の音量情報。

マイクコールバック

onAnchorEnterSeat

発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。

```
- (void)onAnchorEnterSeat:(ChatSalonUserInfo *)user
NS_SWIFT_NAME(onAnchorEnterSeat(user:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user	ChatSalonUserInfo	マイク・オンのユーザーの詳細情報。

©2013-2019 Tencent Cloud. All rights reserved.

onAnchorLeaveSeat

視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。

```
- (void)onAnchorLeaveSeat:(ChatSalonUserInfo *)user
NS_SWIFT_NAME(onAnchorLeaveSeat(user:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
user	ChatSalonUserInfo	マイク・オンのユーザーの詳細情報。

onSeatMute

管理者のマイクミュート。

```
- (void)onSeatMute:(NSString *)userID
isMute:(BOOL)isMute
NS_SWIFT_NAME(onSeatMute(userID:isMute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userID	String	マイクのユーザーid。
isMute	boolean	true：マイクミュート、false：ミュート解除。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。

```
- (void)onAudienceEnter:(ChatSalonUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceEnter(userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	ChatSalonUserInfo	入室したリスナーの情報。

©2013-2019 Tencent Cloud. All rights reserved.

onAudienceExit

リスナー退室通知の受信。

```
- (void)onAudienceExit:(ChatSalonUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceExit(userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	ChatSalonUserInfo	退室したリスナーの情報。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

```
- (void)onRecvRoomTextMsg:(NSString *)message
userInfo:(ChatSalonUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomTextMsg(message:userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
userInfo	ChatSalonUserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージの受信。

```
- (void)onRecvRoomCustomMsg:(NSString *)cmd
message:(NSString *)message
userInfo:(ChatSalonUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomCustomMsg(cmd:message:userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
userInfo	ChatSalonUserInfo	送信者のユーザー情報。

招待シグナリングイベントのコールバック

onReceiveNewInvitation

新規招待リクエストの受信。

```
- (void)onReceiveNewInvitation:(NSString *)identifier
inviter:(NSString *)inviter
cmd:(NSString *)cmd
content:(NSString *)content
NS_SWIFT_NAME(onReceiveNewInvitation(identifier:inviter:cmd:content:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
identifier	String	招待ID。
inviter	String	招待者のユーザーID。
cmd	String	業務指定のコマンドワードは、開発者がカスタマイズします。
content	String	業務指定のコンテンツ。

onInviteeAccepted

被招待者が招待に同意。

```
- (void)onInviteeAccepted:(NSString *)identifier
invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeAccepted(identifier:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
identifier	String	招待ID。
invitee	String	被招待者のユーザーID。

onInviteeRejected

被招待者による招待の拒否。

```
- (void)onInviteeRejected:(NSString *)identifier
invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeRejected(identifier:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
identifier	String	招待ID。
invitee	String	被招待者のユーザーID。

onInvitationCancelled

招待者が招待を取り消し。

```
- (void)onInvitationCancelled:(NSString *)identifier
invitee:(NSString *)invitee NS_SWIFT_NAME(onInvitationCancelled(identifier:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
identifier	String	招待ID。
invitee	String	招待者のユーザーID。

onInvitationTimeout

招待のタイムアウト。

```
- (void)onInvitationTimeout:(NSString *)identifier;
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
identifier	String	招待ID。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCChatSalon(Flutter)

最終更新日：：2021-10-12 09:40:10

.TRTCChatSalonは、Tencent CloudのTRTCおよびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- 管理者は新しいボイスサロンを作成して配信を開始し、リスナーはボイスサロンに参加して視聴/インタラクションを行います。
- 管理者は、リスナーのマイク・オンに同意したり、マイク・オンのキャスターをキックアウトしたりすることもできます。
- リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、他者と音声インタラクションを行うことができます。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
- 各種のテキストメッセージの送信をサポートします。

TRTCChatSalonは、オープンソースのClassであり、Tencent Cloudの2つのクローズドソースであるSDKに依存しています。具体的な実装プロセスについては、ボイスサロン（Flutter）をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用します。
- IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCChatSalon API概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。
registerListener	イベント監視を設定します。
unRegisterListener	イベント監視を廃棄します。
login	ログイン。
logout	ログアウト。
setSelfProfile	個人情報を修正します。

©2013-2019 Tencent Cloud. All rights reserved.

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getRoomMemberList	ルーム内のすべてのユーザー情報を取得します。
getArchorInfoList	ルームのキャスターリストを取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。

マイクのオン・オフインターフェース

API	説明
enterMic	リスナーのマイク・オン。
enterMic	キャスターのマイク・オフ。
muteMic	任意のマイクのミュート/ミュート解除（キャスターが呼び出し）。
kickMic	キックアウトしてマイク・オフ（管理者が呼び出し）。

ローカルのオーディオ操作インターフェース

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
muteLocalAudio	ローカルオーディオミュートの開始/停止。
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
setAudioPlayoutVolume	再生音量の設定。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	バックグラウンド・サウンドエフェクト管理オブジェクトTXAudioEffectManagerの取得。

メッセージ送信関連インターフェース

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャストには、通常、弾幕によるチャットを使用します。

マイク・オン申請シグナリング関連インターフェース

API	説明
raiseHand	リスナーがマイク・オンを申請しました。
agreeToSpeak	管理者はマイク・オンに同意しました。
refuseToSpeak	管理者はマイク・オンを拒否しました。

TRTCChatSalonDelegate API概要

一般的なイベントコールバック

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onKickedOffline	キックアウトされてオフラインになります。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが廃棄されたときのコールバック。
onAnchorListChange	キャスターリストに変更があったときのコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

API	説明
onAnchorEnterMic	発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。
onAnchorLeaveMic	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。
onMicMute	キャスターのマイクミュート。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。

©2013-2019 Tencent Cloud. All rights reserved.

マイク・オン申請シグナリングイベントのコールバック

API	説明
onRaiseHand	挙手してマイク・オン申請しているリスナーがいます。
onAgreeToSpeak	リスナーが挙手を申請すると、管理者から挙手に同意するコールバックを受け取ります。
onRefuseToSpeak	リスナーが挙手を申請すると、管理者が挙手のコールバックを拒否します。

SDK基本関数

sharedInstance

TRTCChatSalonシングルトンオブジェクトを取得します。

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

destroySharedInstance

TRTCChatSalonシングルトンオブジェクトを廃棄します。

> 説明：
> インスタンスを破棄すると、外部キャッシュのTRTCChatSalonインスタンスは再利用できなくなります。
> あらためてsharedInstanceを呼び出し、新しいインスタンスを取得する必要があります。

```
static void destroySharedInstance()
```

registerListener

イベント監視の設定

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

> 説明：
> setDelegateはTRTCChatSalonのプロキシコールバックです。

©2013-2019 Tencent Cloud. All rights reserved.

unRegisterListener

コンポーネントイベントの監視インターフェースを削除します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
func	VoiceListenerFunc	TRTCChatSalonの各種ステータス通知は、指定したfunc関数に発信されます

login

ログイン。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	TRTCコンソール > 【アプリケーション管理】> アプリケーション情報の中でSDKAppIDを確認できます。
user	String	現在のユーザーID、文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。

logout

ログアウト。

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

setSelfProfile

個人情報の修正。

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

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像URL。

ルーム関連インターフェース関数

createRoom

ルームを作成します（新しいルームを作成するときに呼び出します）。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	intt	ルームIDは、ご自身でアサインし、統一管理する必要があります。複数のroomIDを、一つのボイスサロンのルームリストにまとめることができます。Tencent Cloudでは現在、ボイスサロンのルームリストの管理サービスを行っていませんので、ご自身でボイスサロンのルームリストを管理してください。
roomParam	RoomParam	ルーム情報は、ルームについて説明するための情報です。例えば、ルーム名、カバー情報などです。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです。

1. 管理者は `createRoom` を呼び出して新しいボイスサロンを作成し、ルームIDなどルームの属性情報を渡します。
2. 管理者はマイクリストのメンバーが参加した `onAnchorEnterMic` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。

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

enterRoom

©2013-2019 Tencent Cloud. All rights reserved.

入室（リスナーが呼び出し）

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームID。

リスナーが入室し視聴する際の通常の呼び出しプロセスは次のとおりです。

1. リスナーがサーバーから取得する最新のボイスサロンリストには、多くのボイスサロンのroomIdおよびルーム情報が含まれる場合があります。
2. リスナーはボイスサロンの1つを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
3. 入室後、 `getArchorInfoList` を確認してキャスターリストを取得してから、 `getRoomMemberList` に基づいて、すべてのユーザーリストを取得してキャスターリストを差し引けば、リスナーリストを取得することができます。
4. 入室後、マイクリストにキャスターが参加した `onAnchorEnterMic` というイベント通知も受信します。

exitRoom

ルームから退出します。

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

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

```
Future<RoomInfoCallback> getRoomInfoList(List<String> roomIdList)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomIdList	List&dxlt;String&dxgt;	ルームナンバーリスト。

getRoomMemberList

ルーム内の全メンバーのリストを取得します。

> 説明：
> IMライブストリーミンググループは、デフォルトで直近メンバー31人のリストだけをプルすることができます。

```
Future<MemberListCallback> getRoomMemberList(double nextSeq)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
nextSeq	double	ページネーションプルフラグ。最初のプルに0と入力すると、コールバックは成功します。nextSeqがゼロでない場合は、ページネーションが必要であり、0になるまで渡して再度プルされます。

getArchorInfoList

ルーム内のキャスターリストを取得します。

```
Future<UserListCallback> getArchorInfoList()
```

getUserInfoList

指定されたuserIdのユーザー情報を取得します。

```
Future<UserListCallback> getUserInfoList(List<String> userIdList)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIdList	List&dxlt;String&dxgt;	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。

©2013-2019 Tencent Cloud. All rights reserved.

マイクのオン・オフインターフェース

enterMic

マイク・オン（リスナー側/管理者ともに呼び出し可）します。

> 説明：
> マイク・オンが成功した後、ルーム内の全メンバーは `onAnchorEnterSeat` というイベント通知を受信します。

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

このインターフェースを呼び出すと、マイクリストが直ちに変更されます。リスナーがまず `raiseHand` を呼び出して管理者に申請する必要がある場合は、 `onAgreeToSpeak` を受信した後にこの関数を呼び出します。

leaveMic

キャスターのマイク・オフ。

> 説明：
> マイク・オフが成功した後、ルーム内の全メンバーは `onAnchorLeaveMic` というイベント通知を受信します。

```
Future<ActionCallback> leaveMic()
```

muteMic

任意のマイクのミュート/ミュート解除（管理者が呼び出し）。

> 説明：
> マイク・オンの成功後、ルーム内の全メンバーは、 `onAnchorListChange` および `onMicMute` というイベント通知を受信します。

```
Future<ActionCallback> muteMic(bool mute)
```

©2013-2019 Tencent Cloud. All rights reserved.

kickMic

キックアウトしてマイク・オフ（管理者が呼び出し）。

> 説明：
> 管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは `onAnchorLeaveMic` というイベント通知を受信します。

```
Future<ActionCallback> kickMic(String userId)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	キックアウトしてマイク・オフにする必要のあるユーザーID。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

ローカルのオーディオ操作インターフェース

startMicrophone

マイクの集音開始。

```
void startMicrophone(int quality)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	int	音声品質。詳細は TRTC SDKをご参照ください。

stopMicrophone

マイクの集音停止。

```
void stopMicrophone()
```

muteLocalAudio

©2013-2019 Tencent Cloud. All rights reserved.

ローカルのオーディオのミュート/ミュート取り消し。

```
void muteLocalAudio(bool mute)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	bool	ミュート/ミュート取り消し。詳細については、TRTC SDKをご参照ください。

setSpeaker

スピーカーの起動設定。

```
void setSpeaker(bool useSpeaker)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
useSpeaker	bool	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。

```
void setAudioCaptureVolume(int volume)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、0 - 100、 デフォルト100。

setAudioPlayoutVolume

再生音量の設定。

```
void setAudioPlayoutVolume(int volume)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
volume	int	再生音量、0 - 100、 デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。

```
void muteRemoteAudio(String userId, bool mute)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定ユーザーID。
mute	bool	true：ミュート起動、false：ミュート停止。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。

```
void muteAllRemoteAudio(bool mute)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	bool	true：ミュート起動、false：ミュート停止。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

バックグラウンド・サウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

```
TXAudioEffectManager getAudioEffectManager()
```

メッセージ送信関連インターフェース関数

sendRoomTextMsg

©2013-2019 Tencent Cloud. All rights reserved.

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。

マイク・オン申請シグナリング関連インターフェース

raiseHand

リスナーがマイク・オンを申請しました。

```
void raiseHand()
```

agreeToSpeak

管理者がマイク・オンに同意しました。

```
Future<ActionCallback> agreeToSpeak(String userId)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

refuseToSpeak

管理者がユーザーのマイク・オンを拒否しました。

```
Future<ActionCallback> refuseToSpeak(String userId)
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCChatSalonDelegateイベントコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
errCode	int	エラーコード。
errMsg	String	エラーメッセージ。
extraInfo	String	拡張情報フィールドでは、個別のエラーコードによる追加の情報が問題個所の特定に役立つ場合があります。

onWarning

警告のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
warningCode	int	エラーコード。
warningMsg	String	警告メッセージ。
extraInfo	String	拡張情報フィールドでは、個別のエラーコードによる追加の情報が問題個所の特定に役立つ場合があります。

onKickedOffline

他のユーザーが同じアカウントでログインすると、キックアウトされてオフラインになります。

©2013-2019 Tencent Cloud. All rights reserved.

ルームイベントのコールバック

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。

onAnchorListChange

キャスターリスト変更発生の通知。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
mute	bool	ミュート状態。

onUserVolumeUpdate

音量レベルリマインダを有効にして、各メンバーの音量を通知します。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
volume	int	音量の大きさ。値：0～100。

マイクコールバック

onAnchorEnterMic

マイク・オンのメンバー。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	入室したユーザーID。
userName	String	ユーザーのニックネーム。
userAvatar	String	プロフィール画像URL。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
mute	bool	マイクステータス。デフォルトではマイク・オンになっています。

onAnchorLeaveMic

マイク・オフのメンバー。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ルームを退出したユーザーID。

onMicMute

管理者のマイクミュートの有無。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	マイク・オフのユーザーID。
mute	bool	マイクステータス。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	マイク・オンのユーザーID。
userName	String	ユーザーのニックネーム。
userAvatar	String	プロフィール画像URL。

onAudienceExit

リスナー退室通知の受信。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	マイク・オフのユーザーID。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
sendId	String	送信者のユーザーID。
userAvatar	String	送信者のユーザープロフィール画像URL。
userName	String	送信者のユーザーニックネーム。

挙手申請シグナリングイベントのコールバック

onRaiseHand

新規マイクリクエストの受信。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	挙手申請したユーザーID。

onAgreeToSpeak

管理者によるマイク・オン同意のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	String	管理者のユーザーID。

onRefuseToSpeak

管理者によるマイク・オン拒否のコールバック。

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	管理者のユーザーID。

©2013-2019 Tencent Cloud. All rights reserved.

リアルタイム双方向対話型授業
リアルタイムインタラクティブ授業
(Electron)

最終更新日：：2022-04-02 16:29:01

Demo体験

Windows 版　MacOS 版

デモンストレーション

すでに作成してあるAppインストールパッケージをダウンロード、インストールして、リアルタイムインタラクティブ授業の機能を体験することができます。基本的なオーディオビデオ通話、画面共有、ホワイトボード、テキストチャットなどの基本機能を提供するだけでなく、全員のマイクミュート、学生の挙手による発言申請、先生の学生への発言要請、点呼、サインインなどといった高度な機能の提供も実現しました。

リアルタイムインタラクティブ授業クイックスタートのソースコード

ステップ1：アプリケーションの作成、SDKAppIDとキーの取得

すでにTRTCのアプリケーションを作成している場合は、このステップをスキップし、以前作成したアプリケーションのSDKAppIDとキーを直接使用することができます。

1. TRTCコンソールにログインし、**開発支援** > **Demoクイックスタート******を選択し、**アプリケーションの作成**タブで `TestTRTC` などのアプリケーション名を入力して、**作成**ボタンをクリックします。

2. **ソースコードのダウンロード**タブをスキップし、**次のステップ**ボタンを直接クリックして**設定の変更**タブに進み、次のページに表示されたSDKAppIDとキーを以後のステップで使用できるよう、記録します。

ステップ2：Instant Messaging（IM）の設定

©2013-2019 Tencent Cloud. All rights reserved.

説明：
リアルタイムインタラクティブ授業はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期してアクティブ化することができます。IMは付加価値サービスであり、課金ルールの詳細については、IMの料金説明をご参照ください。

1. **関連するクラウドサービス**メニューに進み、下図の**IMアプリケーション**をクリックしてIMアプリケーション管理ページにジャンプします。

2. 先ほど作成したアプリケーションを見つけてクリックするとアプリケーション管理ページに進みます。

3. メニューの**機能設定 > ログインとメッセージ**を開くと、下図のように表示されますので、**ログイン設定**エリアの**編集**リンクをクリックし、**Web端末の同時オンライン可能数**を2以上の値に設定します（現在このアプリケーションで同時にログインを必要とするWeb IMインスタンスの数は最大2つで、引き続き使用する際の設定は追加可能です）。

ステップ3：実行環境の準備

このコードプロジェクトの実行はnode.jsとyarnに依存します。

1. **node.jsのインストール**
   node.jsは14.16.0以降のバージョンの使用をお勧めします。インストール完了後、コマンドライン端末で以下のコマンドを実行してnode.jsのバージョンを確認することをお勧めします。

```
node --version
```

2. **yarnのインストール**

- node.jsのバージョンが16.10より前の場合は、コマンドライン端末で以下のコマンドを実行してyarnをインストールします。

```
npm i -g corepack
```

- node.jsのバージョンが16.10以降の場合は、コマンドライン端末で以下のコマンドを実行してyarnをインストールします。

```
corepack enable
```

©2013-2019 Tencent Cloud. All rights reserved.

注意：
Window10、11で権限不足のエラーメッセージが表示されたときは、管理者としてcmdで実行してみてください。

ステップ4：コードクローンプロジェクト

直接コードのダウンロードができ、解凍後にコードディレクトリ `trtc-education-electron` に進むか、またはgitツールを使用してコードプロジェクトをクローンすることができます。gitツールを使用したコードクローンプロジェクトでは、コマンドライン端末で以下のコマンドを実行してください。

```
git clone https://github.com/TencentCloud/trtc-education-electron.git
cd trtc-education-electron
```

ステップ5：SDKAppIDとキーの設定

1. `src/main/config/generateUserSig.js` ファイルを見つけて開きます。
2. 身分認証発行用のユーザー署名UserSigに使用するために、 `generateUserSig.js` ファイルの関連パラメータを設定します。
   - SDKAPPID：デフォルトでは0です。ステップ1で作成したアプリケーションのSDKAppIDを設定してください。
   - SECRETKEY：デフォルトでは空文字列です。ステップ1で作成したアプリケーションのキーを設定してください。

注意：

- ここで言及したUserSigの新規作成ソリューションでは、クライアントコードでSECRETKEYを設定します。この手法のうちSECRETKEYは逆コンパイルによって逆向きにクラッキングされやすく、キーがいったん漏洩すると、攻撃者はTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのDemoクイックスタートおよび機能デバッグにのみ適合します。**
- UserSigの正しい発行方法は、UserSigの計算コードをサーバー側に統合し、App向けのインターフェースをに提供します。UserSigが必要なときは、Appからサービスサーバーにリクエストを発出し動的にUserSigを取得します。詳細については、サーバーでのUserSig新規作成をご参照ください。

ステップ6：開発モードの実行

コマンドライン端末で、コードディレクトリ `trtc-education-electron` に進み、以下のコマンドを実行します。

©2013-2019 Tencent Cloud. All rights reserved.

```
yarn
yarn start
```

注意：

- 初めてyarnコマンド実行の依存関係をインストールする際、Window10、Window11で権限不足のエラーメッセージが表示された場合は、管理者としてcmdで一度実行してみてください。その後は一般ユーザーとしてcmdまたはVisual Studio Code、WebStormといった統合開発ツール付属の端末で実行することができます。
- 依存関係のインストール中に、Electronのダウンロードが遅い、またはスタックするなどの問題が発生した場合は、お問い合わせして問題を解決できます。

ステップ7：インストールパッケージの作成、実行

コマンドライン端末で、コードディレクトリ `trtc-education-electron` に進み、以下のコマンドを実行してインストールパッケージを作成します。作成したインストールパッケージは `trtc-education-electron/build/release` ディレクトリにあり、インストールと実行が可能です。

```
yarn package
```

注意：
Macインストールパッケージの作成はMac PCのみを使用できます。Windowsインストールパッケージの作成はWindows PCのみを使用できます。

技術的なお問い合わせ

詳細については、お問い合わせにご連絡をいただくか、colleenyu@tencent.comにメールでご連絡ください。

参考ドキュメント

- SDK APIマニュアル
- SDK更新ログ
- Simple Demoソースコード

©2013-2019 Tencent Cloud. All rights reserved.

- API Exampleソースコード
- Electronについてのよくあるご質問

©2013-2019 Tencent Cloud. All rights reserved.

trtc-electron-education API

最終更新日：：2021-04-27 16:49:32

コンポーネント trtc-electron-education は、リアルタイム双方向対話型授業のシーンで使用するTencent Real-Time Communication（TRTC）とInstant Messaging（IM)の機能を対象として、二次パッケージングしたものです。基本の音声・ビデオチャット、画面共有の機能をパッケージングすると同時に、eラーニングのシーンを対象に教師のQA開始、学生の挙手、教師が学生を招待して皆の前で回答させる、回答の終了などの関連機能もパッケージングされています。
trtc-electron-educationは、1つのオープンソースのnpm コンポーネントであり、Tencent Cloudの2つのクローズドソースであるSDKに依存しています。具体的な実現プロセスは、リアルタイム双方向対話型授業(Electron)をご参照ください。

- TRTC SDK：TRTC SDKを低レイテンシーのオーディオビデオ通話コンポーネントとして使用しています。
- IM SDK： IM SDKをシグナリング情報の送信と処理に利用します。

コンポーネントの接続

```
// yarn方式の導入
yarn add trtc-electron-education
// npm方式の導入
npm i trtc-electron-education --save
```

コンポーネントのパラメータ

入力必須となる主要パラメータは下表で示すとおりです。

パラメータ	タイプ	説明
sdkAppId	number	入力必須のパラメータ。TRTCコンソール の中でSDKAppIDを確認できます。
userID	string	入力が必要なパラメータ。ユーザーIDは、お客様のアカウント体系から指定することが可能です。
userSig	string	入力が必要なパラメータ。ID署名（ログインパスワードに相当）は、userIDから算出します。具体的な計算方法は、UserSigの計算方法をご参照ください。

初期化の例示

©2013-2019 Tencent Cloud. All rights reserved.

```
import TrtcElectronEducation from 'trtc-electron-education';
const rtcClient = new TrtcElectronEducation({
sdkAppId: 1400***803,
userID: '123'
userSig: 'eJwtzM9****-reWMQw_'
});
```

コンポーネントの概要

基本関数

on(EventCode, handler, context)

コンポーネントが配布するイベントのモニタリングに利用します。
パラメータ：

パラメータ名	タイプ	説明
EventCode	String	イベントコード
handler	Function	モニタリング関数
context	Object	現在実行しているコンテキスト

例示：

```
const EVENT = rtcClient.EVENT
rtcClient.on(EVENT.MESSAGE_RECEIVED, onMessageReceived);
```

off(EventCode, handler)

イベントモニタリングの取消。
パラメータ：

パラメータ名	タイプ	説明
EventCode	String	イベントコード
handler	Function	取消が必要な名前付きモニタリング関数

例示：

©2013-2019 Tencent Cloud. All rights reserved.

```
const EVENT = rtcClient.EVENT
rtcClient.off(EVENT.MESSAGE_RECEIVED, onMessageReceived);
```

createRoom(params: CreateRoomParams)

教師のクラス作成。
パラメータ：

パラメータ名	タイプ	説明
classId	number	教室 ID
nickName	string	ニックネーム
avatar	string	プロファイル写真URL（未入力可）

例示：

```
interface CreateRoomParams {
classId: number; // 教室 ID
nickName: string; // ニックネーム
avatar?:string; // プロファイル画像URL
}
rtcClient.createRoom(params).then(() => {
})
```

destroyRoom(classId: number)

教師がクラスから退出し、ルームを廃棄します。
パラメータ：

パラメータ名	タイプ	説明
classId	number	教室 ID

例示：

```
rtcClient.destroyRoom(classId)
```

enterRoom(params: EnterRoomParams)

教師が授業を開始し、学生は教室に入り、聴講の準備をします。
パラメータ：

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ名	タイプ	説明
classId	number	教室 ID
nickName	string	ニックネーム
role	string	ロール。teacherは教師、studentは学生を表します。
avatar	string	プロファイル写真URL（未入力可）

例示：

```
interface EnterRoomParams {
role: string; // ロール
classId: number; // 教室 ID
nickName?: string; // ニックネーム
avatar?:string; // プロファイル画像URL
}
rtcClient.enterRoom(params).then(() => {
})
```

exitRoom(role:string, classId: number)

教師が授業を終了させ、学生は教室から退出します。
パラメータ：

パラメータ名	タイプ	説明
classId	number	教室 ID
role	string	ロール。teacherは教師、studentは学生を表します。

例示：

```
rtcClient.exitRoom(role, classId);
```

挙手の操作部分の関数

startQuestionTime(classId: number)

教師がQA時間を開始。教師側が通知を発信し、学生がQA開始のイベントを受信して、挙手の機能を起動します。
パラメータ：

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ名	タイプ	説明
classId	number	教室 ID

例示：

```
rtcClient.startQuestionTime(classId)
```

raiseHand()

学生の挙手。学生が挙手の通知を送信し、教師側が学生の挙手の通知を受信します。
パラメータ：無
例示：

```
rtcClient.raiseHand()
```

inviteToPlatform(userID: string)

教師が皆の前で回答させるために学生を招待。教師側が挙手リストの中の学生userIDを選択し、招待の通知を送信します。学生側は皆の前で回答する招待のイベントを受信し、招待を受けた学生のマイクが起動します。挙手する学生がいない場合は、教師が回答する学生を直接指名します。学生側が皆の前で回答する招待のイベントを受信し、指名された学生のマイクが起動します。
パラメータ：

パラメータ名	タイプ	説明
userID	string	ユーザーID

例示：

```
rtcClient.inviteToPlatform(userID).then(() => {
})
```

finishAnswering(userID: string)

回答の終了。教師側が学生側の回答を終了します。学生は回答終了の通知を受信し、指定された学生のマイク接続が停止します。
パラメータ：

パラメータ名	タイプ	説明
userID	string	ユーザーID

©2013-2019 Tencent Cloud. All rights reserved.

例示：

```
rtcClient.finishAnswering(userID).then(() => {
})
```

stopQuestionTime(classId: number)

QA時間の停止。教師側がQA時間を停止します。学生側は回答時間停止の通知を受信します。マイク接続済みの学生はマイク接続を停止し、挙手の機能をオフにする必要があります。
パラメータ：

パラメータ名	タイプ	説明
classId	number	教室 ID

例示：

```
rtcClient.stopQuestionTime(classId)
```

プッシュプルストリーム操作に関する関数

getScreenShareList()

画面共有のウィンドウのリストを取得。
パラメータ：無
例示：

```
rtcClient.getScreenShareList();
```

startScreenCapture(source: SourceParam)

画面共有を選択し、プッシュを開始。
パラメータ：

パラメータ名	タイプ	説明
type	number	キャプチャソースタイプ
sourceId	string	ソースIDの収集。ウィンドウについては、当該フィールドにウィンドウのハンドルを表示します。画面については、当該フィールドに画面IDを表示します。
sourceName	string	ソース名の収集、UTF8 エンコーディング

©2013-2019 Tencent Cloud. All rights reserved.

例示：

```
interface SourceParam {
type: number; // キャプチャソースタイプ
sourceId: string; // ソースIDの収集
sourceName: string; // ソース名の収集、UTF8 エンコーディング
}
rtcClient.startScreenCapture({
type,
sourceId,
sourceName
})
```

startRemoteView(params: RemoteParams)

リモートのビデオ画面または画面共有の画面の表示を開始。
パラメータ：

パラメータ名	タイプ	説明
userID	string	ユーザーID
streamType	number	画面タイプ。1は大画面、2は小画面、3は画面共有を表します。
view	HTMLElement	表示画面をローディングするDOM

例示：

```
interface RemoteParams {
userID: string; // ユーザーID
streamType: number; // 画面タイプ。1-大画面、2-小画面、3-画面共有
view: HTMLElement; //表示画面をロードするDOM
}
const view = document.getElementById('localVideo');
rtcClient.startRemoteView({
userID: userID,
streamType: 1,//1-大画面、2-小画面、3-画面共有
view: view
});
```

stopRemoteView(params: StopRemoteParams)

リモートのビデオ画面または画面共有の画面の表示を停止すると、同時に当該リモートユーザーのデータストリームを再びプルしなくなります。
パラメータ：

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ名	タイプ	説明
userID	string	ユーザーID
streamType	number	画面タイプ。1は大画面、2は小画面、3は画面共有を表します。

例示：

```
interface StopRemoteParams {
userID: string; // ユーザーID
streamType: number; // 画面タイプ。1-大画面、2-小画面、3-画面共有
}
rtcClient.stopRemoteView({
userID: userID,
streamType: 1 //1-大画面、2-小画面、3-画面共有
});
```

メッセージ送受信に関する関数

sendTextMessage(params: MessageParams)

チャットルームのメッセージ送信。
パラメータ：

パラメータ名	タイプ	説明
classId	number	教室 ID
message	string	テキストメッセージ

例示：

```
interface MessageParams {
classId: number; // 教室 ID
message: string; // メッセージテキスト
}
rtcClient.sendTextMessage(params).then(() => {
})
```

sendCustomMessage(userID: string, data: string)

カスタマイズ C2C メッセージの送信。
パラメータ：

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ名	タイプ	説明
userID	string	ユーザーID
data	string	カスタマイズメッセージ

例示：

```
rtcClient.sendCustomMessage(userID, JSON.stringify(params)
```

sendGroupCustomMessage(classId: number, data: string)

カスタマイズグループメッセージの送信。
パラメータ：

パラメータ名	タイプ	説明
classId	number	教室 ID
data	string	カスタマイズメッセージ

例示：

```
rtcClient.sendGroupCustomMessage(classId, JSON.stringify(params))
```

デバイス操作に関する関数

openCamera(view: HTMLElement)

カメラの起動。
パラメータ：

パラメータ名	タイプ	説明
view	HTMLElement	表示画面をロードするDOM

例示：

```
const domEle = document.getElementById('localVideo');
rtcClient.openCamera(domEle);
```

closeCamera()

©2013-2019 Tencent Cloud. All rights reserved.

カメラの停止。
パラメータ：無
例示：

```
rtcClient.closeCamera();
```

getCameraList()

カメラリストの取得。
パラメータ：無
例示：

```
rtcClient.getCameraList()
```

setCurrentCamera(deviceId:string)

カメラの設定。パラメータは getCameraDevicesListの中から取得できるデバイスIDです。
パラメータ：

パラメータ名	タイプ	説明
deviceId	string	デバイスID

例示：

```
rtcClient.setCurrentCamera(deviceId)
```

openMicrophone()

マイクをオンにします。
パラメータ：無
例示：

```
rtcClient.openMicrophone();
```

closeMicrophone()

マイクをオフにします。
パラメータ：無
例示：

©2013-2019 Tencent Cloud. All rights reserved.

```
rtcClient.closeMicrophone();
```

getMicrophoneList()

マイクのデバイスリストを取得します。
パラメータ：無
例示：

```
rtcClient.getMicrophoneList()
```

setCurrentMicDevice(micId:string)

マイクの設定。パラメータは getMicDevicesListの中から取得できるデバイスIDです。
パラメータ：

パラメータ名	タイプ	説明
micId	string	デバイスID

例示：

```
rtcClient.setCurrentMicDevice(micId)
```

setBeautyStyle(params: BeautyParams)

美顔、美白、肌の色調補正効果のランクを設定します。
パラメータ：

パラメータ名	タイプ	説明
beautyStyle	number	1はすべすべのつや感。綺麗な女性のショーに適用します。効果はかなりはっきりと出ます。 2はナチュラル。肌質補正アルゴリズムで、顔の細部をもっと多く残しますので、主観的によりナチュラルな感じを受けます。
beauty	number	美顔レベル。数値の範囲は0～9です。0はオフを意味し、1～9まで数値が大きくなるほど効果が高くなります
white	number	美白レベル。数値の範囲は0～9です。0はオフを意味し、1～9まで数値が大きくなるほど効果が高くなります
ruddiness	number	肌色補正レベル。数値の範囲は0～9です。0はオフを意味し、1～9まで数値が大きくなるほど効果が高くなります。このパラメータは、Windowsプラッ

©2013-2019 Tencent Cloud. All rights reserved.

		トフォームではまだ有効ではありません

例示：

```
interface BeautyParams {
beautyStyle: number;//スムース、ナチュラル
beauty: number; //美顔レベル
white: number; //美白レベル
ruddiness: number; //肌色補正レベル
}
rtcClient.setBeautyStyle({
beautyStyle: 1,
beauty: 5,
white: 5,
ruddiness: 5
})
```

コンポーネントイベント

例示

```
const EVENT = rtcClient.EVENT
rtcClient.on(EVENT.STUDENT_RAISE_HAND, () => {
//学生の挙手
})
```

詳細イベント

CODE	説明
ENTER_ROOM_SUCCESS	入室の成功
LEAVE_ROOM_SUCCESS	退室の成功
TEACHER_ENTER	教師の入室
TEACHER_LEAVE	教師の退室
STUDENT_ENTER	学生の入室
STUDENT_LEAVE	学生の退室
SCREEN_SHARE_ADD	教師の画面共有開始

©2013-2019 Tencent Cloud. All rights reserved.

SCREEN_SHARE_REMOVE	教師の画面共有終了
REMOTE_VIDEO_ADD	リモートビデオストリームのイベントの追加。リモートユーザーがビデオストリームを公開すると、この通知を受け取ります。
REMOTE_VIDEO_REMOVE	リモートビデオストリームのイベントの削除。リモートユーザーがビデオストリームの公開をキャンセルすると、この通知を受け取ります。
REMOTE_AUDIO_ADD	リモートオーディオストリームのイベントの追加
REMOTE_AUDIO_REMOVE	リモートオーディオストリームのイベントの削除
ROOM_DESTROYED	ルームの廃棄
QUESTION_TIME_STARTED	QA時間の開始
QUESTION_TIME_STOPPED	QA時間の終了
STUDENT_RAISE_HAND	学生の挙手
BE_INVITED_TO_PLATFORM	招待されて質問に回答、指名
ANSWERING_FINISHED	回答の終了、音声禁止
MESSAGE_RECEIVED	メッセージの受信
KICKED_OUT	同じアカウントが他の場所でログインすると、追放されてオフラインになります。
ERROR	異常
WARNING	警告

©2013-2019 Tencent Cloud. All rights reserved.

オンラインカラオケ_KTV
オンラインカラオケ_KTV（iOS）

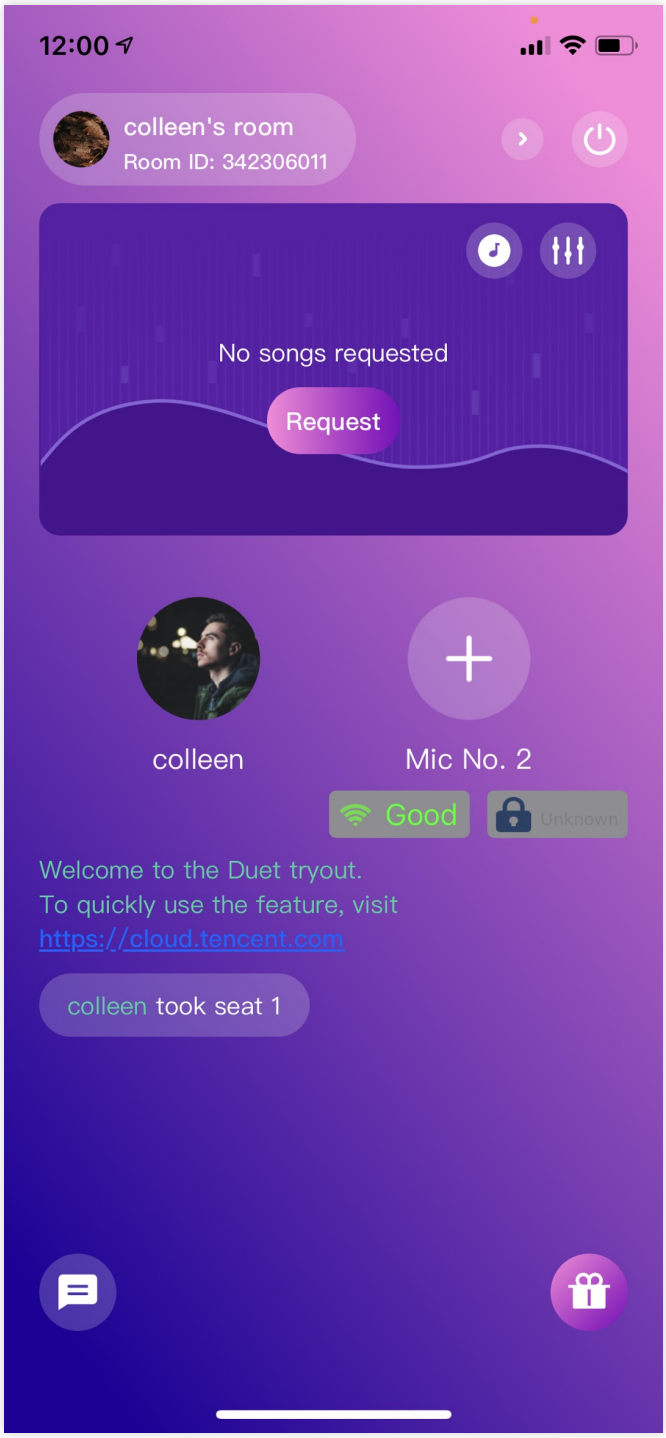
最終更新日：：2022-03-09 17:13:47

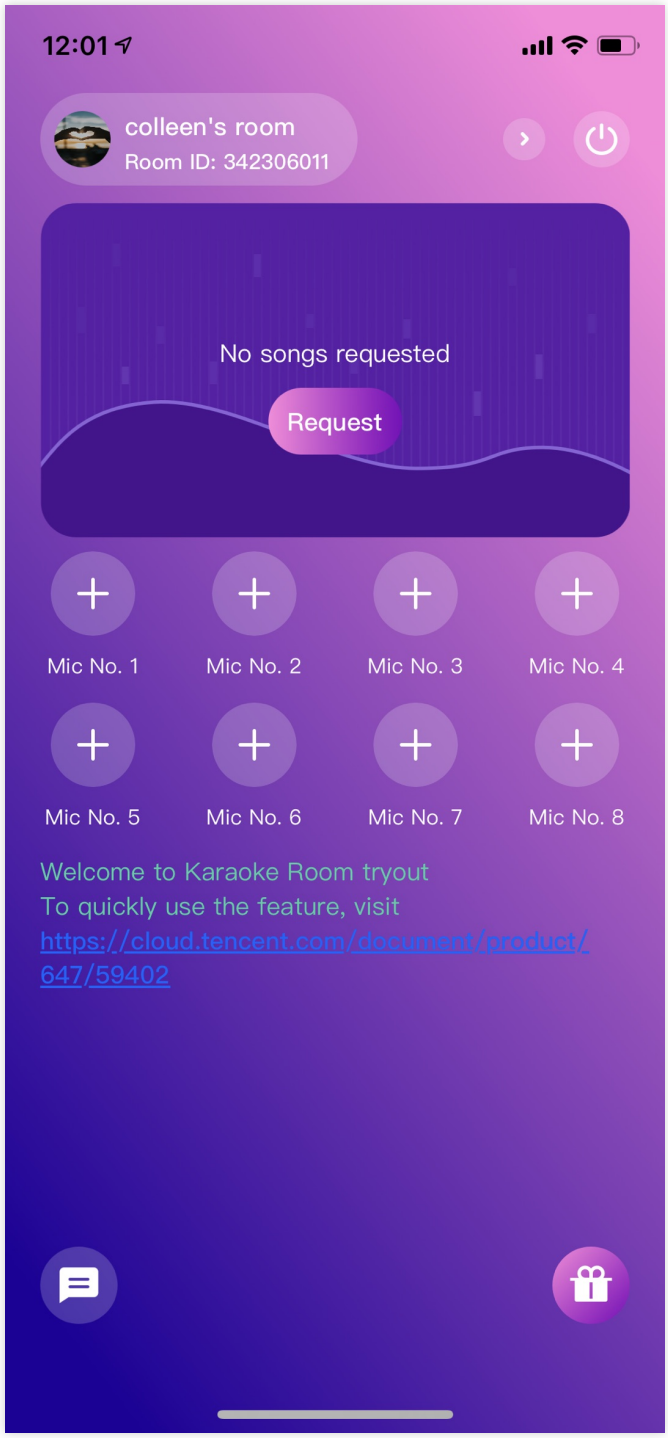
デモンストレーション

ダウンロードからAppをインストールして、低遅延カラオケ、マイク管理、ギフトの送受信、文字チャットなどのKTVのユースケースにおけるTRTCの関連機能を含む、KTVの機能を体験できます。

管理者によるマイク操作	リスナーによるマイク操作

©2013-2019 Tencent Cloud. All rights reserved.





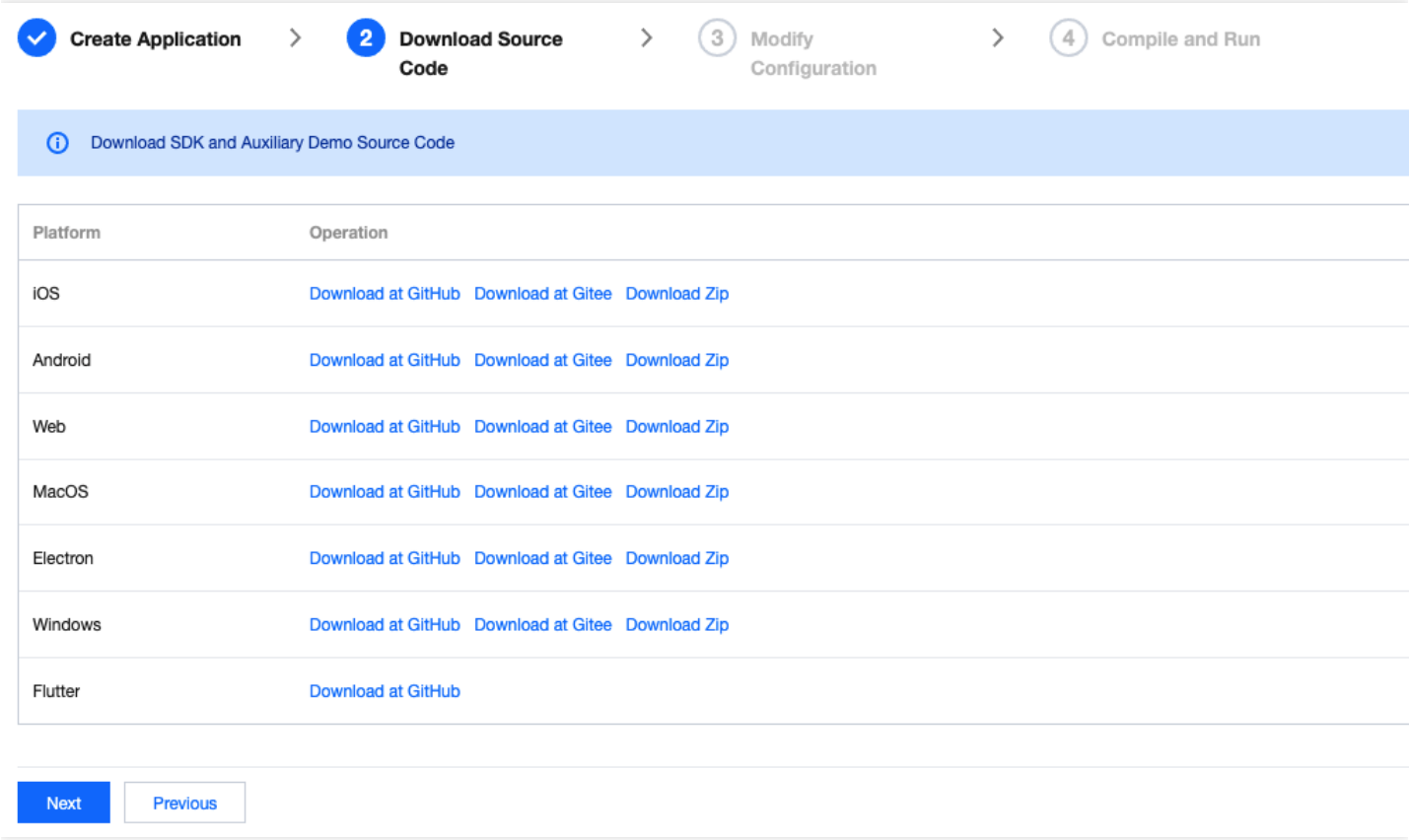
KTV機能にすばやくアクセスする必要がある場合、当社が提供するAppをもとに直接変更を加えて適応させるか、または当社が提供するTUIKaraokeコンポーネントでカスタマイズしたUIを実装することができます。

AppのUIの再利用

手順1：アプリケーションの新規作成

©2013-2019 Tencent Cloud. All rights reserved.

1. TRTCコンソールにログインし、【開発支援】>【快速跑通Demo】を選択します。
2. `TestKaraoke` など、アプリケーション名を入力し、【作成】をクリックします。
3. 【ダウンロードしました。次のステップ】をクリックすると、この手順をスキップします。



説明：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

クリックしてTUIKaraokeに進み、ソースコードをCloneするか、またはダウンロードします。

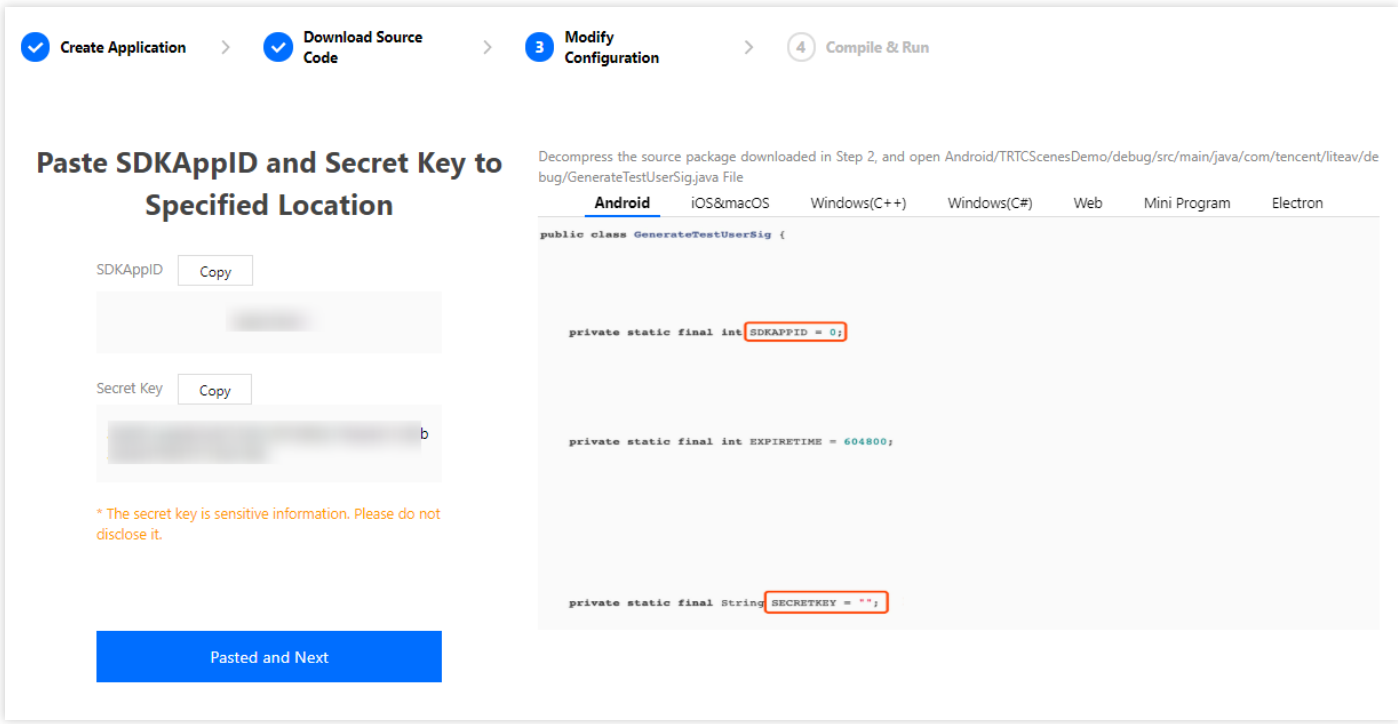
手順3：Appプロジェクトファイルの設定

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `TUIKaraoke/Debug/GenerateTestUserSig.swift` のファイルを見つけて開きます。

©2013-2019 Tencent Cloud. All rights reserved.

3. `GenerateTestUserSig.swift` ファイル内の関連パラメータを設定します。
   - SDKAPPID：デフォルトは0。実際のSDKAppIDを設定してください。
   - SECRETKEY：デフォルトは空文字列。実際のキー情報を設定してください。



4. 貼り付け完了後、【貼り付けました。次へ】をクリックすれば、作成が完了します。

5. コンパイル完了後、【コンソールの概要ページに戻る】をクリックすればOKです。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

©2013-2019 Tencent Cloud. All rights reserved.

Xcode（11.0およびそれ以降のバージョン）を使用してソースコード `TUIKaraoke/TUIKaraokeApp.xcworkspace` プロジェクトを開き、【実行】をクリックすれば、このAppのデバッグが開始されます。

手順5：Appソースコードの修正

ソースコードの `Source` フォルダは、uiとmodelの2つのサブフォルダを含んでいます。uiフォルダには、インターフェースコードとインターフェース関連のロジックが含まれています。以下の表にはそれぞれのswiftファイルまたはフォルダおよびそれらが対応するUIをリストアップして、二次調整を行いやすくしています。

ファイルまたはフォルダ	機能説明
TRTCKaraokeEnteryControl.swift	このファイルには、すべてのViewController初期化メソッドが含まれています。このインスタンスを介してViewControllerオブジェクトをすばやく取得することができます。
TRTCCreateKaraokeViewController	KTVページのロジックを作成します。
TRTCKaraokeViewController	メインルームページは、管理者とリスナーという2種類のインターフェースがあります。

各 `TRTC'XXXX'ViewController` フォルダには、 `ViewController` 、 `RootView` および `ViewModel` が含まれ、各ファイルの役割は下表に示すとおりです。

ファイル	機能説明
ViewController.swift	ページコントローラ。画面のルーティング作業、RootViewおよびViewModelのバインディング作業を担当します。
RootView.swift	ビュー。すべてのビューのレイアウト。
ViewModel.swift	ビューコントローラ。ビューのインタラクションに応答し、ビューの応答ステータスを返す作業を担当します。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

©2013-2019 Tencent Cloud. All rights reserved.

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. 【ルームの作成】をクリックします。
3. ルームトピックを入力し、【一緒に歌う】をクリックします。

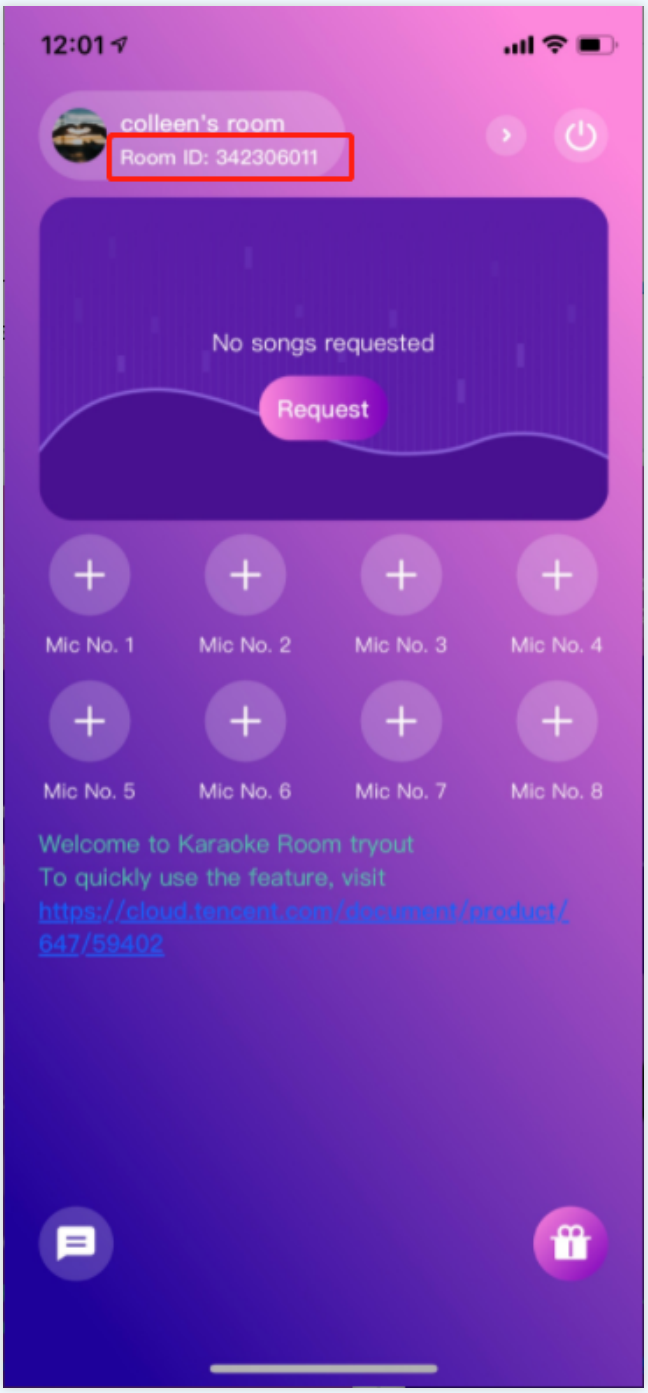
ユーザーB

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ユーザーAが作成したルーム番号を入力し、【入室する】をクリックします。

注意：

©2013-2019 Tencent Cloud. All rights reserved.

下図のように、ルーム番号はユーザーAのルーム上部に表示されます。

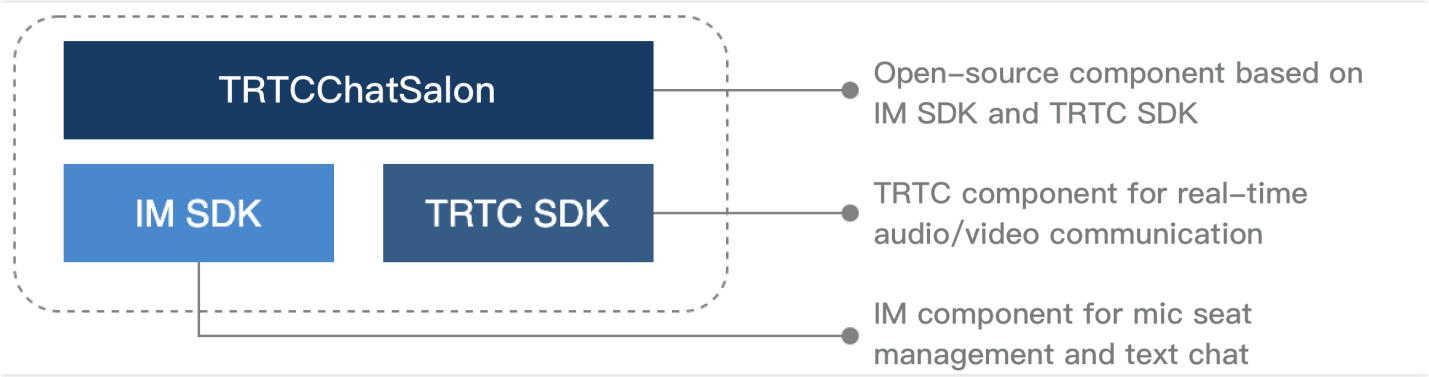


UIカスタマイズの実装

ソースコードの `Source` フォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTRTCKaraokeRoomがあります。 `TRTCKaraokeRoom.h` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタムUIを実装

©2013-2019 Tencent Cloud. All rights reserved.

することができます。



手順1：SDKの統合

KTVコンポーネントTRTCKaraokeRoomは、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

- **方法1：cocoapodsリポジトリを介する依存**

```
pod 'TXIMSDK_iOS'
pod 'TXLiteAVSDK_TRTC'
```

- **方法2：ローカルを介する依存**
  開発環境でのcocoapodsリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド
TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定

info.plistファイルにおいて、`Privacy > Microphone Usage Description` を追加して、マイクの権限を申請する必要があります。

手順3：TUIKaraokeコンポーネントのインポート

cocoapodsを介してコンポーネントをインポートします。具体的な操作は次のとおりです。

1. プロジェクトディレクトリ下の `Source` 、 `Resources` 、 `TXAppBasic` フォルダ、 `TUIKaraoke.podspec` ファイルをプロジェクトディレクトリ下にコピーします。

©2013-2019 Tencent Cloud. All rights reserved.

2. `Podfile` ファイル内に以下の依存関係を追加します。その後、 `pod install` コマンドを実行すると、インポートが完了します。

```
pod 'TXAppBasic', :path =&gt; "TXAppBasic/"
pod 'TXLiteAVSDK_TRTC'
pod 'TUIKaraoke', :path =&gt; "./", :subspecs =&gt; ["TRTC"]
```

手順4：コンポーネントの作成およびログイン

1. TRTCKaraokeRoomの `sharedInstance` クラスメソッドを呼び出すと、TRTCKaraokeRoomプロトコルに準拠したインスタンスオブジェクトを作成することができます。また、 `shared` クラスメソッドを呼び出して、TRTCKaraokeRoomインスタンスオブジェクトを取得し、直接使用することもできます。両者は、TRTCKaraokeRoomのインターフェースでの使用に違いはありません。
2. `setDelegate` 関数を呼び出して、コンポーネントのイベントコールバック通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	機能
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
callback	ログインのコールバック。成功時にcodeは0になります。

```
// Swift例
// コード内でビジネスロジックを担当するクラス
class YourController {
// 属性を計算してシングルトンオブジェクトを取得
var karaokeRoom: TRTCKaraokeRoom {
return TRTCKaraokeRoom.shared()
}

// その他のコードロジック
......
}
// KTVプロキシを設定
self.karaokeRoom.setDelegate(delegate: karaokeRoomDelegate)
```

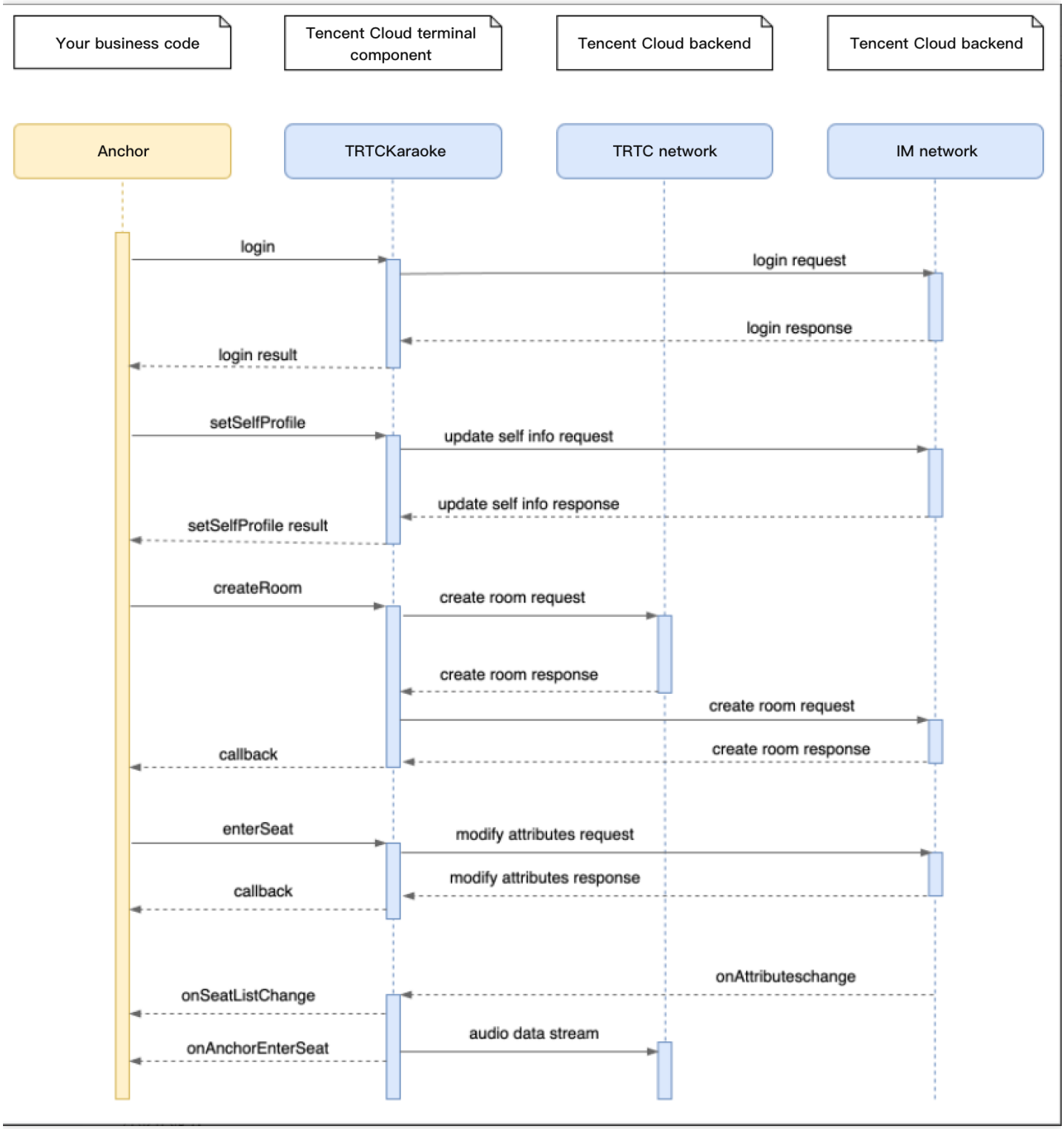
©2013-2019 Tencent Cloud. All rights reserved.

```
// コール方法は以下のとおり。クロージャーの中には weak self を使用して循環引用にしないことをお勧めします（以下の例示コードでは、 weak self 例を省略）
self.karaokeRoom.login(sdkAppId: sdkAppID, userId: userId, userSig: userSig) { [weak self] (code, message) in
guard let `self` = self else { return }
// コールバックビジネスロジック
}
```

手順5：管理者によるルームの作成

1. 管理者は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 管理者は、 `createRoom` を呼び出して新しいKTVルームを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否、ルームタイプなどルームの属性情報を渡します。
3. 管理者はルームの作成に成功した後、 `enterSeat` を呼び出して参加することができます。
4. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
5. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

©2013-2019 Tencent Cloud. All rights reserved.



サンプルコード：

```
// 1.管理者は、ニックネームおよびプロフィール画像を設定します
self.karaokeRoom.setSelfProfile(userName: userName, avatarUrl: avatarURL) { (code, message) in
// 結果のコールバック
}
```

```
// 2.管理者側でルームを作成します
let param = RoomParam.init()
Param.roomName = "ルーム名"
param.needRequest = true // リスナーのマイク・オンに対する管理者の同意の要否
param.coverUrl = "カバーURL"
param.seatCount = 8 // ルームの座席数。ここは計8 席あります
param.seatInfoList = []

for _ in 0..&lt; param.seatCount {
let seatInfo = SeatInfo.init()
param.seatInfoList.append(seatInfo)
}

self.karaokeRoom.createRoom(roomID: yourRoomID, roomParam: param) { (code, message) in
guard code == 0 else { reutrn }
// 作成に成功

// 3.マイク・オン
self.karaokeRoom.enterSeat(seatIndex: 0) { [weak self] (code, message) in
guard let `self` = self else { return }
if code == 0 {
// 管理者による席の占有成功
}else{
// 管理者による席の占有失敗
}
}
}

// 4.マイク・オンに成功した後、onSeatListChangeイベント通知を受信します
func onSeatListChange(seatInfoList: [SeatInfo]) {
// マイクリストの更新
}

// 5. onAnchorEnterSeatイベント通知を受信
func onAnchorEnterSeat(index: Int, user: UserInfo) {
// 管理者のマイク・オンイベントの処理
}
```

手順6：リスナー側の視聴

1. リスナー側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. リスナー側は、業務バックエンドから最新のKTVのルームリストを取得します。

> 説明：

©2013-2019 Tencent Cloud. All rights reserved.

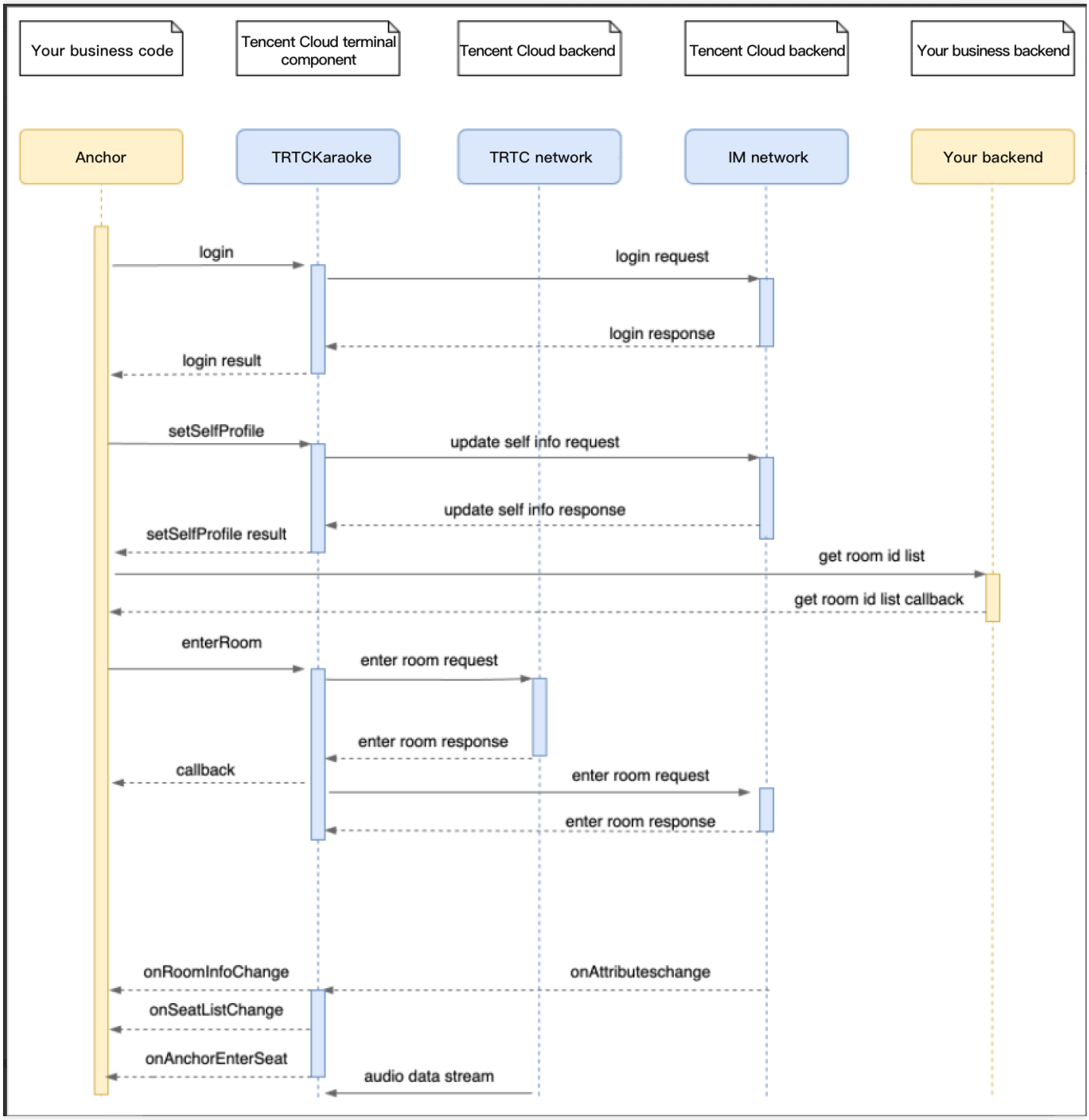
App内のKTVルームリストはデモに使用するためだけのものです。KTVルームリストの業務ロジックは千差万別です。現在、Tencent CloudはKTVルームリスト管理のサービスを提供していません。各自でご自分のKTVルームリストを管理してください。

3. リスナー側は、`getRoomInfoList` を呼び出してルームの詳細情報を取得します。この情報は、管理者側が `createRoom` を呼び出してKTVルームを作成するときに設定する簡単な説明情報です。

注意：
KTVルームリストに十分に包括的な情報がある場合は、`getRoomInfoList` の呼び出しに関する手順をスキップできます。

4. リスナーは1つのKTVルームを選択し、`enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
5. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
6. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
7. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` というイベント通知も受信します。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.リスナーは、ニックネームおよびプロフィール画像を設定します
self.karaokeRoom.setSelfProfile(userName: userName, avatarUrl: avatarURL) { (code, message) in
// 結果のコールバック
}

// 2.業務バックエンドから取得したルームリストをroomListと仮定します
let roomList: [Int] = getRoomIDList() // ルームIDリストを取得する関数
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// 3. getRoomInfoListを呼び出すことによって、ルームの詳細情報を取得します
self.karaokeRoom.getRoomInfoList(roomIdList: roomIdsInt) { (code, message, roomInfos: [RoomInfo])
in
//結果の取得。この時、UIの更新が可能
}

// 4.KTVを選択後、roomIdを渡して入室します
self.karaokeRoom.enterRoom(roomID: roomInfo.roomID) { (code, message) in
// 入室結果コールバック
if code == 0 {
// 入室に成功
}
}

// 5.入室に成功後、onRoomInfoChangeイベント通知を受信します
func onRoomInfoChange(roomInfo: RoomInfo) {
// ルーム名などの情報の更新可
}

// 6.入室に成功後、onSeatListChangeイベント通知を受信します
func onSeatListChange(seatInfoList: [SeatInfo]) {
// マイクリストを更新
}

// 7. onAnchorEnterSeatイベント通知を受信します
func onAnchorEnterSeat(index: Int, user: UserInfo) {
// マイク・オンイベントの処理
}
```

手順7：マイク管理

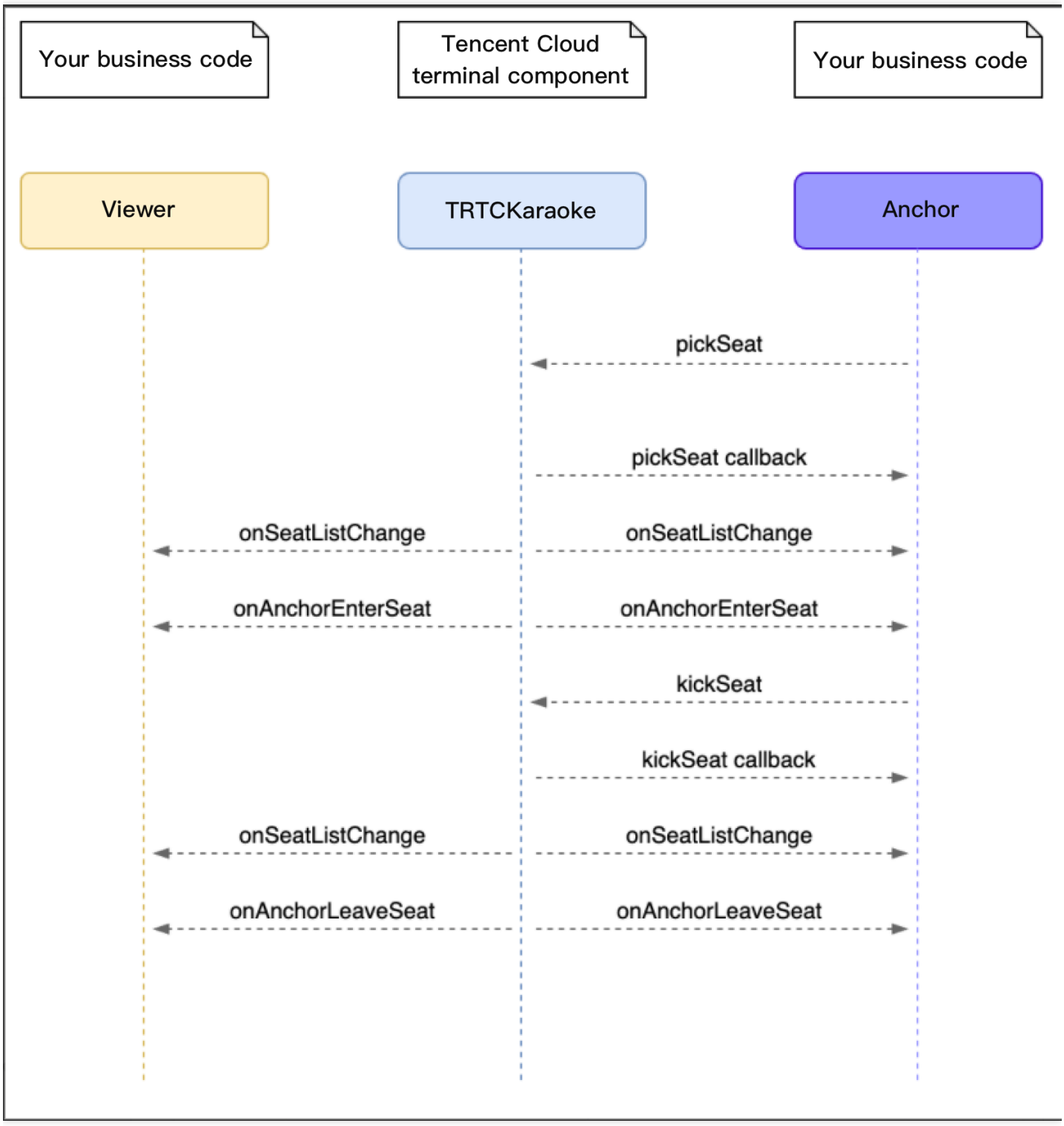
- 管理者側
- リスナー側

1. `pickSeat` は、対応するマイクおよびリスナーのuserIdを渡し、視聴者を発言できるように招待できます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。
2. `kickSeat` は、対象となるマイクを渡した後、キックアウトしてマイク・オフにすることができます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。
3. `muteSeat` は、対応するマイクを渡した後、ミュート/ミュート解除をすることができます。ルーム内の全参加者は `onSeatListChange` および `onSeatMute` というイベント通知を受信します。

4. `closeSeat` は、対応するマイクを送信後、任意のマイクのクローズ/解除をすることができます。クローズ後は、リスナー側はこれ以上マイクを使用することはできません。ルーム内の全参加者は `onSeatListChange` およ

©2013-2019 Tencent Cloud. All rights reserved.

び `onSeatClose` というイベント通知を受信します。



手順8：招待シグナリングの使用

マイク管理では、リスナーがマイク・オン/オフにする場合や、管理者が視聴者を発言できるように招待する場合は、相手の同意を必要とせずに直接操作することができます。

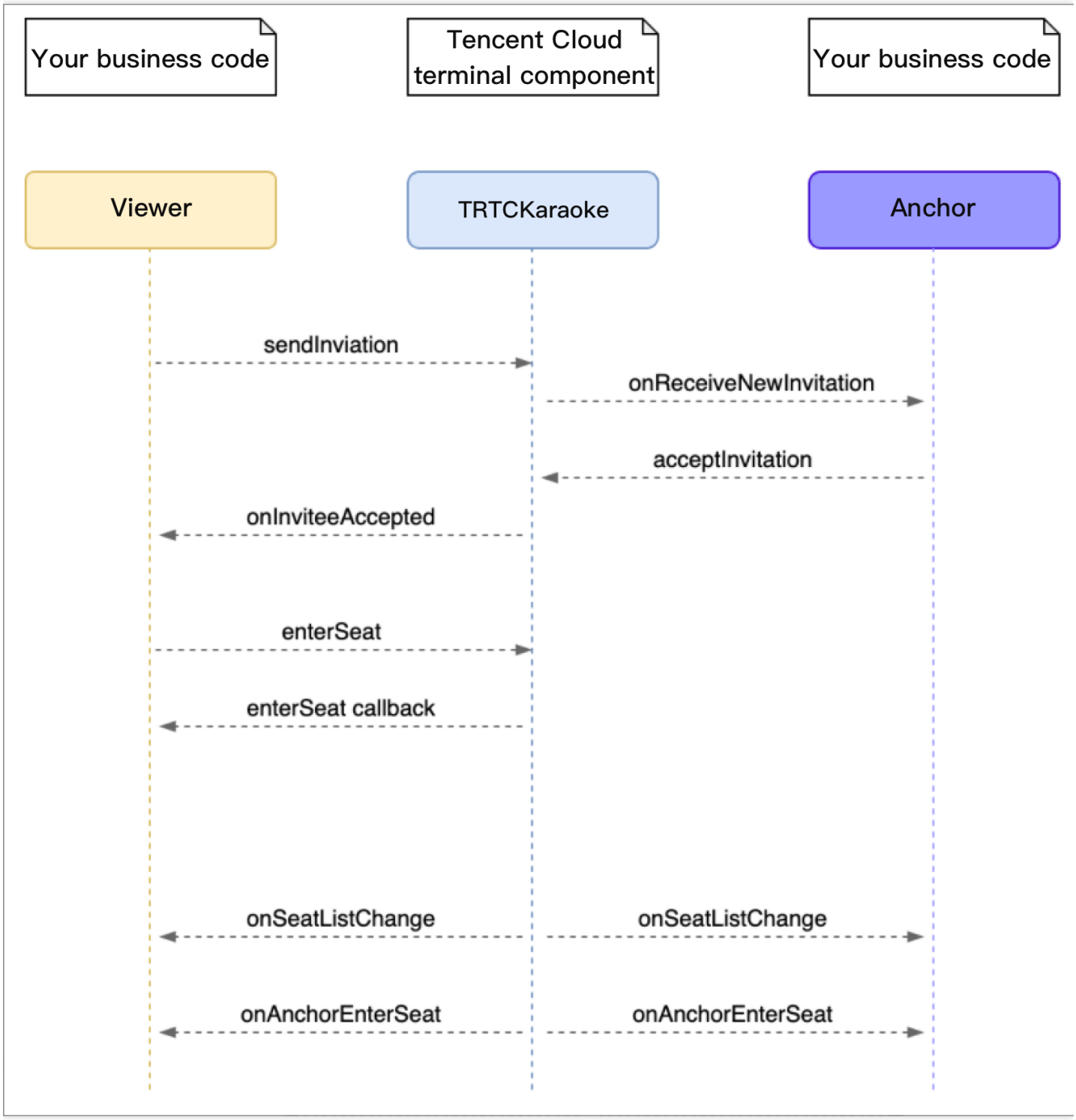
©2013-2019 Tencent Cloud. All rights reserved.

Appにおいて相手の同意がなければ、次の操作の業務フローを実施できない場合は、招待シグナリングによって相応のサポートを行うことができます。

- リスナーからの自主的なマイク・オン申請
- 管理者からの招待によるリスナーのマイク・オン

1. リスナー側が `sendInvitation` を呼び出し、管理者のuserIdおよび業務のカスタムコマンドワードなどを渡します。この時、関数が1つのinviteIdを返しますので、このinviteIdを記録します。
2. 管理者側は、 `onReceiveNewInvitation` というイベント通知を受信します。この時UIでウィンドウをポップアップさせ、管理者に同意の有無をたずねることができます。
3. 管理者が同意を選択後、 `acceptInvitation` を呼び出してinviteIdを渡します。
4. リスナー側は、 `onInviteeAccepted` というイベント通知を受信し、 `enterSeat` を呼び出してマイク・オンにします。

©2013-2019 Tencent Cloud. All rights reserved.




```
// リスナー側の視点
// 1. sendInvitationを呼び出し、1号マイクの使用をリクエストします
let inviteId = self.karaokeRoom.sendInvitation(cmd: "ENTER_SEAT", userId: ownerUserId, content:
"1") { (code, message) in
// 発信結果のコールバック
}
// 2.招待のリクエスト同意を受信し、正式にマイク・オンになります
```

©2013-2019 Tencent Cloud. All rights reserved.

```
func onInviteeAccepted(identifier: String, invitee: String) {
if identifier == selfID {
self.karaokeRoom.enterSeat(seatIndex: ) { (code, message) in
// マイク・オン結果のコールバック
}
}
}

// 管理者側の視点
// 1.管理者がリクエストを受信します
func onReceiveNewInvitation(identifier: String, inviter: String, cmd: String, content: String) {
if cmd == "ENTER_SEAT" {
// 2.管理者がリスナーのリクエストに同意します
self.karaokeRoom.acceptInvitation(identifier: identifier, callback: nil)
}
}
```

手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、このルーム内のすべてのキャスターおよびリスナーが `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

```
// 発信側：テキストメッセージの発信
self.karaokeRoom.sendRoomTextMsg(message: message) { (code, message) in


}
// 受信側：テキストメッセージのモニタリング
func onRecvRoomTextMsg(message: String, userInfo: UserInfo) {
// 受信したmessage情報の処理方法
}
```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターとリスナーが `onRecvRoomCustomMsg` コールバックを受信できます。
  カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

```
// 例：発信側：カスタマイズCmdによって、弾幕と「いいね」情報を区分することができます
// eg:「CMD_DANMU」は弾幕コメントを表し、「CMD_LIKE」は「いいね」情報を表します
self.karaokeRoom.sendRoomCustomMsg(cmd: "CMD_DANMU", message: "hello world", callback: nil)
self.karaokeRoom.sendRoomCustomMsg(cmd: "CMD_LIKE", message: "", callback: nil)
// 受信側：カスタムメッセージのモニタリング
func onRecvRoomCustomMsg(cmd: String, message: String, userInfo: UserInfo) {
if cmd == "CMD_DANMU" {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
// 弾幕コメントの受信
}
if cmd == "CMD_LIKE" {
// 「いいね」情報の受信
}
}
```

©2013-2019 Tencent Cloud. All rights reserved.

オンラインカラオケ_KTV（Android）

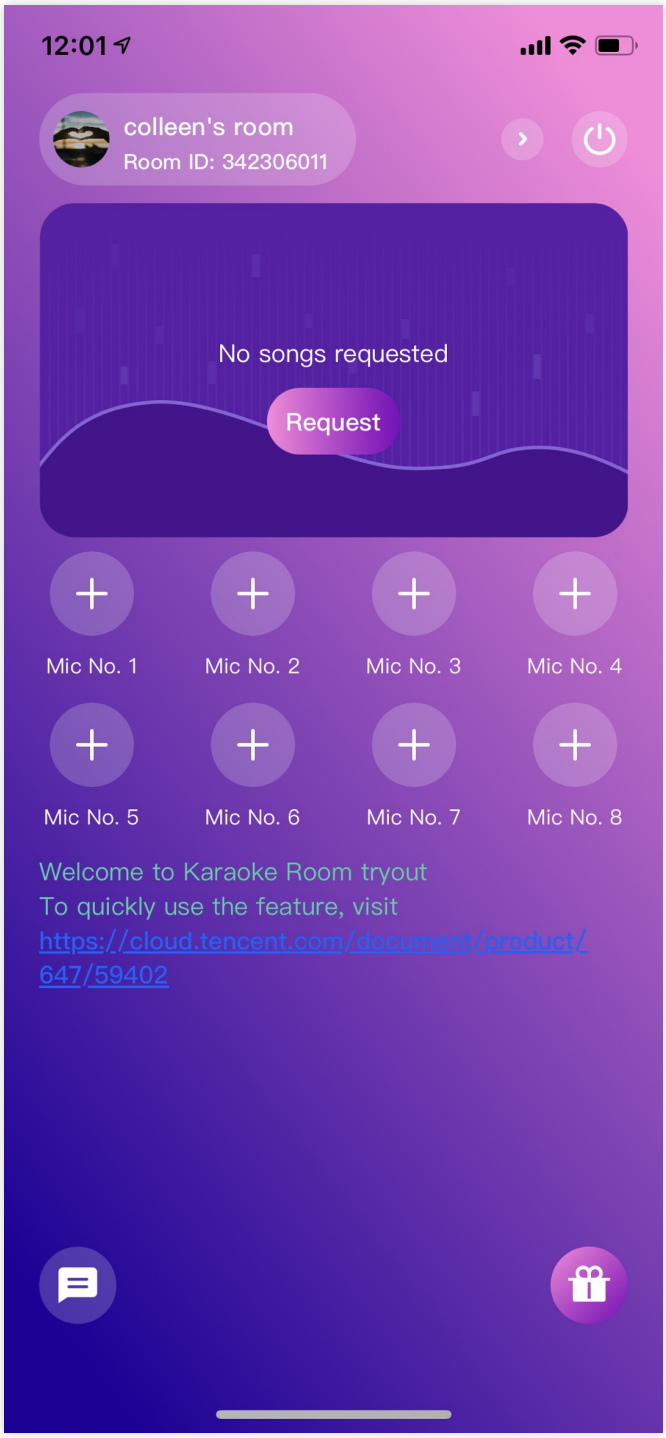
最終更新日：：2022-03-09 17:19:45

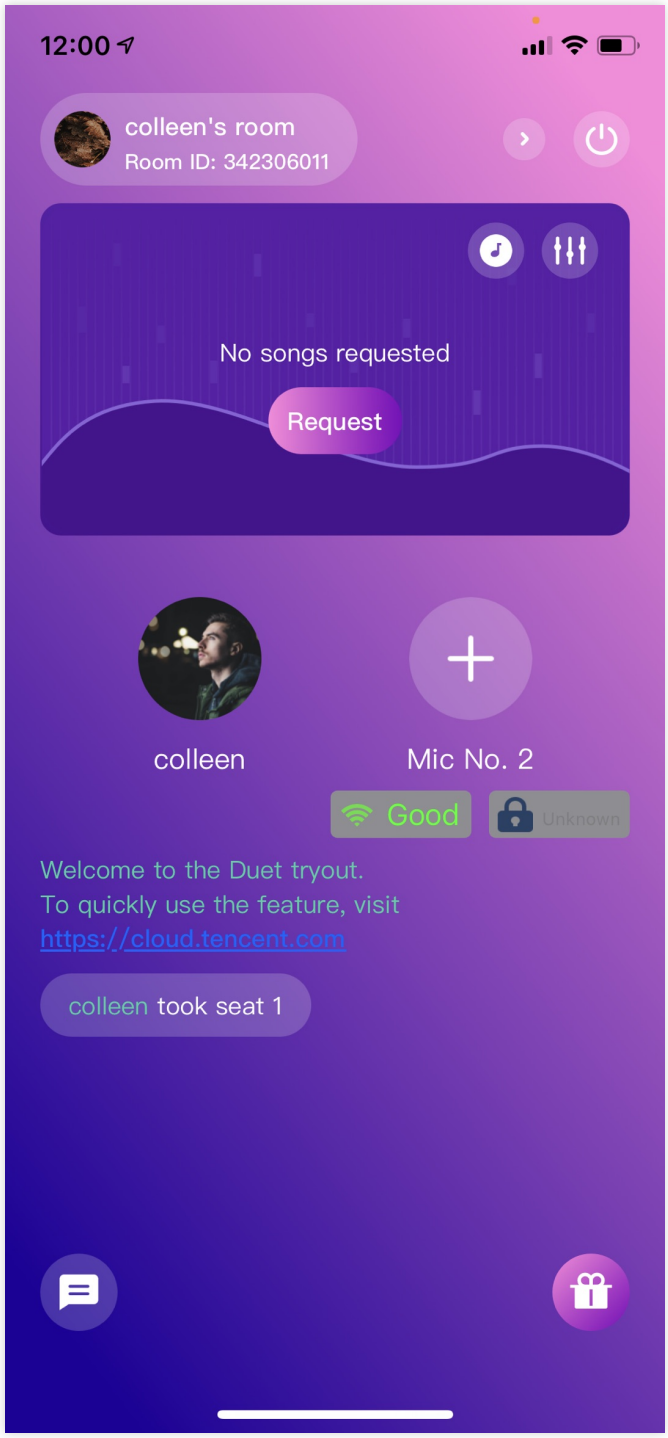
デモンストレーション

ダウンロードからAppをインストールして、低遅延カラオケ、マイク管理、ギフトの送受信、文字チャットなどのKTVのユースケースにおけるTRTCの関連機能を含む、KTVの機能を体験できます。

管理者によるマイク操作	リスナーによるマイク操作

©2013-2019 Tencent Cloud. All rights reserved.





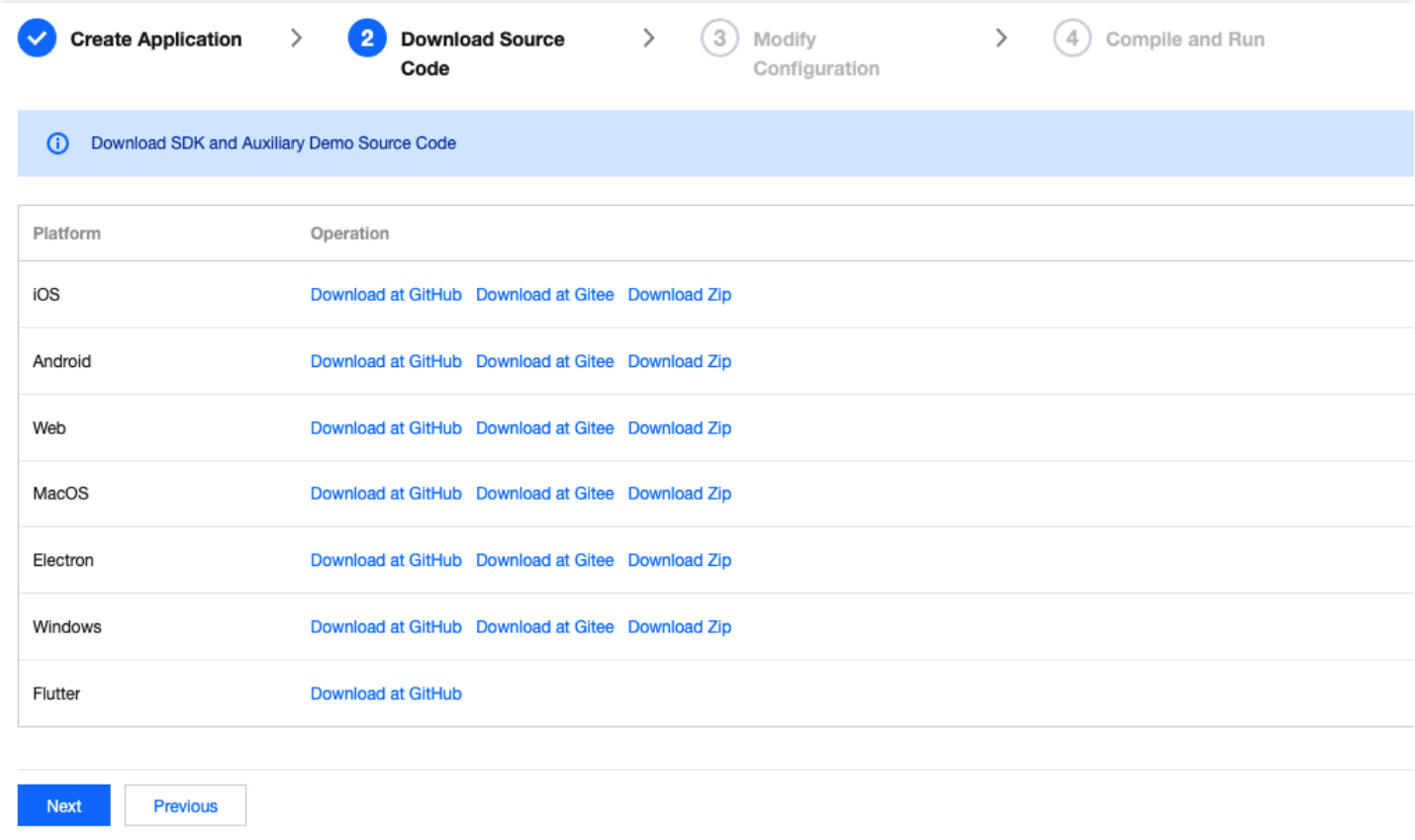
KTV機能にすばやくアクセスする必要がある場合、当社が提供するAppをもとに直接変更を加えて適応させるか、または当社が提供するTUIKaraokeコンポーネントでカスタマイズしたUIを実装することができます。

AppのUIの再利用

手順1：アプリケーションの新規作成

©2013-2019 Tencent Cloud. All rights reserved.

1. TRTCコンソールにログインし、【開発支援】>【快速跑通Demo】を選択します。
2. `TestKaraoke` など、アプリケーション名を入力し、【作成】をクリックします。
3. 【ダウンロードしました。次のステップ】をクリックすると、この手順をスキップします。



注意：
本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

手順2：Appソースコードのダウンロード

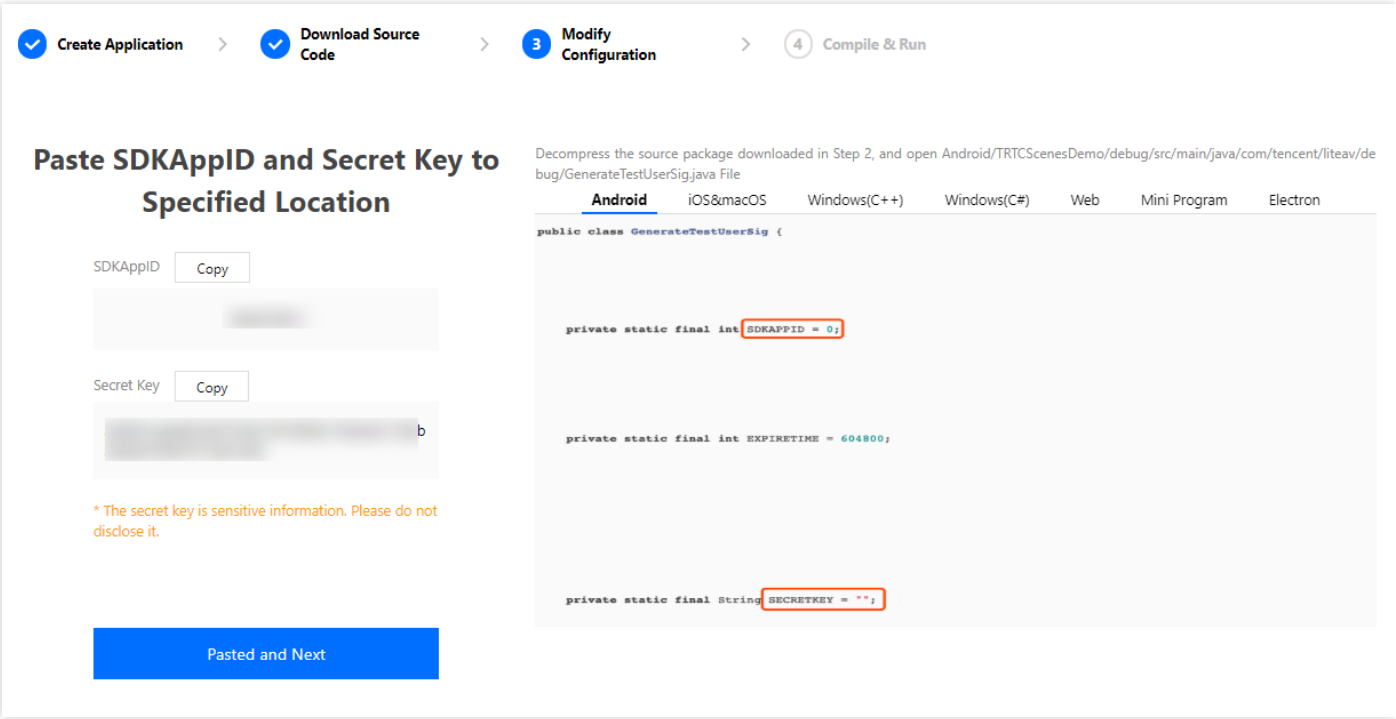
クリックしてTUIKaraokeに進み、ソースコードをCloneするか、またはダウンロードします。

手順3：Appプロジェクトファイルの設定

1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。
2. `Android/Debug/src/main/java/com/tencent/liteav/debug/GenerateTestUserSig.java` のファイルを見つけて開きます。
3. `GenerateTestUserSig.java` ファイル内の関連パラメータを設定します。

©2013-2019 Tencent Cloud. All rights reserved.

- SDKAPPID：デフォルトはプレースホルダー(PLACEHOLDER)。実際のSDKAppIDを設定してください。
- SECRETKEY：デフォルトはプレースホルダー(PLACEHOLDER)。実際のキー情報を設定してください。



4. 貼り付け完了後、【貼り付けました。次へ】をクリックすれば、作成が完了します。
5. コンパイル完了後、【コンソールの概要ページに戻る】をクリックすればOKです。

注意：

- ここで言及するUserSigの発行方法は、クライアントコードの中でのSECRETKEY設定となりますが、この手法のSECRETKEYは逆コンパイルによって逆クラッキングされやすく、キーがいったん漏洩すると、攻撃者がお客様のTencent Cloudトラフィックを盗用できるようになります。そのため**この手法は、ローカルのAppクイックスタートおよび機能デバッグにのみ適しています**。
- UserSigの正しい発行方法は、UserSigの計算コードをサーバーに統合し、Appのインターフェース向けに提供します。UserSigが必要なときは、Appから業務サーバーにリクエストを発出し動的にUserSigを取得します。詳細はUserSigに関するご質問をご参照ください。

手順4：Appの実行

Android Studio（バージョン3.5以降）を使用してソースコード `TUIKaraoke` プロジェクトを開き、【実行】をクリックすれば、このAppのデバッグが開始されます。

手順5：Appソースコードの修正

©2013-2019 Tencent Cloud. All rights reserved.

ソースコードのSourceフォルダは、uiとmodelという2つのサブフォルダを含んでいます。uiフォルダにはインターフェースコードが含まれています。以下の表にファイルまたはフォルダおよび対応するUIをリストアップして、二次調整を行いやすくしています。

ファイルまたはフォルダ	機能説明
base	UIに使用される基礎となるクラス。
room	メインルームページは、管理者とリスナーという2種類のインターフェースがあります。
widget	汎用ウィジェット。

体験アプリケーション

> 注意：
> 体験アプリケーションには、少なくとも2台のデバイスが必要です。

ユーザーA

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. 【ルームの作成】をクリックします。
3. ルームトピックを入力し、【一緒に歌う】をクリックします。

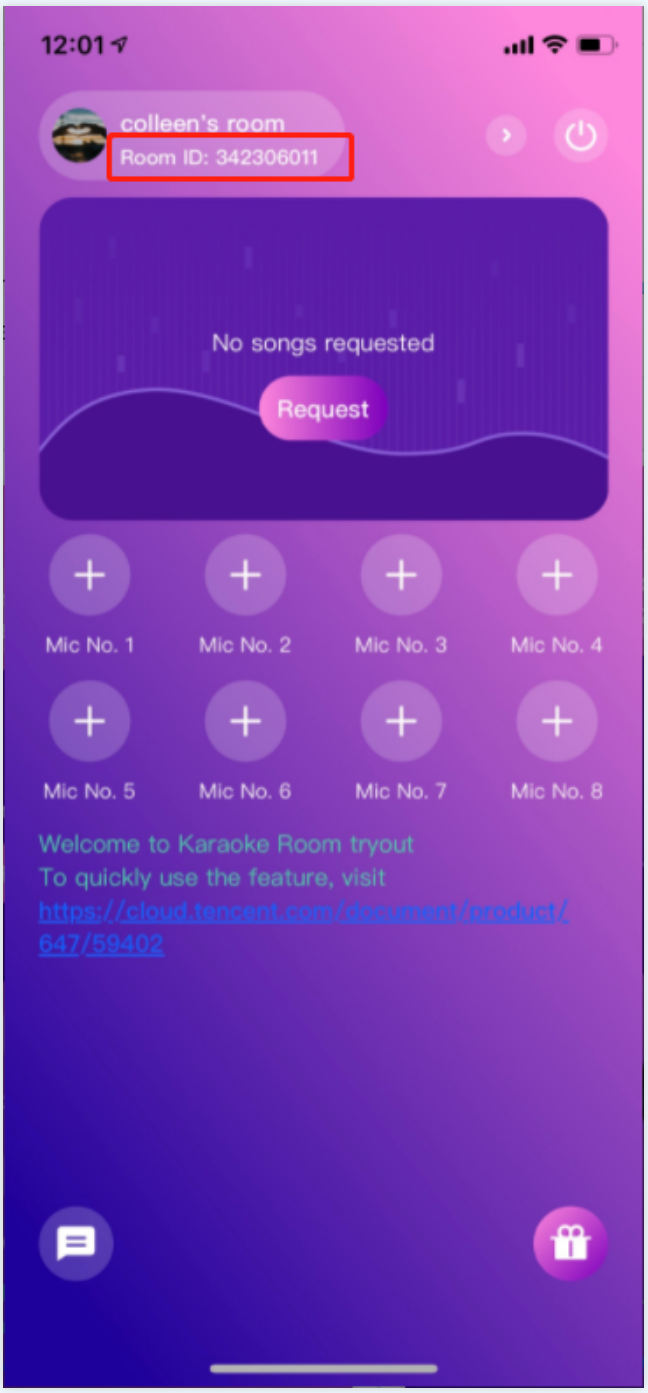
ユーザーB

1. ユーザー名を入力し（**ユーザー名は一意のものとし、他のユーザーと重複しないようにしてください**）、ログインします。
2. ユーザーAが作成したルーム番号を入力し、【入室する】をクリックします。

> 注意：

©2013-2019 Tencent Cloud. All rights reserved.

下図のように、ルーム番号はユーザーAのルーム上部に表示されます。

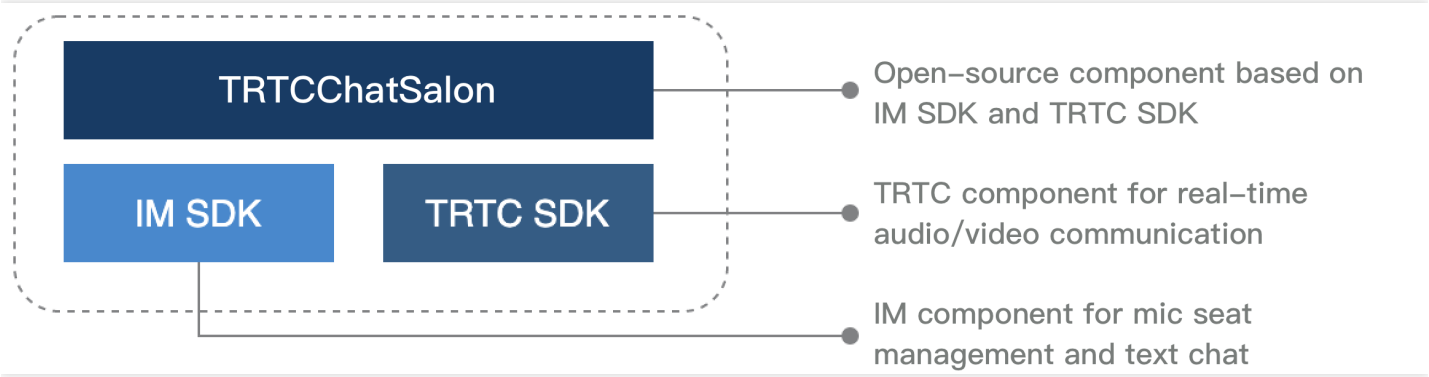


UIカスタマイズの実装

ソースコードのSourceフォルダには、uiとmodelという2つのサブフォルダがあり、modelフォルダには再利用できるオープンソースコンポーネントTRTCKaraokeRoomがあります。 `TRTCKaraokeRoom.java` ファイルでこのコンポーネントが提供するインターフェース関数を確認し、対応するインターフェースを使用してカスタマイズ

©2013-2019 Tencent Cloud. All rights reserved.

したUIを実装することができます。



手順1：SDKの統合

KTVコンポーネントTRTCKaraokeRoomは、TRTC SDKとIM SDKに依存し、次の手順で2つのSDKをプロジェクトに統合することができます。

方法1：Mavenリポジトリを介する依存

1. dependenciesにTRTCSDKとIMSDKの依存を追加します。

```
dependencies {
complie "com.tencent.liteav:LiteAVSDK_TRTC:latest.release"
complie 'com.tencent.imsdk:imsdk:latest.release'
compile 'com.google.code.gson:gson:2.3.1'
}
```

2. defaultConfigでAppが使用するCPUアーキテクチャを指定します。

```
defaultConfig {
ndk {
abiFilters "armeabi-v7a"
}
}
```

3. 【Sync Now】をクリックし、自動でSDKをダウンロードし、プロジェクトに統合します。

方法2：ローカルAARを介する依存

開発環境でのMavenリポジトリへのアクセスが遅い場合は、ZIPパッケージを直接ダウンロードし、統合ドキュメントに従って手動でプロジェクトに統合することができます。

SDK	ダウンロードページ	統合ガイド

©2013-2019 Tencent Cloud. All rights reserved.

TRTC SDK	DOWNLOAD	統合ドキュメント
IM SDK	DOWNLOAD	統合ドキュメント

手順2：権限の設定および難読化ルール

AndroidManifest.xmlにAppの権限を設定します。SDKには次の権限が必要です（6.0以降のAndroidシステムではストレージ読み取りの権限を動的に申請する必要があります）。

```
<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.READ_PHONE_STATE">
```

proguard-rules.proファイルでは、SDK関連を非難読化リストに追加します。

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

手順3：TRTCKaraokeコンポーネントのインポート

次のディレクトリ内のすべてのファイルをプロジェクトにコピーします。

```
Source/src/main/java/com/tencent/liteav/tuikaraoke/model
```

手順4：コンポーネントの作成およびログイン

1. `sharedInstance` インターフェースを呼び出すと、TRTCKaraokeコンポーネントのインスタンスオブジェクトを作成できます。
2. `setDelegate` 関数を呼び出してコンポーネントのイベント通知を登録します。
3. `login` 関数を呼び出してコンポーネントのログインを完了します。下表を参考にキーパラメータを入力してください。

パラメータ名	作用
sdkAppId	TRTCコンソール で SDKAppIDを表示できます。
userId	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-z、A-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。業務の実際のアカウントシステムと組み合わせてご自身で設定することをお勧めします。

©2013-2019 Tencent Cloud. All rights reserved.

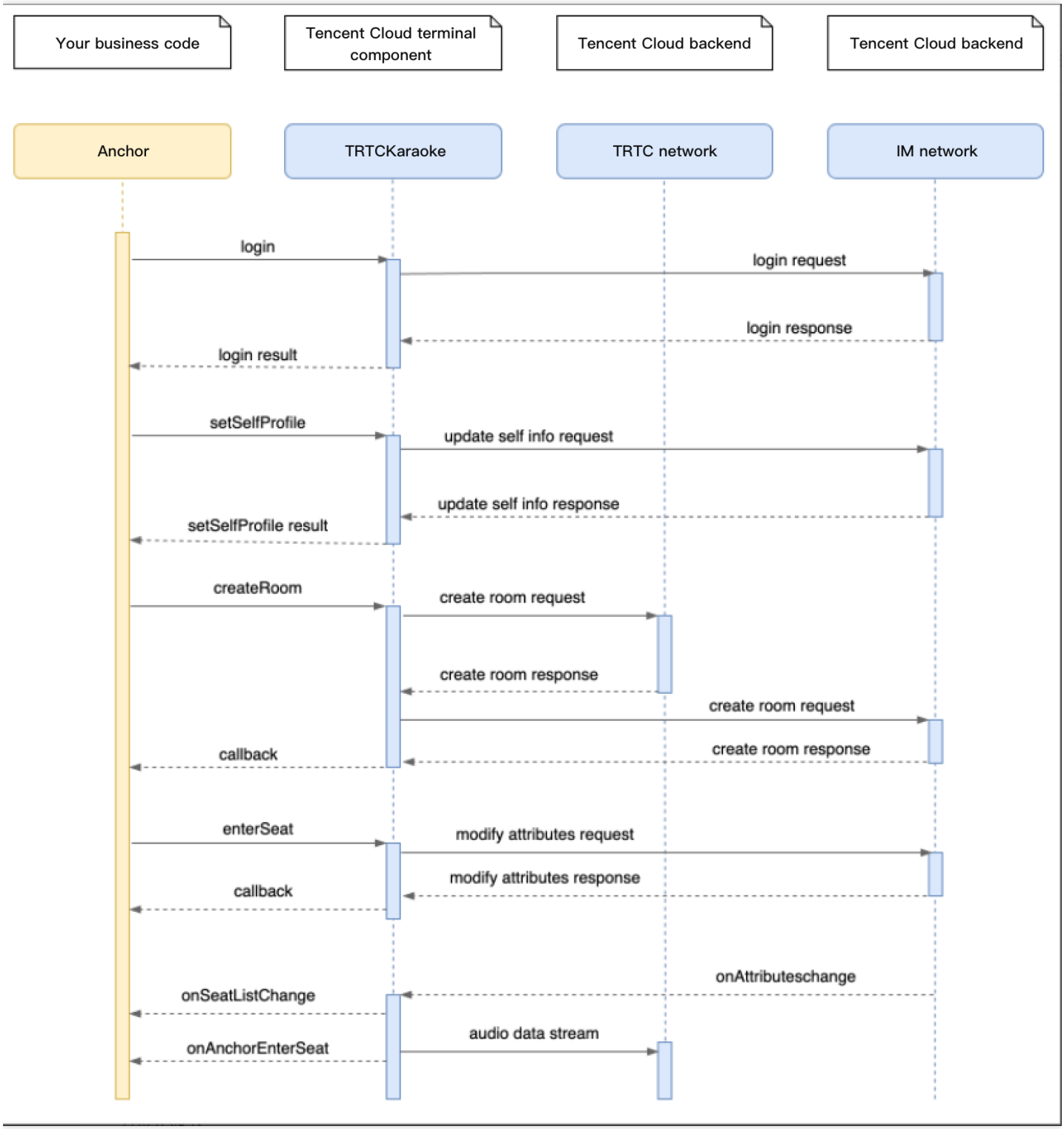
userSig	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、UserSigの計算方法をご参照ください。
callback	ログインのコールバック。成功時にcodeは0になります。

```
TRTCKaraoke mTRTCKaraokeRoom = TRTCKaraokeRoom.sharedInstance(this);
mTRTCKaraokeRoom.setDelegate(this);
mTRTCKaraokeRoom.login(SDKAPPID, userId, userSig, new TRTCKaraokeRoomCallback.ActionCallback()
{
@Override
public void onCallback(int code, String msg) {
if (code == 0) {![](https://main.qcloudimg.com/raw/7e61af0f3de8302260a1e10d7da6231e.png)
//ログイン成功
}
}
});
```

手順5：管理者側での配信開始

1. 管理者は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 管理者は、 `createRoom` を呼び出して新しいKTVルームを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否、ルームタイプなどルームの属性情報を渡します。
3. 管理者はルームの作成に成功した後、 `enterSeat` を呼び出して参加することができます。
4. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
5. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.管理者は、ニックネームおよびプロフィール画像を設定します
mTRTCKaraokeRoom.setSelfProfile("my_name", "my_face_url", null);

// 2.管理者は、createRoomを呼び出してルームを作成します
final TRTCKaraokeRoomDef.RoomParam roomParam = new TRTCKaraokeRoomDef.RoomParam();
roomParam.roomName = "ルーム名";
roomParam.needRequest = false; // マイク・オンに対する管理者の確認の要否
```

©2013-2019 Tencent Cloud. All rights reserved.

```
roomParam.seatCount = 7; // ルームの座席数。ここでは計7席あり、管理者が1席を占め、残り6席がリスナーとなります
roomParam.coverUrl = "ルームカバー図のURL ";
mTRTCKaraokeRoom.createRoom(mRoomId, roomParam, new TRTCKaraokeRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//3. 0番の席を占有します
mTRTCKaraokeRoom.enterSeat(0, new TRTCKaraokeRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
}
}
});
}
}
});

// 4.席の占有に成功した後、onSeatListChangeイベント通知を受信します
@Override
public void onSeatListChange(final List<trtckaraokeroomdef.seatinfo> seatInfoList) {
// マイクリストの表示
}

// 5. onAnchorEnterSeatイベント通知を受信
@Override
public void onAnchorEnterSeat(TRTCKaraokeRoomDef.UserInfo userInfo) {
}
```

手順6：リスナー側の視聴

1. リスナー側は、手順4でログイン後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. リスナー側は、業務バックエンドから最新のKTVのルームリストを取得します。

> 説明：
> App内のKTVルームリストはデモに使用するためだけのものです。KTVルームリストの業務ロジックは千差万別です。現在、Tencent CloudはKTVルームリスト管理のサービスを提供していません。各自でご自分のKTVルームリストを管理してください。

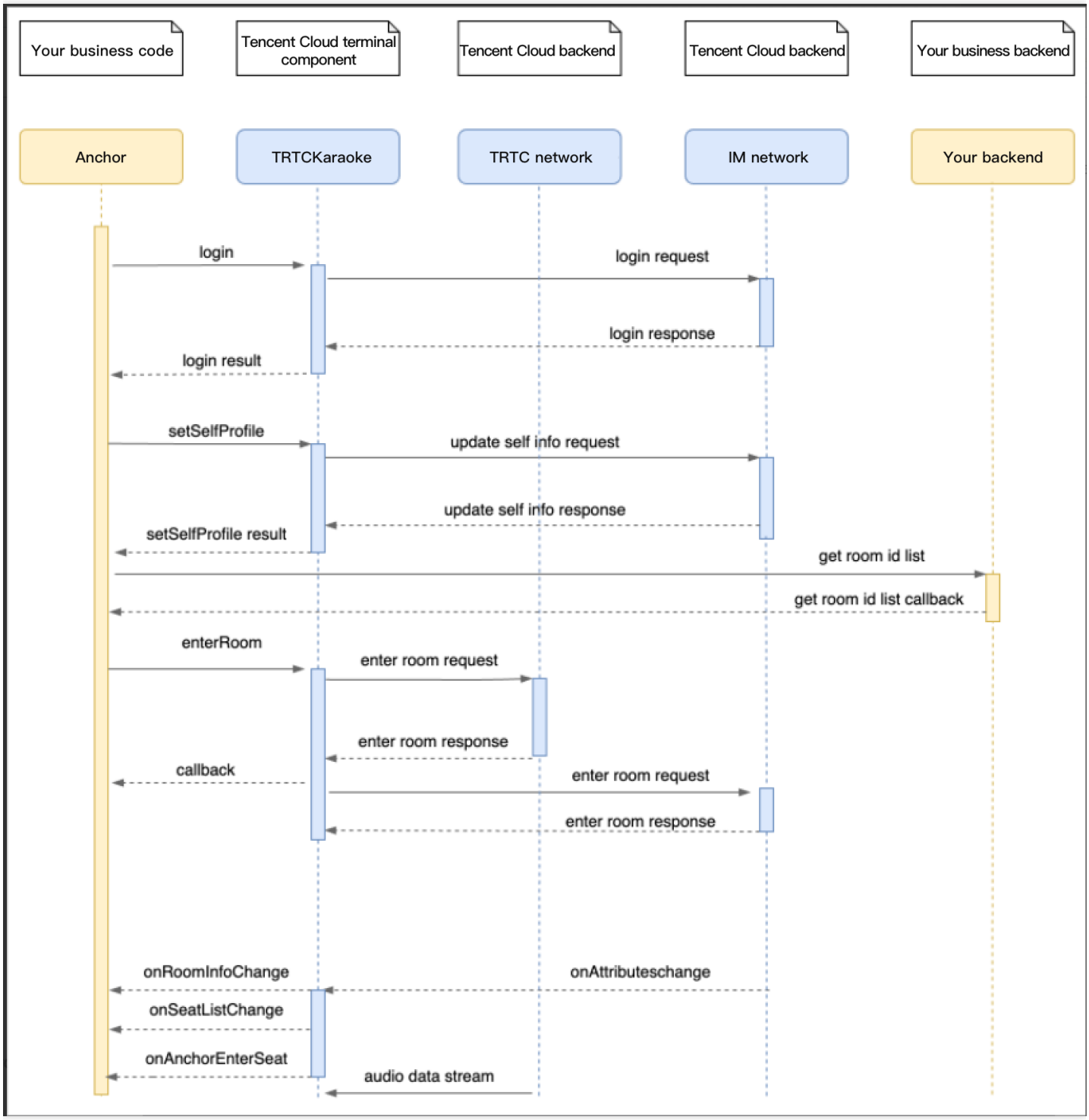
3. リスナー側は、 `getRoomInfoList` を呼び出してルームの詳細情報を取得します。この情報は、管理者側が `createRoom` を呼び出してKTVルームを作成するときに設定する簡単な説明情報です。

©2013-2019 Tencent Cloud. All rights reserved.

> 注意：
> KTVルームリストに十分に包括的な情報がある場合は、 `getRoomInfoList` の呼び出しに関する手順をスキップできます。

4. リスナーは1つのKTVルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
5. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
6. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
7. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。

©2013-2019 Tencent Cloud. All rights reserved.




```
// 1.リスナーは、ニックネームおよびプロフィール画像を設定します
mTRTCKaraokeRoom.setSelfProfile("my_name", "my_face_url", null);

// 2.業務バックエンドから取得したルームリストをroomListと仮定します
List<integer> roomList = GetRoomList();

// 3. getRoomInfoListを呼び出すことによって、ルームの詳細情報を取得します
mTRTCKaraokeRoom.getRoomInfoList(roomList, new TRTCKaraokeRoomCallback.RoomInfoCallback() {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
@Override
public void onCallback(int code, String msg, List<trtckaraokeroomdef.roominfo> list) {
if (code == 0) {
// この時、自分のUIルームリストを更新することができます
}
}
});

// 4.KTVを選択後、roomIdを渡して入室します
mTRTCKaraokeRoom.enterRoom(roomId, new TRTCKaraokeRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//入室に成功
}
}
});

// 5.入室に成功後、onRoomInfoChangeイベント通知を受信します
@Override
public void onRoomInfoChange(TRTCKaraokeRoomDef.RoomInfo roomInfo) {
mNeedRequest = roomInfo.needRequest;
mRoomName = roomInfo.roomName;
// UIはタイトルなどを表示することができます
}

// 6.入室に成功後、onSeatListChangeイベント通知を受信します
@Override
public void onSeatListChange(final List<trtckaraokeroomdef.seatinfo> seatInfoList) {
// マイクリストの表示
}

// 7. onAnchorEnterSeatイベント通知を受信します
@Override
public void onAnchorEnterSeat(TRTCKaraokeRoomDef.UserInfo userInfo) {
}
```

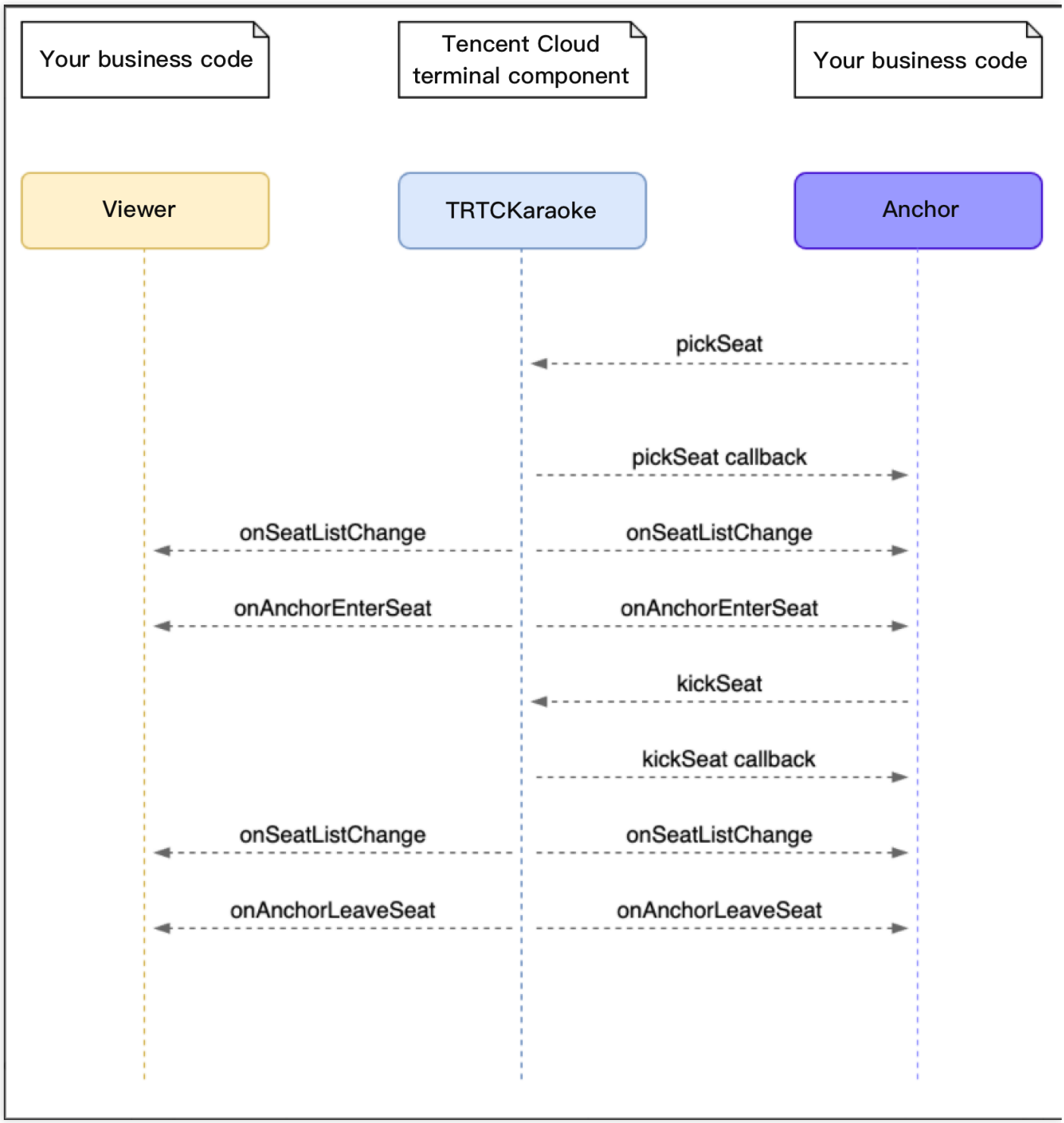
手順7：マイク管理

- 管理者側
- リスナー側

1. `pickSeat` は、対象となるマイクおよびリスナーのuserIdを渡し、発言できるように視聴者を招待できます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。
2. `kickSeat` は、対象となるマイクを渡した後、キックアウトしてマイク・オフにすることができます。ルーム内の全メンバーは `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

3. `muteSeat` は、対応するマイクを渡した後、ミュート／ミュート解除をすることができます。ルーム内の全参加者は `onSeatListChange` および `onSeatMute` のイベント通知を受信します。
4. `closeSeat` は、対応するマイクを送信後、任意のマイクのクローズ/解除をすることができます。クローズ後は、リスナー側はこれ以上マイクを使用することはできません。ルーム内の全参加者は `onSeatListChange` および `onSeatClose` というイベント通知を受信します。



手順8：招待シグナリングの使用

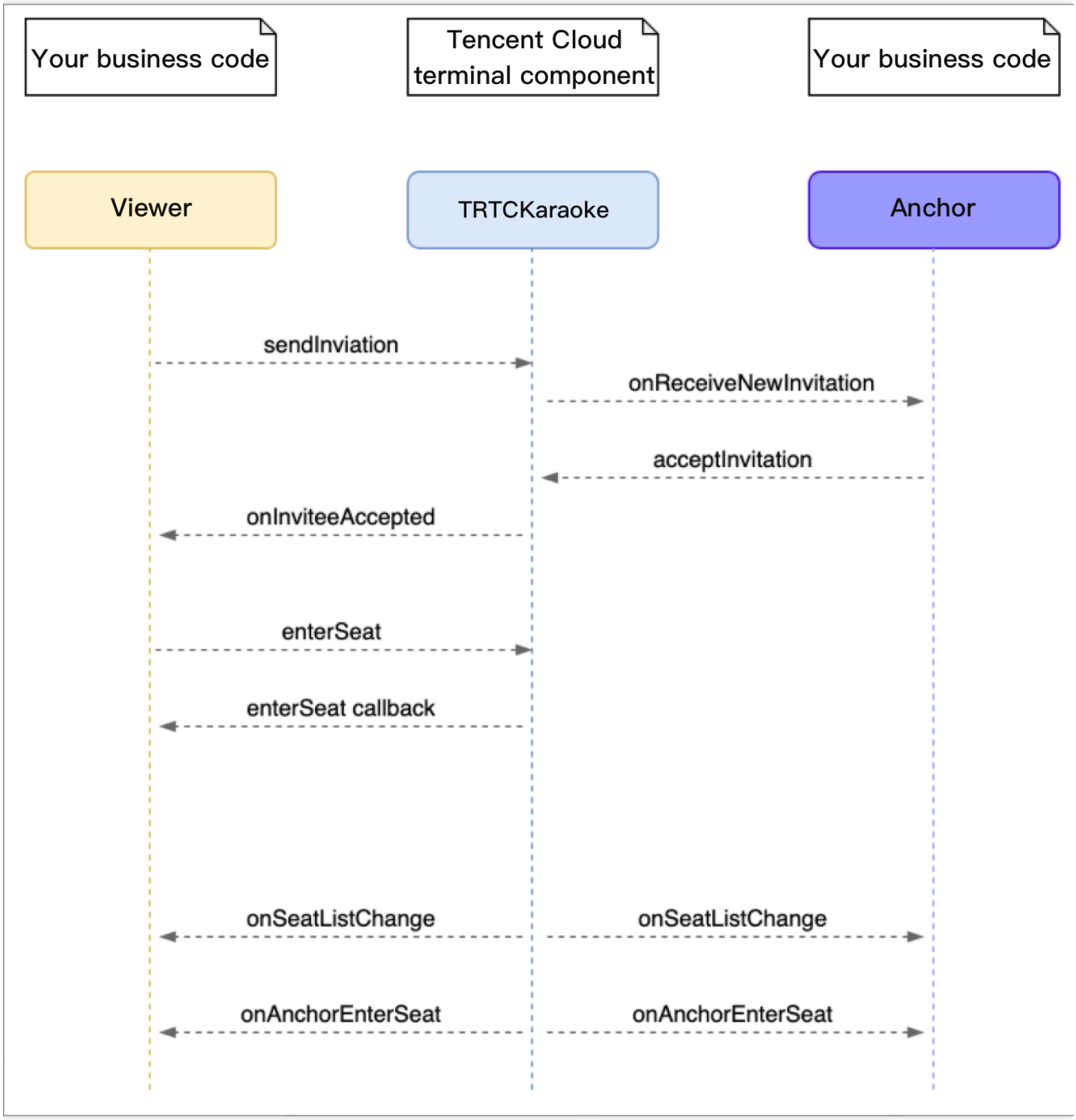
©2013-2019 Tencent Cloud. All rights reserved.

マイク管理では、リスナーがマイク・オン/オフにする場合や、管理者が視聴者を発言できるように招待する場合は、相手の同意を必要とせずに直接操作することができます。
Appにおいて相手の同意がなければ、次の操作の業務フローを実施できない場合は、招待シグナリングによって相応のサポートを行うことができます。

- リスナーからの自主的なマイク・オン申請
- 管理者からの招待によるリスナーのマイク・オン

1. リスナー側が `sendInvitation` を呼び出し、管理者のuserIdおよび業務のカスタムコマンドワードなどを渡します。この時、関数が1つのinviteIdを返しますので、このinviteIdを記録します。
2. 管理者側は、 `onReceiveNewInvitation` というイベント通知を受信します。この時UIでウィンドウをポップアップさせ、管理者に同意の有無をたずねることができます。
3. 管理者が同意を選択後、 `acceptInvitation` を呼び出してinviteIdを渡します。
4. リスナー側は、 `onInviteeAccepted` というイベント通知を受信し、 `enterSeat` を呼び出してマイク・オンにします。

©2013-2019 Tencent Cloud. All rights reserved.




```
// リスナー側の視点
// 1. sendInvitationを呼び出し、1号マイクの使用をリクエストします
String inviteId = mTRTCKaraokeRoom.sendInvitation("ENTER_SEAT", ownerUserId, "1", null);

// 2.招待のリクエスト同意を受信し、正式にマイク・オンになります
@Override
public void onInviteeAccepted(String id, String invitee) {
```

©2013-2019 Tencent Cloud. All rights reserved.

```
if(id.equals(inviteId)) {
mTRTCKaraokeRoom.enterSeat(1, null);
}
}

// 管理者側の視点
// 1.管理者がリクエストを受信します
@Override
public void onReceiveNewInvitation(final String id, String inviter, String cmd, final String cont
ent) {
if (cmd.equals("ENTER_SEAT")) {
// 2.管理者がリスナーのリクエストに同意します
mTRTCKaraokeRoom.acceptInvitation(id, null);
}
}
```

手順9：文字チャットおよび弾幕コメントの実装

- `sendRoomTextMsg` によって通常のテキストメッセージを送信できるようになり、このルーム内のすべてのキャスターおよびリスナーが `onRecvRoomTextMsg` のコールバックを受信することができます。
  IMバックエンドは、デフォルトでNGワードフィルター規則を有し、NGワードと認識されたテキストメッセージはクラウドに転送されることはありません。

```
// 発信側：テキストメッセージの発信
mTRTCKaraokeRoom.sendRoomTextMsg("Hello Word!", null);
// 受信側：テキストメッセージのモニタリング
mTRTCKaraokeRoom.setDelegate(new TRTCKaraokeRoomDelegate() {
@Override
public void onRecvRoomTextMsg(String message, TRTCKaraokeRoomDef.UserInfo userInfo) {
Log.d(TAG,"が" + userInfo.userName + "から受信したメッセージ:" + message);
}
});
```

- `sendRoomCustomMsg` によって、カスタム（シグナル）メッセージを送信できます。当該ルーム内のすべてのキャスターとリスナーが `onRecvRoomCustomMsg` コールバックを受信できます。
  カスタムメッセージは、カスタマイズ信号の送信によく用いられます。例えば、「いいね」情報の発信やブロードキャストに使用します。

```
// 発信側：カスタマイズCmdによって弾幕と「いいね」情報を区別することができます
// eg:「CMD_DANMU」は弾幕コメントを表し、「CMD_LIKE」は「いいね」情報を表します
mTRTCKaraokeRoom.sendRoomCustomMsg("CMD_DANMU", "Hello world", null);
mTRTCKaraokeRoom.sendRoomCustomMsg("CMD_LIKE", "", null);
// 受信側：カスタムメッセージのモニタリング
mTRTCKaraokeRoom.setDelegate(new TRTCKaraokeRoomDelegate() {
@Override
public void onRecvRoomCustomMsg(String cmd, String message, TRTCKaraokeRoomDef.UserInfo userIn
```

©2013-2019 Tencent Cloud. All rights reserved.

```
fo) {
if ("CMD_DANMU".equals(cmd)) {
// 弾幕コメントの受信
Log.d(TAG, "が" + userInfo.userName + "から受信した弾幕コメント:" + message);
} else if ("CMD_LIKE".equals(cmd)) {
// 「いいね」情報の受信
Log.d(TAG, userInfo.userName + "いいねを付けました！");
}
}
});
```

©2013-2019 Tencent Cloud. All rights reserved.

TRTCKaraoke(iOS)

最終更新日：：2021-11-29 16:43:04

TRTCKaraokeRoomは、Tencent CloudのTRTCおよびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- 管理者が作成した新しいKaraokeルームが配信を開始すると、リスナーはKaraokeルームに入室して聴取/インタラクションを行います。
- 管理者は、楽曲の順序を管理し、マイク・オンのキャスターをキックアウトすることもできます。
- 管理者はまた、座席をクローズすることができ、その他のリスナーはマイク・オンを申請することができなくなります。
- リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、マイク・オン後は楽曲の選択や歌唱ができるようになります。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
- ギフトや各種のテキストメッセージ、カスタムメッセージの送信をサポートします。カスタムメッセージを弾幕、「いいね」などを実装するために使用することができます。

TRTCKaraokeRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実装プロセスについては、 [Karaoke(iOS)をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用しています。
- IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCKaraokeRoom API概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。
setDelegate	イベントコールバックを設定します。
delegateQueue	イベントコールバックが配置されているスレッドを設定します。
login	ログイン。
logout	ログアウト。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。nilの場合は、ルーム内全員の情報を取得します。

音楽再生インターフェース

API	説明
startPlayMusic	音楽の再生を開始します。
stopPlayMusic	音楽の再生を停止します。
pausePlayMusic	音楽の再生を一時停止します。
resumePlayMusic	音楽の再生を再開します。

マイク管理インターフェース

API	説明
enterSeat	ユーザーが発言者になる（リスナー側/管理者ともに呼び出し可）。
leaveSeat	自主的にマイク・オフ（キャスターが呼び出し）。
pickSeat	視聴者が発言できるように招待（管理者が呼び出し）。
kickSeat	キックアウトしてマイク・オフ（管理者が呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
muteSeat	任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
closeSeat	任意のマイクのクローズ/解除（管理者が呼び出し）。

ローカルの音声操作インターフェース

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
setAudioQuality	音質の設定。
muteLocalAudio	ローカルオーディオミュートの開始/停止。
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。
setVoiceEarMonitorEnable	インイヤーモニタリングのオン/オフ。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerを取得します。

メッセージ送信関連インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

招待シグナリング関連インターフェース

API	説明
sendInvitation	ユーザーに招待を送信。
acceptInvitation	招待の同意。
rejectInvitation	招待の辞退。
cancelInvitation	招待の取り消し。

TRTCKaraokeRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが廃棄された時のコールバック。
onRoomInfoChange	ボイスチャットルーム情報変更のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onSeatListChange	全量のマイクリストの変更。
onAnchorEnterSeat	発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。
onAnchorLeaveSeat	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。
onSeatMute	管理者のマイクミュート。
onUserMicrophoneMute	ユーザーのマイクがミュートされているかどうか。
onSeatClose	管理者のマイククローズ。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

シグナリングイベントのコールバック

API	説明
onReceiveNewInvitation	新規招待リクエストの受信。
onInviteeAccepted	被招待者が招待に同意。
onInviteeRejected	被招待者が招待を拒否。
onInvitationCancelled	招待者が招待を取り消し。

楽曲イベントコールバック

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onMusicProgressUpdate	楽曲再生進捗度のコールバック。
onMusicPrepareToPlay	音楽再生準備のコールバック。
onMusicCompletePlaying	音楽再生完了のコールバック。

SDK基本関数

sharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを取得します。

```
/**
* TRTCKaraokeRoomシングルトンオブジェクトの取得
*
* - returns: TRTCKaraokeRoomインスタンス
* - note: {@link TRTCKaraokeRoom#destroySharedInstance()}を呼び出してシングルトンオブジェクトを破棄することができます
*/
+ (instancetype)sharedInstance NS_SWIFT_NAME(shared());
```

destroySharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを破棄します。

> 説明：
> インスタンスの破棄後は、外部にキャッシュされたTRTCKaraokeRoomインスタンスの再使用ができません。改めてsharedInstanceを呼び出し、インスタンスを新規取得する必要があります。

```
/**
* TRTCKaraokeRoomシングルトンオブジェクトの破棄
*
* - note:インスタンスの破棄後は、外部にキャッシュされたTRTCKaraokeRoomインスタンスは再使用ができません。改めて{@link TRTCKaraokeRoom#sharedInstance()}を呼び出し、インスタンスを新規取得する必要があります
*/
+ (void)destroySharedInstance NS_SWIFT_NAME(destroyShared());
```

©2013-2019 Tencent Cloud. All rights reserved.

setDelegate

TRTCKaraokeRoomイベントコールバック。 TRTCKaraokeRoomDelegate を介してTRTCKaraokeLiveRoomの各種ステータス通知を受け取ることができます。

```
/**
* コンポーネントコールバックインターフェースの設定
*
* TRTCKaraokeRoomDelegateによってTRTCKaraokeRoomの各種ステータス通知を取得することができます
*
* - parameter delegateコールバックインターフェース
* - note: TRTCKaraokeRoomのコールバックイベントは、デフォルトではMain Queueでコールバックされます。イベントのコールバックが設定されているキューを指定する必要がある場合は、{@link TRTCKaraokeRoom#setDelegateQueue(queue)}を使用することができます
*/
- (void)setDelegate:(id<TRTCKaraokeRoomDelegate>)delegate NS_SWIFT_NAME(setDelegate(delegate:));
```

説明：
setDelegateはTRTCKaraokeRoomのプロキシコールバックです。

setDelegateQueue

イベントコールバックが所在するスレッドキューを設定し、デフォルトはメインスレッド MainQueue に発送します。

```
/**
* イベントコールバックが配置されているキューの設定
*
* - parameter queueキュー、TRTCKaraokeRoomの各種ステータス通知コールバックは、指定したqueueに発信されます。
*/
- (void)setDelegateQueue:(dispatch_queue_t)queue NS_SWIFT_NAME(setDelegateQueue(queue:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
queue	dispatch_queue_t	TRTCKaraokeRoomの各種ステータス通知は、指定したスレッドキューに発信されます。

login

ログイン

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)login:(int)sdkAppID
userId:(NSString *)userId
userSig:(NSString *)userSig
callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(login(sdkAppID:userId:userSig:callback
:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	TRTCコンソール> 【[アプリケーション管理]（https://console.cloud.tencent.com/trtc/app）】>アプリケーション情報の中でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法についてはUserSigの計算方法をご参照ください。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

logout

ログアウト。

```
- (void)logout:(ActionCallback _Nullable)callback NS_SWIFT_NAME(logout(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ログアウトのコールバック。成功時にcodeは0になります。

setSelfProfile

個人情報の修正。

```
- (void)setSelfProfile:(NSString *)userName avatarURL:(NSString *)avatarURL callback:(ActionCallb
ack _Nullable)callback NS_SWIFT_NAME(setSelfProfile(userName:avatarURL:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像のアドレス。
callback	ActionCallback	個人情報設定のコールバック。成功時にcodeは0になります。

ルーム関連インターフェース関数

createRoom

ルームの作成（管理者が呼び出し）。

```
- (void)createRoom:(int)roomID roomParam:(RoomParam *)roomParam callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(createRoom(roomID:roomParam:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomIDを、1つのボイスチャットルームリストにまとめることができます。Tencent Cloudでは現在、ボイスチャットルームリストの管理サービスを行っていませんので、ご自身でボイスチャットルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報は、ルーム名、マイク情報、カバー情報など、ルームを説明するために用いる情報に使用します。マイク管理が必要な場合は、ルームのマイク数を記入する必要があります。
callback	ActionCallback	ルームの新規作成結果のコールバック。成功時にcodeは0になります。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです。

1. 管理者は、 `createRoom` を呼び出して新しいKaraokeルームを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否、ルームタイプなどルームの属性情報を渡します。
2. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
3. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリスト変更をUI上に更新することができます。

©2013-2019 Tencent Cloud. All rights reserved.

4. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。

```
- (void)destroyRoom:(ActionCallback _Nullable)callback NS_SWIFT_NAME(destroyRoom(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

enterRoom

入室（リスナーが呼び出し）。

```
- (void)enterRoom:(NSInteger)roomID callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(enterRoom(roomID:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

リスナーが入室し聴取する際の通常の呼び出しプロセスは次のとおりです。

1. リスナーがサーバーから取得する最新のKaraokeルームリストには、多くのボイスチャットルームのroomIdおよびルーム情報が含まれる場合があります。
2. リスナーは1つのKaraokeルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
3. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
4. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時にマイクリストの変更をUI上に更新することができます。
5. 入室後にマイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。

©2013-2019 Tencent Cloud. All rights reserved.

exitRoom

ルームから退出します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

```
- (void)getRoomInfoList:(NSArray<NSNumber *> *)roomIdList callback:(KaraokeInfoCallback _Nullable)callback NS_SWIFT_NAME(getRoomInfoList(roomIdList:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIdList	List&dxlt;Integer&dxgt;	ルームナンバーリスト。
callback	RoomInfoCallback	ルーム詳細情報のコールバック。

getUserInfoList

指定されたuserIdのユーザー情報を取得します。

```
- (void)getUserInfoList:(NSArray<NSString *> * _Nullable)userIDList callback:(KaraokeUserListCallback _Nullable)callback NS_SWIFT_NAME(getUserInfoList(userIDList:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userIdList	List&dxlt;String&dxgt;	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。
userlistcallback	UserListCallback	ユーザーの詳細情報のコールバック。

音楽再生インターフェース

startPlayMusic

音楽を再生します（マイク・オン後に呼び出し）。

説明：

- 音楽を再生すると、 `onMusicPrepareToPlay` というイベント通知を受信します。
- 音楽の再生中、ルーム内の全メンバーは、 `onMusicProgressUpdate` というイベント通知を継続して受け取ります。
- 音楽の再生が完了すると、 `onMusicCompletePlaying` というイベント通知を受信します。

```
- (void)startPlayMusic:(int32_t)musicID originalUrl:(NSString *)originalUrl accompanyUrl:(NSStrin
g *)backingUrl NS_SWIFT_NAME(startPlayMusic(musicID:originalUrl:accompanyUrl:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int32_t	音楽のID。
originalUrl	String	原曲の絶対パス。
accompanyUrl	String	伴奏の絶対パス。

このインターフェースを呼び出すと、最後に再生されていた楽曲が停止します。

stopPlayMusic

音楽の再生を停止します（音楽を再生するときに呼び出します）。

説明：

©2013-2019 Tencent Cloud. All rights reserved.

再生が停止すると、 `onMusicCompletePlaying` というイベント通知を受信します。

```
- (void)stopPlayMusic NS_SWIFT_NAME(stopPlayMusic());
```

pausePlayMusic

音楽の再生を一時停止します（音楽を再生するときに呼び出します）。

説明：

- `onMusicProgressUpdate` というイベント通知が一時停止されます
- `onMusicCompletePlaying` というイベント通知を受信しません。

```
- (void)pausePlayMusic NS_SWIFT_NAME(pausePlayMusic());
```

resumePlayMusic

一時停止した音楽を再開します（一時停止後に呼び出します）。

説明：
`onMusicPrepareToPlay` というイベント通知を受信しません。

```
- (void)resumePlayMusic NS_SWIFT_NAME(resumePlayMusic());
```

マイク管理インターフェース

enterSeat

ユーザーが発言者になります（リスナー側/管理者ともに呼び出し可）。

説明：
マイク・オンの成功後、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)enterSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(enterSeat(seatIndex:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	マイク・オンの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、`onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

leaveSeat

ユーザーが視聴者になる（キャスターが呼び出し）。

> 説明：
> マイク・オフの成功後、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

```
- (void)leaveSeat:(ActionCallback _Nullable)callback NS_SWIFT_NAME(leaveSeat(callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

pickSeat

視聴者が発言できるように招待（管理者が呼び出し）。

> 説明：
> 管理者が視聴者を発言できるように招待すると、ルーム内の全メンバーは、`onSeatListChange` と `onAnchorEnterSeat` というイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)pickSeat:(NSInteger)seatIndex userId:(NSString *)userId callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(pickSeat(seatIndex:userId:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	視聴者が発言できるように招待する必要があるマイク番号。
userId	String	ユーザーID。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。管理者がリスナーの同意がなければマイク・オンできないケースの場合は、まず `sendInvitation` を呼び出してからリスナーに申請し、`onInvitationAccept` を受信すると、この関数を呼び出せるようになります。

kickSeat

キックアウトしてマイク・オフ（管理者が呼び出し）。

> 説明：
> 管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

```
- (void)kickSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(kickSeat(seatIndex:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	キックアウトしてマイク・オフの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

muteSeat

任意のマイクのミュート/ミュート解除（管理者が呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

説明：
任意のマイクのミュート/ミュート解除は、ルーム内の全メンバーが `onSeatListChange` および `onSeatMute` というイベント通知を受信します。

```
- (void)muteSeat:(NSInteger)seatIndex isMute:(BOOL)isMute callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(muteSeat(seatIndex:isMute:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isMute	boolean	true：該当するマイクのミュート；false：該当するマイクのミュート解除。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。seatIndexの座席に該当するキャスターは、muteAudioを自動的に呼び出してミュート/ミュートオフにします。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。

説明：
管理者は、該当するマイクをクローズ/解除し、ルーム内の全メンバーは `onSeatListChange` および `onSeatClose` というイベント通知を受信します。

```
- (void)closeSeat:(NSInteger)seatIndex isClose:(BOOL)isClose callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(closeSeat(seatIndex:isClose:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isClose	boolean	true：該当するマイクのクローズ； false：該当するマイクのクローズ解除。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。該当するseatIndexの座席上のキャスターはクローズされ、自動的にマイク・オフになります。

ローカル音声操作のインターフェース

startMicrophone

マイクの集音開始。

```
- (void)startMicrophone;
```

stopMicrophone

マイクの集音停止。

```
- (void)stopMicrophone;
```

setAudioQuality

音質の設定。

```
- (void)setAuidoQuality:(NSInteger)quality NS_SWIFT_NAME(setAuidoQuality(quality:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	int	音声品質。詳細は TRTC SDKをご参照ください。

muteLocalAudio

ローカルの音声のミュート/ミュート取り消し。

```
- (void)muteLocalAudio:(BOOL)mute NS_SWIFT_NAME(muteLocalAudio(mute:));
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
mute	boolean	ミュート/ミュート取り消し。詳細は TRTC SDKをご参照ください。

setSpeaker

スピーカーの起動設定。

```
- (void)setSpeaker:(BOOL)userSpeaker NS_SWIFT_NAME(setSpeaker(userSpeaker:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
useSpeaker	boolean	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。

```
- (void)setAudioCaptureVolume:(NSInteger)volume NS_SWIFT_NAME(setAudioCaptureVolume(volume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、0 - 100、 デフォルト100。

setAudioPlayoutVolume

再生音量の設定。

```
- (void)setAudioPlayoutVolume:(NSInteger)volume NS_SWIFT_NAME(setAudioPlayoutVolume(volume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	再生音量、0 - 100、 デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)muteRemoteAudio:(NSString *)userId mute:(BOOL)mute NS_SWIFT_NAME(muteRemoteAudio(userId:mute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定ユーザーID。
mute	boolean	true：ミュート起動；false：ミュート停止。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。

```
- (void)muteAllRemoteAudio:(BOOL)isMute NS_SWIFT_NAME(muteAllRemoteAudio(isMute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
isMute	boolean	true：ミュートを起動、false：ミュートを停止。

setVoiceEarMonitorEnable

インイヤーモニタリングのオン/オフ。

```
- (void)setVoiceEarMonitorEnable:(BOOL)enable NS_SWIFT_NAME(setVoiceEarMonitor(enable:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
enable	boolean	true：インイヤーモニタリングをオン。false：インイヤーモニタリングをオフ。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

バックグラウンド・サウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

```
- (TXAudioEffectManager * _Nullable)getAudioEffectManager;
```

©2013-2019 Tencent Cloud. All rights reserved.

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

```
- (void)sendRoomTextMsg:(NSString *)message callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendRoomTextMsg(message:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

```
- (void)sendRoomCustomMsg:(NSString *)cmd message:(NSString *)message callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendRoomCustomMsg(cmd:message:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

招待シグナリング関連インターフェース

sendInvitation

ユーザーに招待を送信。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (NSString *)sendInvitation:(NSString *)cmd
userId:(NSString *)userId
content:(NSString *)content
callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendInvitation(cmd:userId:content:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	業務カスタマイズコマンド。
userId	String	招待ユーザーID。
content	String	招待コンテンツ。
callback	ActionCallback	送信結果のコールバック。

戻り値：

戻り値	タイプ	意味
inviteId	String	今回の招待IDの識別に使用。

acceptInvitation

招待の同意。

```
- (void)acceptInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(acceptInvitation(identifier:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

rejectInvitation

招待の拒否。

```
- (void)rejectInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(rejectInvitation(identifier:callback:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

cancelInvitation

招待の取り消し。

```
- (void)cancelInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(cancelInvitation(identifier:callback:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

TRTCKaraokeRoomDelegateイベントコールバック

一般的なイベントコールバック

onError

エラーのコールバック。

> 説明：
> SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

```
- (void)onError:(int)code
message:(NSString*)message
NS_SWIFT_NAME(onError(code:message:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラーメッセージ。

onWarning

警告のコールバック。

```
- (void)onWarning:(int)code
message:(NSString *)message
NS_SWIFT_NAME(onWarning(code:message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	警告メッセージ。

onDebugLog

Logコールバック。

```
- (void)onDebugLog:(NSString *)message
NS_SWIFT_NAME(onDebugLog(message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	ログ情報。

ルームイベントのコールバック

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onRoomDestroy:(NSString *)message
NS_SWIFT_NAME(onRoomDestroy(message:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	コールバック情報。

onRoomInfoChange

入室に成功後、このインターフェースをコールバックします。roomInfoの情報は、管理者がルームを作成するときに渡されます。

```
- (void)onRoomInfoChange:(KaraokeInfo *)roomInfo
NS_SWIFT_NAME(onRoomInfoChange(roomInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomInfo	RoomInfo	ルーム情報。

onUserMicrophoneMute

ユーザーのマイクがミュートになっているかどうかについて、ユーザーがmuteLocalAudioを呼び出すと、ルームの他のユーザーがこの通知を受信します。

```
- (void)onUserMicrophoneMute:(NSString *)userId mute:(BOOL)mute
NS_SWIFT_NAME(onUserMicrophoneMute(userId:mute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
mute	boolean	音量の大きさ。値：0～100。

onUserVolumeUpdate

音量レベルリマインダを有効にして、各メンバーの音量を通知します。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onUserVolumeUpdate:(NSArray<TRTCVolumeInfo *> *)userVolumes totalVolume:(NSInteger)totalVolume
NS_SWIFT_NAME(onUserVolumeUpdate(userVolumes:totalVolume:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userVolumes	List	ユーザーリスト。
totalVolume	int	音量の大きさ。値：0～100。

マイクコールバック

onSeatListChange

全量のマイクリストの変更は、全てのマイクリストを含みます。

```
- (void)onSeatInfoChange:(NSArray<KaraokeSeatInfo *> *)seatInfolist
NS_SWIFT_NAME(onSeatListChange(seatInfoList:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatInfoList	List&dxlt;SeatInfo&dxgt;	全量のマイクリスト。

onAnchorEnterSeat

発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。

```
- (void)onAnchorEnterSeat:(NSInteger)index
user:(KaraokeUserInfo *)user
NS_SWIFT_NAME(onAnchorEnterSeat(index:user:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	メンバーがマイク・オンのマイク。
user	UserInfo	マイク・オンのユーザーの詳細情報。

©2013-2019 Tencent Cloud. All rights reserved.

onAnchorLeaveSeat

視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。

```
- (void)onAnchorLeaveSeat:(NSInteger)index
user:(KaraokeUserInfo *)user
NS_SWIFT_NAME(onAnchorLeaveSeat(index:user:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	マイク・オフのマイク。
user	UserInfo	マイク・オンのユーザーの詳細情報。

onSeatMute

管理者のマイクミュート。

```
- (void)onSeatMute:(NSInteger)index
isMute:(BOOL)isMute
NS_SWIFT_NAME(onSeatMute(index:isMute:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	操作するマイク。
isMute	boolean	true：マイクミュート； false：ミュート解除。

onSeatClose

管理者のマイククローズ。

```
- (void)onSeatClose:(NSInteger)index
isClose:(BOOL)isClose
NS_SWIFT_NAME(onSeatClose(index:isClose:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	操作するマイク。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
isClose	boolean	true：マイクのクローズ； false：マイクのクローズ解除。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。

```
- (void)onAudienceEnter:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceEnter(userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	UserInfo	入室したリスナーの情報。

onAudienceExit

リスナー退室通知の受信。

```
- (void)onAudienceExit:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceExit(userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	UserInfo	退室したリスナーの情報。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

```
- (void)onRecvRoomTextMsg:(NSString *)message
userInfo:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomTextMsg(message:userInfo:));
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージの受信。

```
- (void)onRecvRoomCustomMsg:(NSString *)cmd
message:(NSString *)message
userInfo:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomCustomMsg(cmd:message:userInfo:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
command	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

招待シグナリングイベントのコールバック

onReceiveNewInvitation

新規招待リクエストの受信。

```
- (void)onReceiveNewInvitation:(NSString *)identifier
inviter:(NSString *)inviter
cmd:(NSString *)cmd
content:(NSString *)content
NS_SWIFT_NAME(onReceiveNewInvitation(identifier:inviter:cmd:content:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
id	String	招待ID。
inviter	String	招待者のユーザーID。
cmd	String	業務指定のコマンドワードは、開発者がカスタマイズします。
content	UserInfo	業務指定のコンテンツ。

onInviteeAccepted

被招待者が招待に同意。

```
- (void)onInviteeAccepted:(NSString *)identifier
invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeAccepted(identifier:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInviteeRejected

被招待者による招待の拒否。

```
- (void)onInviteeRejected:(NSString *)identifier
invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeRejected(identifier:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInvitationCancelled

招待者が招待を取り消し。

©2013-2019 Tencent Cloud. All rights reserved.

```
- (void)onInvitationCancelled:(NSString *)identifier
invitee:(NSString *)invitee NS_SWIFT_NAME(onInvitationCancelled(identifier:invitee:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	招待者のユーザーID。

音楽再生ステータスコールバック

onMusicPrepareToPlay

音楽再生準備のコールバック

```
- (void)onMusicPrepareToPlay:(int32_t)musicID
NS_SWIFT_NAME(onMusicPrepareToPlay(musicID:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int32_t	再生時に渡されたmusicID。

onMusicProgressUpdate

楽曲再生進捗度のコールバック

```
- (void)onMusicProgressUpdate:(int32_t)musicID
progress:(NSInteger)progress total:(NSInteger)total
NS_SWIFT_NAME(onMusicProgressUpdate(musicID:progress:total:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int32_t	再生時に渡されたmusicID。
progress	NSInteger	現在の再生時間。単位： ms。
total	NSInteger	合計時間。単位： ms。

©2013-2019 Tencent Cloud. All rights reserved.

onMusicCompletePlaying

音楽再生完了のコールバック

```
- (void)onMusicCompletePlaying:(int32_t)musicID
NS_SWIFT_NAME(onMusicCompletePlaying(musicID:));
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int32_t	再生時に渡されたmusicID。

©2013-2019 Tencent Cloud. All rights reserved.

TRTCKaraoke(Android)

最終更新日：：2021-11-29 16:43:05

TRTCKaraokeRoomは、Tencent CloudのTRTCおよびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。

- 管理者が作成した新しいKaraokeルームが配信を開始すると、リスナーはKaraokeルームに入室して聴取/インタラクションを行います。
- 管理者は、楽曲の順序を管理し、マイク・オンのキャスターをキックアウトすることもできます。
- 管理者はまた、座席をクローズすることができ、その他のリスナーはマイク・オンを申請することができなくなります。
- リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、マイク・オン後は楽曲の選択や歌唱ができるようになります。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
- ギフトや各種のテキストメッセージ、カスタムメッセージの送信をサポートします。カスタムメッセージを弾幕、「いいね」などを実装するために使用することができます。

TRTCKaraokeRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実装プロセスについては、Karaoke（Android)をご参照ください。

- TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用しています。
- IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCKaraokeRoom API概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。
setDelegate	イベントコールバックを設定します。
setDelegateHandler	イベントのコールバックが配置されているスレッドを設定します。
login	ログイン。
logout	ログアウト。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。 nullの場合は、ルーム内全員の情報を取得します。

音楽再生インターフェース

API	説明
startPlayMusic	音楽の再生を開始します。
stopPlayMusic	音楽の再生を停止します。
pausePlayMusic	音楽の再生を一時停止します。
resumePlayMusic	音楽の再生を再開します。

マイク管理インターフェース

API	説明
enterSeat	ユーザーが発言者になる（リスナー側/管理者ともに呼び出し可）。
leaveSeat	自主的にマイク・オフ（キャスターが呼び出し）。
pickSeat	視聴者が発言できるように招待（管理者が呼び出し）。
kickSeat	キックアウトしてマイク・オフ（管理者が呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
muteSeat	任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
closeSeat	任意のマイクのクローズ/解除（管理者が呼び出し）。

ローカルの音声操作インターフェース

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
setAudioQuality	音質の設定。
muteLocalAudio	ローカルオーディオミュートの開始/停止。
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。
setVoiceEarMonitorEnable	インイヤーモニタリングのオン/オフ。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerを取得します。

メッセージ送信関連インターフェース

API	説明

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

招待シグナリング関連インターフェース

API	説明
sendInvitation	ユーザーに招待を送信。
acceptInvitation	招待の同意。
rejectInvitation	招待の辞退。
cancelInvitation	招待の取り消し。

TRTCKaraokeRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが廃棄された時のコールバック。
onRoomInfoChange	Karaoke情報変更のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

©2013-2019 Tencent Cloud. All rights reserved.

API	説明
onSeatListChange	全量のマイクリストの変更。
onAnchorEnterSeat	発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。
onAnchorLeaveSeat	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。
onSeatMute	管理者のマイクミュート。
onUserMicrophoneMute	ユーザーのマイクがミュートされているかどうか。
onSeatClose	管理者のマイククローズ。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

シグナリングイベントのコールバック

API	説明
onReceiveNewInvitation	新規招待リクエストの受信。
onInviteeAccepted	被招待者が招待に同意。
onInviteeRejected	被招待者が招待を拒否。
onInvitationCancelled	招待者が招待を取り消し。

©2013-2019 Tencent Cloud. All rights reserved.

楽曲イベントコールバック

API	説明
onMusicProgressUpdate	楽曲再生進捗度のコールバック。
onMusicPrepareToPlay	音楽再生準備のコールバック。
onMusicCompletePlaying	音楽再生完了のコールバック。

SDK基本関数

sharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを取得します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
context	Context	Androidコンテキスト。内部ではApplicationContextに変換してシステムAPIの呼び出しに使用します

destroySharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを破棄します。

> 説明：
> インスタンスの破棄後は、外部にキャッシュされたTRTCKaraokeRoomインスタンスの再使用ができません。改めてsharedInstanceを呼び出し、インスタンスを新規取得する必要があります。

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

setDelegate

TRTCKaraokeRoomイベントコールバック。 TRTCKaraokeRoomDelegateを介してTRTCKaraokeLiveRoomの各種ステータス通知を受け取ることができます。

©2013-2019 Tencent Cloud. All rights reserved.

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

説明：
setDelegateはTRTCKaraokeRoomのプロキシコールバックです。

setDelegateHandler

イベントコールバックが配置されているスレッドを設定します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
handler	Handler	TRTCKaraokeRoomの各種ステータス通知は、指定したhandlerスレッドに発信されます。

login

ログイン

```
public abstract void login(int sdkAppId,
String userId, String userSig,
TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
sdkAppId	int	TRTCコンソール> 【[アプリケーション管理]（https://console.cloud.tencent.com/trtc/app）】>アプリケーション情報の中でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプでは、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法についてはUserSigの計算方法をご参照ください。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

©2013-2019 Tencent Cloud. All rights reserved.

logout

ログアウト。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ログアウトのコールバック。成功時にcodeは0になります。

setSelfProfile

個人情報の修正。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像のアドレス。
callback	ActionCallback	個人情報設定のコールバック。成功時にcodeは0になります。

ルーム関連インターフェース関数

createRoom

ルームの作成（管理者が呼び出し）。

```
public abstract void createRoom(int roomId, TRTCKaraokeRoomDef.RoomParam roomParam, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomI	int	ルームIDは、ご自身でアサインし、一元管理する必要があります。複数のroomIDを、一つのKaraokeルームリストにまとめることができます。Tencent Cloudでは現在、Karaokeルームリストの管理サービスを行っていませんので、ご自身でKaraokeルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報は、ルーム名、マイク情報、カバー情報など、ルームを説明するために用いる情報に使用します。マイク管理が必要な場合は、ルームのマイク数を記入する必要があります。
callback	ActionCallback	ルームの新規作成結果のコールバック。成功時にcodeは0になります。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです。

1. 管理者は、 `createRoom` を呼び出して新しいKaraokeルームを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否、ルームタイプなどルームの属性情報を渡します。
2. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
3. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
4. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

enterRoom

入室（リスナーが呼び出し）。

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

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	int	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

リスナーが入室し聴取する際の通常の呼び出しプロセスは次のとおりです。

1. リスナーがサーバーから取得する最新のKaraokeルームリストには、多くのKaraokeルームのroomIdおよびルーム情報が含まれる場合があります。
2. リスナーは1つのKaraokeルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
3. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、マイク・オンの際の管理者への同意リクエストの要否の記録など。
4. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
5. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。

exitRoom

ルームから退出します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

> 説明：
> ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視してもかまいません。

©2013-2019 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomIdList	List&dxlt;Integer&dxgt;	ルームナンバーリスト。
callback	RoomInfoCallback	ルーム詳細情報のコールバック。

getUserInfoList

指定されたuserIdのユーザー情報を取得します。

```
public abstract void getUserInfoList(List<String> userIdList, TRTCKaraokeRoomCallback.UserListCallback userlistcallback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userIdList	List&dxlt;String&dxgt;	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。
userlistcallback	UserListCallback	ユーザーの詳細情報のコールバック。

音楽再生インターフェース

startPlayMusic

音楽を再生します（マイク・オン後に呼び出し）。

説明：

- 音楽を再生すると、 `onMusicPrepareToPlay` というイベント通知を受信します。
- 音楽の再生中、ルーム内の全メンバーは、 `onMusicProgressUpdate` というイベント通知を継続して受け取ります。
- 音楽の再生が完了すると、 `onMusicCompletePlaying` というイベント通知を受信します。

```
public abstract void startPlayMusic(int musicID, String originalUrl, String accompanyUrl);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int	音楽のID。
originalUrl	String	原曲の絶対パス。
accompanyUrl	String	伴奏の絶対パス。

このインターフェースを呼び出すと、最後に再生されていた楽曲が停止します。

stopPlayMusic

音楽の再生を停止します（音楽を再生するときに呼び出します）。

> 説明：
> 再生が停止すると、 `onMusicCompletePlaying` というイベント通知を受信します。

```
public abstract void stopPlayMusic();
```

pausePlayMusic

音楽の再生を一時停止します（音楽を再生するときに呼び出します）。

> 説明：
>
> - `onMusicProgressUpdate` というイベント通知が一時停止されます
> - `onMusicCompletePlaying` というイベント通知を受信しません。

```
public abstract void pausePlayMusic();
```

resumePlayMusic

一時停止した音楽を再開します（一時停止後に呼び出します）。

> 説明：

©2013-2019 Tencent Cloud. All rights reserved.

`onMusicPrepareToPlay` というイベント通知を受信しません。

```
public abstract void resumePlayMusic();
```

マイク管理インターフェース

enterSeat

ユーザーが発言者になります（リスナー側/管理者ともに呼び出し可）。

説明：
マイク・オンの成功後、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。

```
public abstract void enterSeat(int seatIndex, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	マイク・オンの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、 `onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

leaveSeat

ユーザーが視聴者になる（キャスターが呼び出し）。

説明：
マイク・オフの成功後、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

```
public abstract void leaveSeat(TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

pickSeat

視聴者が発言できるように招待（管理者が呼び出し）。

> 説明：
> 、管理者が視聴者を発言できるように招待すると、ルーム内の全メンバーは、 `onSeatListChange` と `onAnchorEnterSeat` というイベント通知を受信します。

```
public abstract void pickSeat(int seatIndex, String userId, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	視聴者が発言できるように招待する必要があるマイク番号。
userId	String	ユーザーID。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。管理者がリスナーの同意がなければマイク・オンできないケースの場合は、まず `sendInvitation` を呼び出してからリスナーに申請し、 `onInvitationAccept` を受信すると、この関数をコールできるようになります。

kickSeat

キックアウトしてマイク・オフ（管理者が呼び出し）。

> 説明：
> 管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは、 `onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。

©2013-2019 Tencent Cloud. All rights reserved.

```
public abstract void kickSeat(int seatIndex, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	キックアウトしてマイク・オフの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

muteSeat

任意のマイクのミュート/ミュート解除（管理者が呼び出し）。

> 説明：
> 任意のマイクのミュート/ミュート解除は、ルーム内の全メンバーが `onSeatListChange` および `onSeatMute` というイベント通知を受信します。

```
public abstract void muteSeat(int seatIndex, boolean isMute, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isMute	boolean	true：該当するマイクのミュート；false：該当するマイクのミュート解除。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。seatIndexの座席に該当するキャスターは、muteAudioを自動的に呼び出してミュート/ミュートオフにします。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。

©2013-2019 Tencent Cloud. All rights reserved.

説明：
管理者は、該当するマイクをクローズ/解除し、ルーム内の全メンバーは `onSeatListChange` および `onSeatClose` というイベント通知を受信します。

```
public abstract void closeSeat(int seatIndex, boolean isClose, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isClose	boolean	true：該当するマイクのクローズ； false：該当するマイクのクローズ解除。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。該当するseatIndexの座席上のキャスターはクローズされ、自動的にマイク・オフになります。

ローカル音声操作のインターフェース

startMicrophone

マイクの集音開始。

```
public abstract void startMicrophone();
```

stopMicrophone

マイクの集音停止。

```
public abstract void stopMicrophone();
```

setAudioQuality

音質の設定。

```
public abstract void setAudioQuality(int quality);
```

©2013-2019 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
quality	int	音声品質。詳細は TRTC SDKをご参照ください。

muteLocalAudio

ローカルの音声のミュート/ミュート取り消し。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	ミュート/ミュート取り消し。詳細は TRTC SDKをご参照ください。

setSpeaker

スピーカーの起動設定。

```
public abstract void setSpeaker(boolean useSpeaker);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
useSpeaker	boolean	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。

```
public abstract void setAudioCaptureVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	集音音量、0 - 100、 デフォルト100。

setAudioPlayoutVolume

©2013-2019 Tencent Cloud. All rights reserved.

再生音量の設定。

```
public abstract void setAudioPlayoutVolume(int volume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volume	int	再生音量、0 - 100、 デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	指定ユーザーID。
mute	boolean	true：ミュート起動、false：ミュート停止。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
mute	boolean	true：ミュート起動、false：ミュート停止。

setVoiceEarMonitorEnable

インイヤーモニタリングのオン/オフ。

```
public abstract void setVoiceEarMonitorEnable(boolean enable);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
enable	boolean	true：インイヤーモニタリングをオン。false：インイヤーモニタリングをオフ。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

バックグラウンド・サウンドエフェクト管理オブジェクト TXAudioEffectManagerの取得。

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

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
cmd	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

招待シグナリング関連インターフェース

sendInvitation

ユーザーに招待を送信。

```
public abstract String sendInvitation(String cmd, String userId, String content, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
cmd	String	業務カスタマイズコマンド。
userId	String	招待ユーザーID。
content	String	招待コンテンツ。
callback	ActionCallback	送信結果のコールバック。

戻り値：

戻り値	タイプ	意味
inviteId	String	今回の招待IDの識別に使用。

acceptInvitation

招待の同意。

```
public abstract void acceptInvitation(String id, TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

rejectInvitation

招待の拒否。

```
public abstract void rejectInvitation(String id, TRTCKaraokeRoomCallback.ActionCallback callback)
;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

cancelInvitation

招待の取り消し。

```
public abstract void cancelInvitation(String id, TRTCKaraokeRoomCallback.ActionCallback callback)
;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

TRTCKaraokeRoomDelegateイベントコールバック

一般的なイベントコールバック

onError

©2013-2019 Tencent Cloud. All rights reserved.

エラーのコールバック。

説明：
SDKリカバリー不能なエラーは必ず監視し、状況に応じてユーザーに適切なインターフェースプロンプトを表示します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラーメッセージ。

onWarning

警告のコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	警告メッセージ。

onDebugLog

Logコールバック。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	ログ情報。

©2013-2019 Tencent Cloud. All rights reserved.

ルームイベントのコールバック

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。

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

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomId	String	ルームID。

onRoomInfoChange

入室に成功後、このインターフェースをコールバックします。roomInfoの情報は、管理者がルームを作成するときに渡されます。

```
void onRoomInfoChange(TRTCKaraokeRoomDef.RoomInfo roomInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomInfo	RoomInfo	ルーム情報。

onUserMicrophoneMute

ユーザーのマイクがミュートになっているかどうかについて、ユーザーがmuteLocalAudioを呼び出すと、ルームの他のユーザーがこの通知を受信します。

```
void onUserMicrophoneMute(String userId, boolean mute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userId	String	ユーザーID。
mute	boolean	音量の大きさ。値：0～100。

onUserVolumeUpdate

©2013-2019 Tencent Cloud. All rights reserved.

音量レベルリマインダを有効にして、各メンバーの音量を通知します。

```
void onUserVolumeUpdate(List<TRTCCloudDef.TRTCVolumeInfo> userVolumes, int totalVolume);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userVolumes	List	ユーザーリスト。
totalVolume	int	音量の大きさ。値：0～100。

マイクコールバック

onSeatListChange

全量のマイクリストの変更は、全てのマイクリストを含みます。

```
void onSeatListChange(List<SeatInfo> seatInfoList);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
seatInfoList	List&dxlt;SeatInfo&dxgt;	全量のマイクリスト。

onAnchorEnterSeat

発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。

```
void onAnchorEnterSeat(int index, TRTCKaraokeRoomDef.UserInfo user);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	メンバーがマイク・オンのマイク。
user	UserInfo	マイク・オンのユーザーの詳細情報。

onAnchorLeaveSeat

視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。

©2013-2019 Tencent Cloud. All rights reserved.

```
void onAnchorLeaveSeat(int index, TRTCKaraokeRoomDef.UserInfo user);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	マイク・オフのマイク。
user	UserInfo	マイク・オフのユーザーの詳細情報。

onSeatMute

管理者のマイクミュート。

```
void onSeatMute(int index, boolean isMute);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	操作するマイク。
isMute	boolean	true：マイクミュート； false：ミュート解除。

onSeatClose

管理者のマイククローズ。

```
void onSeatClose(int index, boolean isClose);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
index	int	操作するマイク。
isClose	boolean	true：マイクのクローズ； false：マイクのクローズ解除。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。

©2013-2019 Tencent Cloud. All rights reserved.

```
void onAudienceEnter(TRTCKaraokeRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	UserInfo	入室したリスナーの情報。

onAudienceExit

リスナー退室通知の受信。

```
void onAudienceExit(TRTCKaraokeRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
userInfo	UserInfo	退室したリスナーの情報。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージの受信。

```
void onRecvRoomTextMsg(String message, TRTCKaraokeRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージの受信。

```
void onRecvRoomCustomMsg(String cmd, String message, TRTCKaraokeRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです。

©2013-2019 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
command	String	コマンドワードは、開発者がカスタマイズします。主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

招待シグナリングイベントのコールバック

onReceiveNewInvitation

新規招待リクエストの受信。

```
void onReceiveNewInvitation(String id, String inviter, String cmd, String content);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
inviter	String	招待者のユーザーID。
cmd	String	業務指定のコマンドワードは、開発者がカスタマイズします。
content	String	業務指定のコンテンツ。

onInviteeAccepted

被招待者が招待に同意。

```
void onInviteeAccepted(String id, String invitee);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInviteeRejected

©2013-2019 Tencent Cloud. All rights reserved.

被招待者による招待の拒否。

```
void onInviteeRejected(String id, String invitee);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInvitationCancelled

招待者が招待を取り消し。

```
void onInvitationCancelled(String id, String inviter);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
id	String	招待ID。
inviter	String	招待者のユーザーID。

音楽再生ステータスコールバック

onMusicPrepareToPlay

音楽再生準備のコールバック

```
void onMusicPrepareToPlay(int musicID);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int	再生時に渡されたmusicID。

onMusicProgressUpdate

楽曲再生進捗度のコールバック

©2013-2019 Tencent Cloud. All rights reserved.

```
void onMusicProgressUpdate(int musicID, long progress, long total);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int	再生時に渡されたmusicID。
progress	long	現在の再生時間。単位： ms。
total	long	合計時間。単位： ms。

onMusicCompletePlaying

音楽再生完了のコールバック

```
void onMusicCompletePlaying(int musicID);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
musicID	int	再生時に渡されたmusicID。

©2013-2019 Tencent Cloud. All rights reserved.