Tencent Real-Time Communication

UIを含む統合ソリューション

製品ドキュメント



©2013-2022 Tencent Cloud. All rights reserved.

Copyright Notice

©2013-2023 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：

UIを含む統合ソリューション
TUIKitの概要
オーディオビデオ通話
コンポーネントの説明（TUICallKit）
クイックアクセス（TUICallKit）
Android
iOS
Web
インターフェースのカスタマイズ（TUICallKit）
Android
iOS
Web
オフライン通知（TUICallKit）
Android
iOS
クラウドレコーディング（TUICallKit）
API参照（TUICallKit）
Android
API概要
TUICallKit
TUICallEngine
TUICallObserver
iOS
API概要
TUICallKit
TUICallEngine
TUICallObserver
Web
API概要
TUICallKit
TUICallEngine
TUICallEvent
アップグレード方法
Web
公開ログ（TUICallKit）

©2013-2022 Tencent Cloud. All rights reserved.

Web
Android & iOS
よくある質問
Android
iOS
Web
多人数ビデオミーティング
TUIRoom（Web）の統合
TUIRoom (Android)の統合
TUIRoom (iOS)の統合
TUIRoom (Flutter)の統合
TUIRoom (Windows&Mac)の統合
TUIRoom (Electron)の統合
TUIRoom APIのクエリー
TUIRoom(Android)
TUIRoom API（iOS）
TRTCMeeting API（Flutter）
TUIRoom(Windows&Mac)
音声インタラクティブライブストリーミング
TUIVoiceRoom (Android)の統合
TUIVoiceRoom (iOS)の統合
TUIVoiceRoom API照会
TRTCVoiceRoom (iOS)
TRTCVoiceRoom (Android)
ビデオインタラクティブライブストリーミング
TUILiveRoom (Android)の統合
TUILiveRoom (iOS)の統合
TUILiveRoom (Flutter)の統合
TUIPusher&TUIPlayer (Web)の統合
TUILiveRoom APIのクエリー
TRTCLiveRoom API (iOS)
TRTCLiveRoom API (Android)
TRTCLiveRoom API (Flutter)
オンラインカラオケ
Quick Integration (TUIKaraoke)
TUIKaraoke (iOS)の統合
TUIKaraoke (Android)の統合
TUIKaraoke APIのクエリー

©2013-2022 Tencent Cloud. All rights reserved.

TRTCKaraoke(iOS)

TRTCKaraoke(Android)

©2013-2022 Tencent Cloud. All rights reserved.

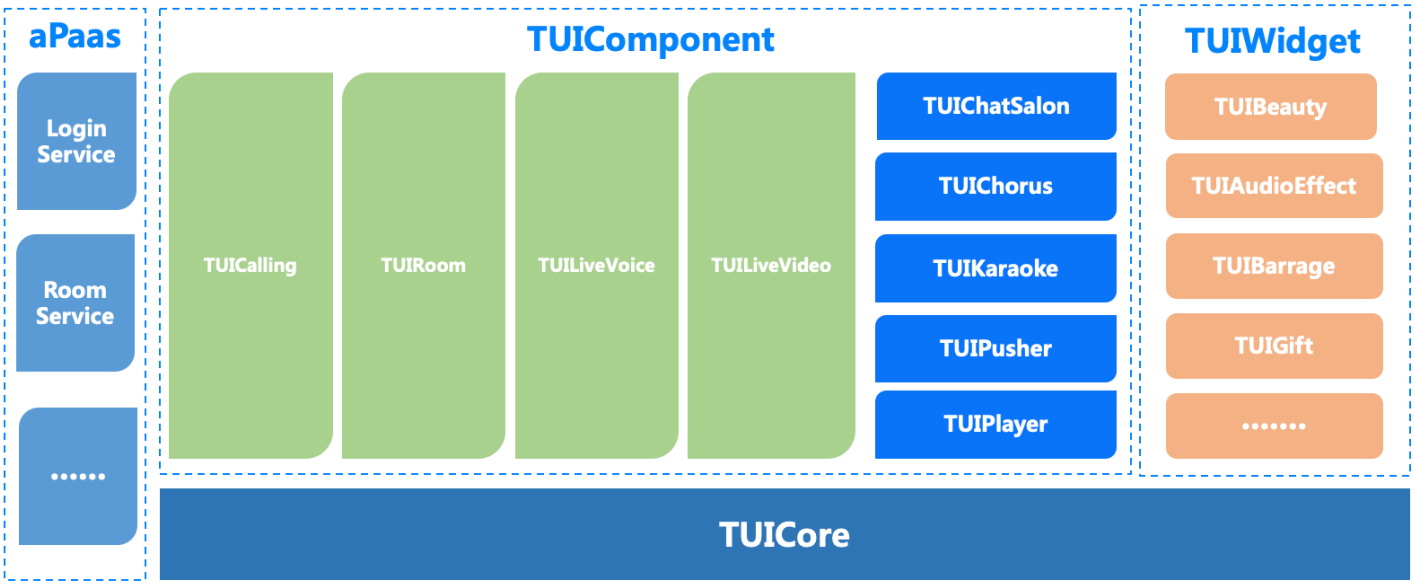
UIを含む統合ソリューション
TUIKitの概要

最終更新日：：2022-10-31 14:41:31

TUIkitは、Tencent Cloudオーディオビデオチームが5000を超えるお客様へのサービス経験を蓄積し、業界内で主流のオーディオビデオシーンを踏まえてブラッシュアップしたオープンソースソリューションです。ビデオ通話コンポーネント、ライブストリーミングコンポーネント、ビデオルームコンポーネントなどの複数のクライアントオーディオビデオコンポーネントが含まれ、通話、カスタマーサービス、ライブストリーミング、ボイスチャット、教育などのシーンをスピーディーに構築したい開発者に役立つソリューションとなっています。

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブ化すると、関連するIM SDKの体験版がデフォルトでアクティブ化されます。これは100 DAUのみをサポートします。

TUIKit製品ラインナップ



©2013-2022 Tencent Cloud. All rights reserved.

上の図のように、TUIKitはTUICompenontとTUIWidgetの2種類に分かれており、オプションの基本バックエンドサービスも同時にサポートしています。

```
├── TUIComponent
|   ├── TUICalling // 通話コンポーネント（WeChat通話に類似）。ビデオ通話、カスタマーサービス、金融審査などのオーディオビデオシーン向けです。
|   ├── TUIRoom // 多人数ビデオルームコンポーネント。教育、ミーティング、面接試験などのオーディオビデオシーン向けです。ミュート、画面オフなどの機能を備えています。
|   ├── TUILiveVoice // ボイスチャットライブストリーミングコンポーネント。フレンドとのボイスチャット、キューボイスチャット、音楽ルームなどのオーディオシーン向けです。
|   ├── TUILiveVideo // 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です。マイク接続、PK、オーディオエフェクトなどの機能を備えています。
|   ├── TUIChatSolon // ボイスサロンコンポーネントです。円卓会議、コミュニティフォーラムなどのオーディオビデオシーン向けです。
|   ├── TUIPusher // 完全なUIを備えたプッシュコンポーネントです。複数のプロトコルプッシュ、PKなどの機能をサポートし、オーディオエフェクト、弾幕などのウィジェットのマウントをサポートします。
|   ├── TUIPlayer // 完全なUIを備えたプルコンポーネントです。複数のプロトコルプル、マイク接続などの機能をサポートし、弾幕、ギフトなどのウィジェットのマウントをサポートします。
|   ├── TUIChorus // 新しいコンポーネントです。合唱の新しい楽しみ方向けのオーディオビデオコンポーネントであり、Tencent Cloud正規版楽曲データベースと組み合わせるなど、楽しみ方をさらに広げることができます。
|   ├── TUIKaraoke // 新しいコンポーネントです。オンラインKTVなどの新しい楽しみ方向けのオーディオビデオコンポーネントであり、Tencent Cloud正規版楽曲データベースと組み合わせるなど、楽しみ方をさらに広げることができます。
├── TUWidget
|   ├── TUIBeauty // 美顔ウィジェット
|   ├── TUIAudioEffect // オーディオエフェクトウィジェット
|   ├── TUIBarrage // 弾幕ウィジェット
|   ├── TUIGift // ギフトウィジェット
├── aPass （オプション）
|   ├── Room Service // ルームリスト、ルーム人気度、キャスターリストなどの機能をご提供します。
```

ご意見&フィードバック

環境、手順が同じであっても異なるエラーが発生することはあり、エンジニアにとっては避けて通れない問題です。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。

©2013-2022 Tencent Cloud. All rights reserved.

オーディオビデオ通話
コンポーネントの説明（TUICallKit）

最終更新日：：2022-11-21 16:34:34

コンポーネントの説明

TUICallKitはTencent Cloudがリリースするオーディオビデオ通話UIコンポーネントです。このコンポーネントを統合することにより、数行のコードを書くだけでAppにオーディオビデオ通話機能を追加することができ、さらにオフライン通知機能もサポートします。TUICallKitはAndroid、iOS、Webなどの複数の開発プラットフォームでサポートしています。基本機能は下図のとおりです。



注意：
開発者からのフィードバックと市場ニーズのリサーチを基に、2022年8月、TUICallingはTUICallKitへとアップグレードしました。新しいコンポーネントはグループ内の多人数通話やオフライン通知などの機能をサ

©2013-2022 Tencent Cloud. All rights reserved.

ポートします。また、より割引率の高いオーディオビデオ通話SDKパッケージおよび60日間の無料トライアル期間をご用意しています。この機会にぜひご利用ください。

機能のメリット

- 便利なアクセス：UI付きのオープンソースコンポーネントを提供することで、開発時間を90%節約でき、オーディオビデオ通話アプリケーションをスピーディーにリリースできます。
- プラットフォーム相互運用性：各プラットフォームのTUICallKitコンポーネントは通話の発信、応答、終了などを相互にサポートし、相互運用が可能です。
- 多人数通話：1対1のビデオ通話だけでなく、グループ内で多人数ビデオ通話を開始することもできます。
- オフラインプッシュ：Android&iOSのオフラインプッシュをサポートしています。着呼側ユーザーのAppはオフラインのときでも新しい着信通知を受け取ることができます。

適用ケース

2人または複数人のオーディオビデオ通話が行えます。ゲームSNS、オンラインカスタマーサービス、ビデオカスタマーサービス、オンライン問診、保険相談などのシーンに適します。

TUICallKitのダウンロード



Android TUICallKit

「WeChat」のようなUIで、1V1通話、グループ通話、フローティングウィンドウ、オフラインプッシュなどの強力な機能をサポートします。

Github　クイックアクセス　API参照　よくあるご質問



iOS TUICallKit

©2013-2022 Tencent Cloud. All rights reserved.

「WeChat」のようなUIで、1V1通話、グループ通話、フローティングウィンドウ、オフラインプッシュなどの強力な機能をサポートします。

Github　クイックアクセス　API参照　よくあるご質問



Web TUICallKit

「WeChat」のようなUIで、1V1通話、グループ通話、フローティングウィンドウ、オフラインプッシュなどの強力な機能をサポートします。

Github　クイックアクセス　API参照　よくあるご質問

オープンソース構築

アップグレード前のTUICallingオープンソースプロジェクトはここで見つけることができます。

©2013-2022 Tencent Cloud. All rights reserved.

クイックアクセス（TUICallKit）
Android

最終更新日：：2022-11-23 15:47:54

ここでは、最短の時間でTUICallKitコンポーネントの統合を完了する方法についてご説明します。このドキュメントに沿って操作することで、1時間以内に次の重要手順を完了し、最終的にUIインターフェースを完備したビデオ通話機能を手にすることができます。

環境の準備

互換性のある最低バージョンはAndroid 4.1（SDK API Level 16）です。Android 5.0（SDK API Level 21）およびそれ以降のバージョンの使用を推奨します。

Android Studio 3.5およびそれ以降のバージョン（Gradle 3.5.4およびそれ以降のバージョン）。

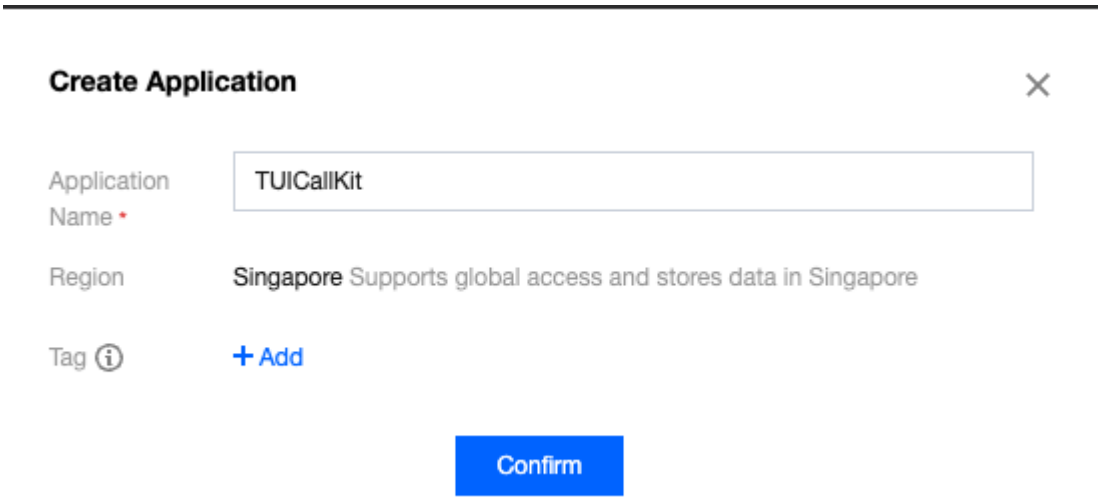
Android 4.1およびそれ以降のスマートフォンデバイス。

ステップ1：サービスのアクティブ化

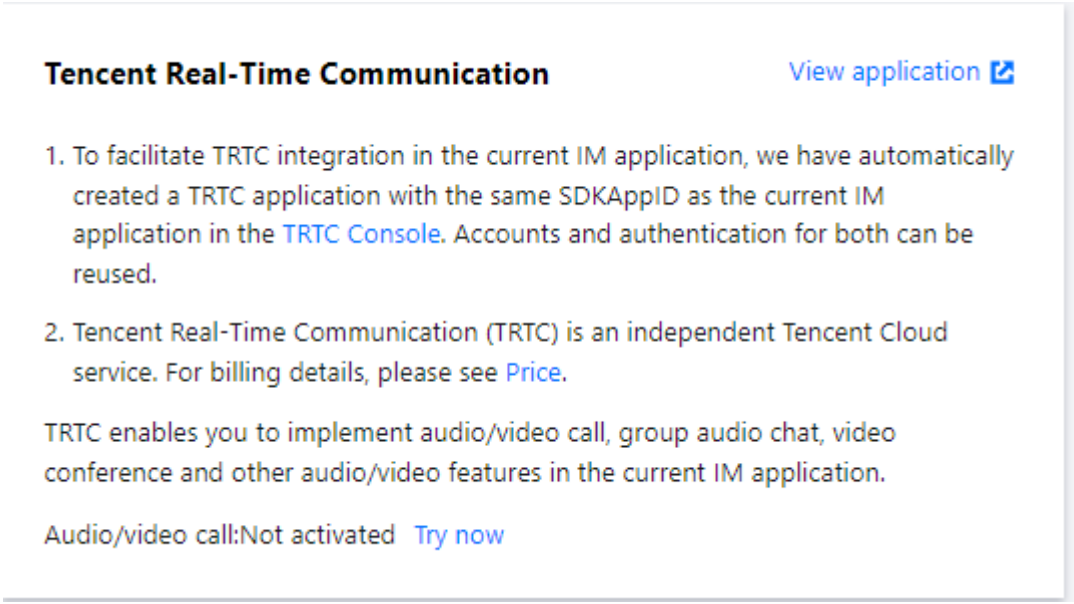
TUICallKitはTencent CloudのIMと、TRTCという2つの有料PaaSサービスをベースに構築したオーディオビデオ通信コンポーネントです。以下の手順で関連のサービスをアクティブ化し、60日間の無料トライアルサービスを体験することができます。

1. IMコンソールにログインし、**新しいアプリケーションの作成**をクリックし、ポップアップしたダイアログボックスにアプリケーション名を入力して**OK**をクリックします。

©2013-2022 Tencent Cloud. All rights reserved.



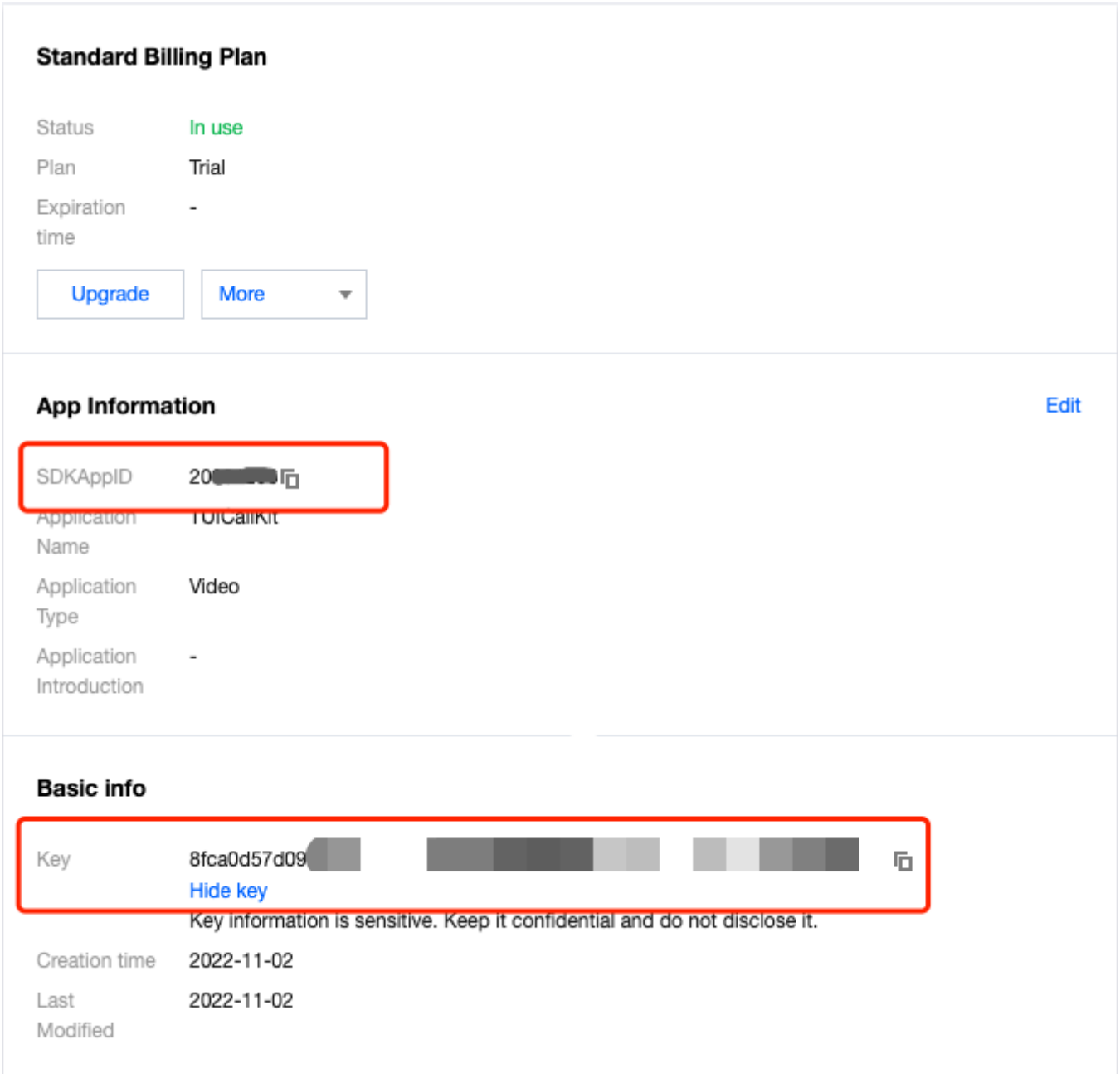
2. 作成したアプリケーションをクリックし、基本設定ページに進み、ページ右下隅のTRTCサービスのアクティブ化機能エリアで無料体験をクリックすると、TUICallKitの7日間無料トライアルサービスをアクティブ化することができます。正式アプリケーションのリリースが必要な場合は、追加購入をクリックすると購入ページに進むことができます。



注意：
IMのオーディオビデオ通話機能は業務ニーズに応じて差別化した有料バージョンをご提供しています。含まれる機能をIM購入ページでご確認の上、ご自身に合ったバージョンを選択してご購入いただけます。

©2013-2022 Tencent Cloud. All rights reserved.

3. 同じページで**SDKAppID**と**キー(SecretKey)**を見つけて記録します。この2つはその後のステップ4：TUIコンポーネントへのログインで使用します。



説明：
お知らせ：**無料体験**をクリックすると、それまでにTRTCサービスを使用したことがある一部のユーザーには、次のような内容が表示されることがあります。

```
[-100013]:TRTC service is suspended. Please check if the package balance is 0 or the Tencent Cloud accountis in arrears
```

©2013-2022 Tencent Cloud. All rights reserved.

新しいIMオーディオビデオ通話機能は、Tencent CloudのTRTCとIMという2つの基本PaaSサービスを統合したもののため、TRTCの無料利用枠（10000分）が期限切れまたは使用済みの場合、このサービスをアクティブ化しようとすると失敗します。その場合は、TRTCコンソールをクリックし、SDKAppIDに対応するアプリケーション管理ページを見つけ、図のように後払い機能をアクティブ化してから**アプリケーションを有効にする**ことで、オーディオビデオ通話機能を正常に体験することができるようになります。

ステップ2：コンポーネントのダウンロードとインポート

Githubでコードをクローン/ダウンロードした後、下図のように、Androidディレクトリ下のtuicallkitサブディレクトリを現在のプロジェクトのappの同階層のディレクトリにコピーします。



ステップ3：プロジェクト設定の完了

1. プロジェクトのルートディレクトリ下で `setting.gradle` ファイルを見つけ、その中に次のコードを追加します。その役割は、ステップ2でダウンロードしたtuicallkitコンポーネントを現在のプロジェクトにインポートするものです。

```
include ':tuicallkit'
```

2. appディレクトリ下で `build.gradle` ファイルを見つけ、その中に次のコードを追加します。その役割は、現在のappの新たに追加したtuicallkitコンポーネントへの依存を明確に述べるものです。

©2013-2022 Tencent Cloud. All rights reserved.

```
api project(':tuicallkit')
```

説明：
TUICallKitプロジェクトの内部はデフォルトで `TRTC SDK` 、 `IM SDK` 、 `tuicallengine` およびパブリックコーパスの `tuicore` に依存しており、開発者が単独で設定する必要はありません。バージョンアップが必要な場合は、 `tuicallkit/build.gradle` ファイルを変更するだけで済みます。

3. SDKの内部ではJavaのリフレクション機能を使用しているため、 SDK内の一部のクラスに非難読化リストを追加する必要があります。このため、 `proguard-rules.pro` ファイルに次のコードを追加する必要があります。

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

注意：
TUICallKitは内部で、カメラ、マイク、ストレージ読み取り権限などの動的申請を補助します。業務上、削減が必要な場合は `tuicallkit/src/main/AndroidManifest.xml` を変更することができます。

ステップ4：TUIコンポーネントへのログイン

プロジェクトに次のコードを追加します。この役割は、TUICore内の関連インターフェースを呼び出して、TUIコンポーネントへのログインを完了することです。ログインに成功しなければTUICallKitの各機能を正常に使用できないため、この手順に異常がないことが重要です。関連のパラメータが正しく設定されているかどうかを注意深く確認してください。

```
//ログイン結果に対するリスナーを設定します
private final TUILoginListener mLoginListener = new TUILoginListener() {
@Override
public void onKickedOffline() {
super.onKickedOffline();
Log.i(TAG, "You have been kicked off the line. Please login again!");
//logout();
}
@Override
```

©2013-2022 Tencent Cloud. All rights reserved.

```
public void onUserSigExpired() {
super.onUserSigExpired();
Log.i(TAG, "Your user signature information has expired");
//logout();
}
};
TUILogin.addLoginListener(mLoginListener);
//ログイン
TUILogin.login(context,
1400000001, // ステップ1で取得したSDKAppIDに置き換えてください
"denny", // ご自身のUserIDに置き換えてください
"xxxxxxxxxx", // コンソールでUserSigを計算し、この位置に入力することができます
new TUICallback() {
@Override
public void onSuccess() {
Log.i(TAG, "login success");
}
@Override
public void onError(int errorCode, String errorMessage) {
Log.e(TAG, "login failed, errorCode: " + errorCode + " msg:" + errorMessage);
}
});
```

パラメータの説明 ここではlogin関数内で使用するいくつかの重要パラメータについて詳しくご説明します。

- **SDKAppID**：ステップ1の最後の段階で取得済みです。ここでは説明を省略します。
- **UserID**：現在のユーザーIDです。文字列タイプでは、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**：ステップ1の第3段階で取得したSecretKeyを使用してSDKAppID、userIDなどの情報を暗号化し、UserSigを取得します。これは認証用の証明書であり、現在のユーザーがTRTCサービスを使用できるかどうかをTencent Cloudが識別するために用いられます。コンソールの**支援ツール**で一時的に使用可能なUserSigを生成することができます。

その他の情報については、UserSigの計算、使用方法をご参照ください。

説明：
この手順は、現在開発者から最も多くのフィードバックが寄せられる手順でもあります。よくみられる問題には次のようなものがあります。

- SDKAppIDの設定に誤りがある。国内サイトのSDKAppIDは一般的に140で始まる10桁の整数です。
- UserSigを誤って暗号化鍵（SecretKey）に設定している。UserSigはSecretKeyを使用して、SDKAppID、UserIDおよび有効期限などの情報を暗号化するためのものであり、SecretKeyは直接UserSigに設定するものではありません。

©2013-2022 Tencent Cloud. All rights reserved.

- UserIDが「1」、「123」、「111」などの簡単な文字列で設定されている。**TRTCは同一のUserIDによる複数端末へのログインをサポートしていない**ため、複数人によるコラボレーション開発の場合、「1」、「123」、「111」のようなUserIDは同僚に占有されていることが多く、ログイン失敗につながりやすいです。このため、デバッグの際に、識別度の高いUserIDを設定することをお勧めします。GithubのサンプルコードでgenTestUserSig関数を使用してローカルでUserSigを計算すると、現在のアクセスフローをさらに速くスタートさせることができますが、この方法ではSecretKeyがAppのコード内で開示されてしまい、その後のアップグレードおよびSecretKeyの保護にとって不利益になります。そのため、UserSigの計算ロジックはサーバーに置いて実行し、またAppがTUICallKitコンポーネントを使用するたびに、リアルタイムに計算したUserSigをサーバーにリクエストするようにすることを強く推奨します。

ステップ5：通話する

1対1ビデオ通話

TUICallKitのcall関数を呼び出し、 通話タイプと着呼者のuserIdを指定すると、音声またはビデオ通話を開始することができます。

```
// 1対1ビデオ通話を開始します(UserIDはmikeとします)
TUICallKit.createInstance(context).call("mike", TUICallDefine.MediaType.Video);
```

パラメータ	タイプ	意味
userId	String	ターゲットユーザーのUserID： `"mike"`
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。 例： `TUICallDefine.MediaType.Video`

グループ内ビデオ通話

TUICallKitのgroupCall関数を呼び出し、 通話タイプと着呼者のUserIDリストを指定すると、グループ内で音声またはビデオ通話を開始することができます。

```
TUICallKit.createInstance(context).groupCall("12345678", Arrays.asList("jane", "mike", "tommy"),TUICallDefine.MediaType.Video);
```

パラメータ	タイプ	意味
groupId	String	グループID。例： `"12345678"`

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userIdList	List	ターゲットユーザーのUserIDリスト。例：`{"jane", "mike", "tommy"}`
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。 例：`TUICallDefine.MediaType.Video`

説明：
グループの作成の詳細についてはIMグループ管理をご参照ください。もしくはIM TUIKitを直接使用し、チャット、通話などのシナリオをワンストップで統合することもできます。

TUICallKitでは現在、グループ以外の多人数ビデオ通話はサポートしていません。もし必要な場合は、colleenyu@tencent.comまでお問い合わせください。

ステップ6：通話に応答する

通話リクエストを受信した後、TUICallKitコンポーネントは着信を通知する応答画面を自動的に呼び出しますが、Androidシステムの権限上の理由により、状況は次のいくつかに分かれます。

- Appがフロントエンドにある場合は、招待を受信した際に呼び出し画面が自動的にポップアップし、着信音が再生されます。
- Appがバックエンドにあっても、`フローティングウィンドウ権限`または`バックエンドポップアップアプリケーション`などの権限が承認されている場合は、呼び出し画面が自動的にポップアップし、着信音が再生されます。
- Appがバックエンドにあり、なおかつ`フローティングウィンドウ権限`または`バックエンドポップアップアプリケーション`などの権限が承認されていない場合は、TUICallKitが着信音を再生し、ユーザーに応答または終了を促します。
- Appのプロセスがすでに破棄されている場合は、オフライン通知（Android）にアクセスし、通知欄のメッセージによってユーザーに応答または終了を促すのみとなります。

ステップ7：その他の特徴

1、ニックネーム&プロフィール画像の設定

カスタムニックネームまたはプロフィール画像を設定したい場合は、次のインターフェースを使用して更新できます。

©2013-2022 Tencent Cloud. All rights reserved.

```
TUICallKit.createInstance(context).setSelfInfo("jack", "https:/****/user_avatar.png", callback);
```

注意：
ユーザーのプライバシー上の制限により、フレンド以外との通話では、呼ばれるニックネームとプロフィール画像の更新が遅れる可能性があります。一度通話が成功するとスムーズに更新されるようになります。

2、オフライン通知

上記の手順が完了すると、オーディオビデオ通話の発信と接続が実現できますが、業務上「アプリケーションのプロセスが強制終了された後」または「アプリケーションがバックエンドに戻った後」もオーディオビデオ通話リクエストを正常に受信できる必要がある場合は、オフライン通知機能を追加する必要があります。詳細についてはオフライン通知（Android）をご参照ください。

3、フローティングウィンドウ機能

業務上、フローティングウィンドウ機能を有効にする必要がある場合は、TUICallKitコンポーネントの初期化の際に次のインターフェースを呼び出してこの機能を有効化することができます。

```
TUICallKit.createInstance(context).enableFloatWindow(true);
```

4、通話ステータスの監視

業務上、通話の開始、終了、通話中のネットワーク品質などの**通話ステータスの監視**が必要な場合は、次のイベントを監視することができます。

```
TUICallEngine.createInstance(context).addObserver(new TUICallObserver() {
@Override
public void onCallBegin(TUICommonDefine.RoomId roomId, TUICallDefine.MediaType callMediaType, TUICallDefine.Role callRole) {
}
public void onCallEnd(TUICommonDefine.RoomId roomId, TUICallDefine.MediaType callMediaType,TUICallDefine.Role callRole, long totalTime) {
}
public void onUserNetworkQualityChanged(List<TUICommonDefine.NetworkQualityInfo> networkQualityList) {
}
});
```

5、カスタム着信音

©2013-2022 Tencent Cloud. All rights reserved.

着信音をカスタマイズしたい場合は、次のインターフェースによって設定できます。

```
TUICallKit.createInstance(context).setCallingBell(filePath);
```

よくあるご質問

1、「The package you purchased does not support this ability」というエラーが表示されました。

上記のエラーが表示された場合は、現在のアプリケーションのオーディオビデオ通話機能パッケージが期限切れまたはアクティブ化されていないことを表します。ステップ1：サービスのアクティブ化を参照してオーディオビデオ通話機能を取得またはアクティブ化し、引き続きTUICallKitコンポーネントをご利用ください。

2、TUICallKitがクラッシュし、クラッシュログが"No implementation found for xxxx"となっています。

スタック情報は次のとおりです。

```
java.lang.UnsatisfiedLinkError: No implementation found for void com.tencent.lite
av.base.Log.nativeWriteLogToNative(int, java.lang.String, java.lang.String) (trie
d Java_com_tencent_liteav_base_Log_nativeWriteLogToNative and Java_com_tencent_li
teav_base_Log_nativeWriteLogToNative__ILjava_lang_String_2Ljava_lang_String_2)
at com.tencent.liteav.base.Log.nativeWriteLogToNative(Native Method)
at com.tencent.liteav.base.Log.i(SourceFile:177)
at com.tencent.liteav.basic.log.TXCLog.i(SourceFile:36)
at com.tencent.liteav.trtccalling.model.impl.base.TRTCLogger.i(TRTCLogger.java:15
)
at com.tencent.liteav.trtccalling.model.impl.ServiceInitializer.init(ServiceIniti
alizer.java:36)
at com.tencent.liteav.trtccalling.model.impl.ServiceInitializer.onCreate(ServiceI
nitializer.java:101)
at android.content.ContentProvider.attachInfo(ContentProvider.java:2097)
at android.content.ContentProvider.attachInfo(ContentProvider.java:2070)
at android.app.ActivityThread.installProvider(ActivityThread.java:8168)
at android.app.ActivityThread.installContentProviders(ActivityThread.java:7709)
at android.app.ActivityThread.handleBindApplication(ActivityThread.java:7573)
at android.app.ActivityThread.access$2600(ActivityThread.java:260)
at android.app.ActivityThread$H.handleMessage(ActivityThread.java:2435)
at android.os.Handler.dispatchMessage(Handler.java:110)
at android.os.Looper.loop(Looper.java:219)
at android.app.ActivityThread.main(ActivityThread.java:8668)
at java.lang.reflect.Method.invoke(Native Method)
at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:5
```

©2013-2022 Tencent Cloud. All rights reserved.

```
13)
at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:1109)
```

TUICallkitは仮想マシンをサポートしていませんので、実機でご体験ください。実機で上記の異常が生じる原因は、TUICallKitの依存するTRTCなどのSDKの一部インターフェースにJNIを使用しているため、Android Studioがいくつかの条件でAPKコンパイルを行う際にNativeの.soライブラリをパッケージ化できず、このようなエラーが発生します。再Cleanを行ってください。

> 説明：
> その他のヘルプ情報についての詳細は、Androidについてのよくあるご質問をご参照ください。

ご意見とフィードバック

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。

©2013-2022 Tencent Cloud. All rights reserved.

iOS

最終更新日：：2022-11-23 15:49:53

ここでは、最短の時間でTUICallKitコンポーネントの統合を完了する方法についてご説明します。このドキュメントに沿って操作することで、1時間以内に次の重要手順を完了し、最終的にUIインターフェースを完備したビデオ通話機能を手にすることができます。

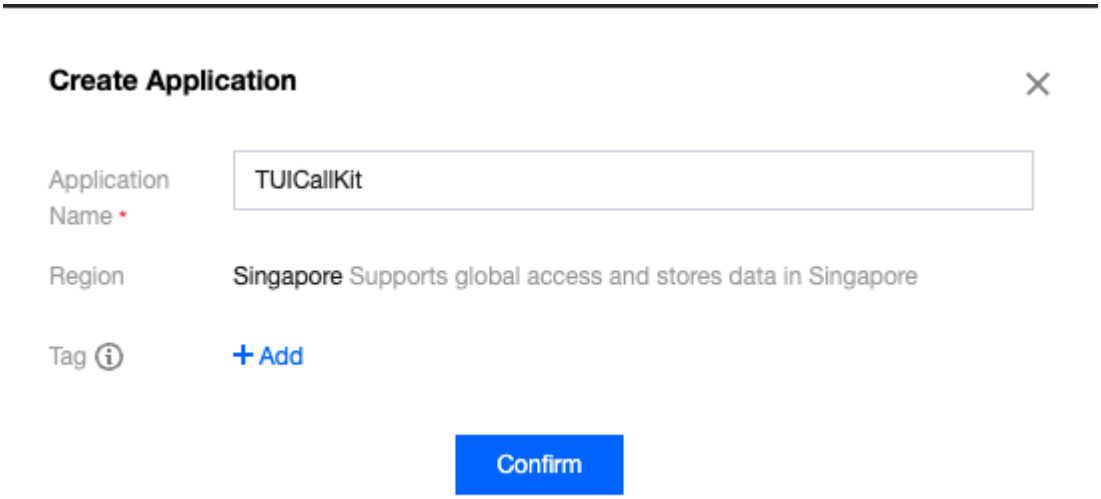
環境の準備

iOS 9.0 (API level 16)およびそれ以上。

ステップ1：サービスのアクティブ化

TUICallKitはTencent CloudのIMと、TRTCという2つの有料PaaSサービスをベースに構築したオーディオビデオ通信コンポーネントです。以下の手順で関連のサービスをアクティブ化し、60日間の無料トライアルサービスを体験することができます。

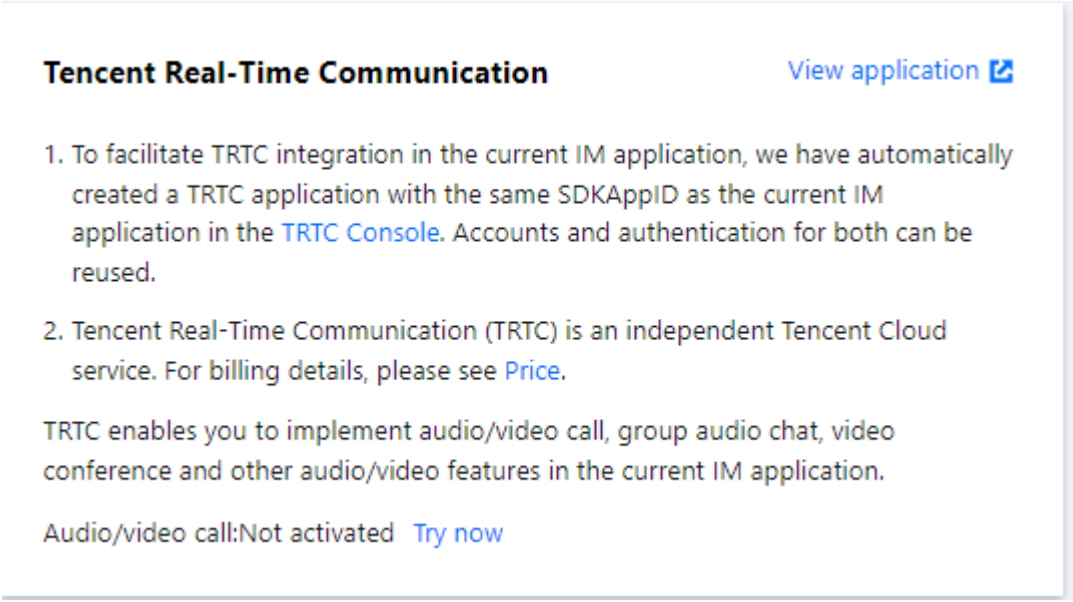
1. IMコンソールにログインし、**新しいアプリケーションの作成**をクリックし、ポップアップしたダイアログボックスにアプリケーション名を入力して**OK**をクリックします。



2. 作成したアプリケーションをクリックし、基本設定ページに進み、ページ右下隅のTRTCサービスのアクティブ化機能エリアで無料体験をクリックすると、TUICallKitの7日間無料トライアルサービスをアクティブ化する

©2013-2022 Tencent Cloud. All rights reserved.

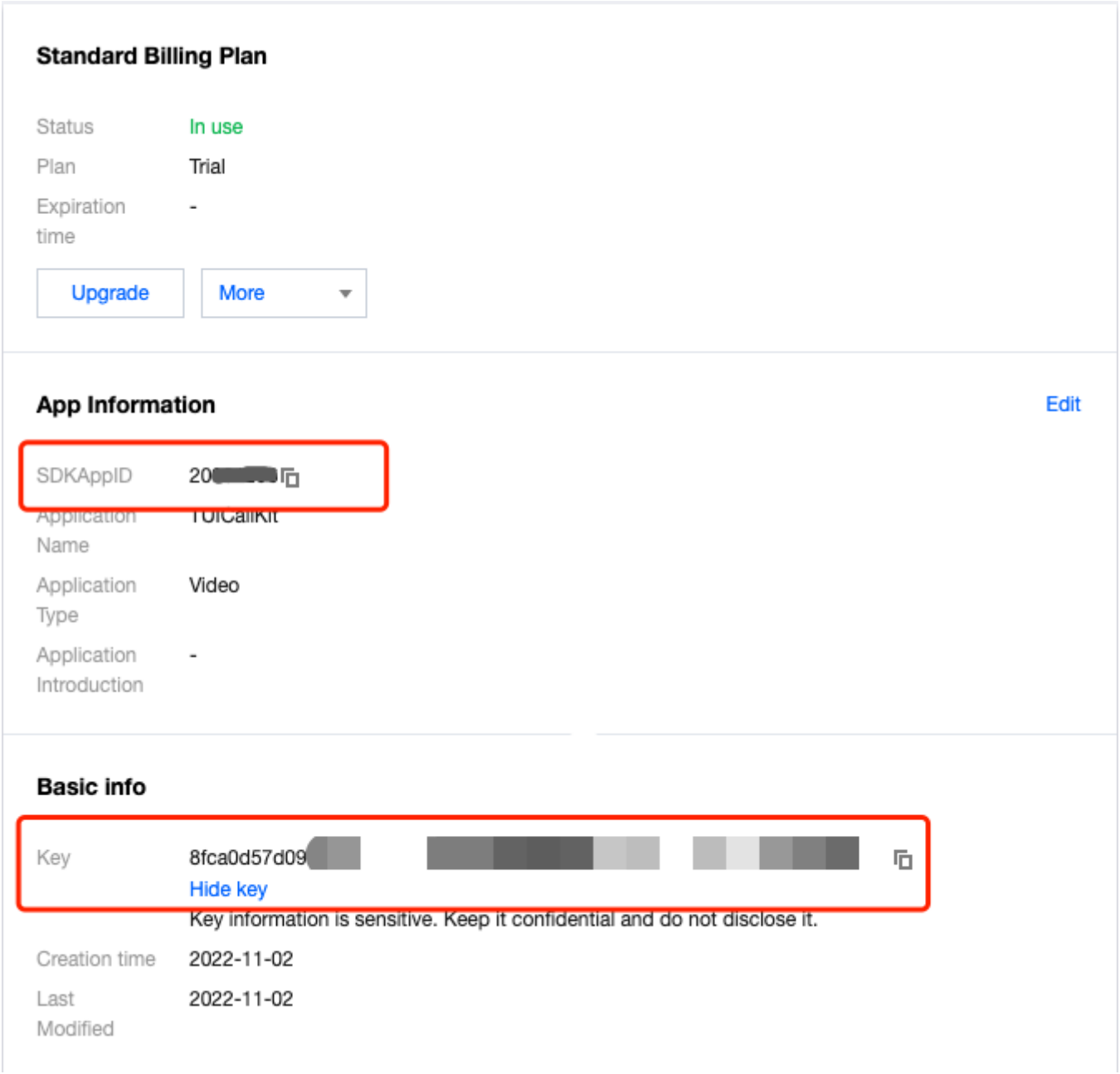
ことができます。正式アプリケーションのリリースが必要な場合は、お問い合わせください。



注意：
IMのオーディオビデオ通話機能は業務ニーズに応じて差別化した有料バージョンをご提供しています。お問い合わせいただくことで、含まれる機能をご確認の上、ご自身に合ったバージョンを選択してご購入いただけます。

3. 同じページで**SDKAppID**と**キー(SecretKey)**を見つけて記録します。この2つはその後のステップ4：TUIコンポーネントへのログインで使用します。

©2013-2022 Tencent Cloud. All rights reserved.



説明：

お知らせ：**無料体験**をクリックすると、それまでにTRTCサービスを使用したことがある一部のユーザーには、次のような内容が表示されることがあります。

```
[-100013]:TRTC service is suspended. Please check if the package balance is
0 or the Tencent Cloud accountis in arrears
```

©2013-2022 Tencent Cloud. All rights reserved.

新しいIMオーディオビデオ通話機能は、Tencent CloudのTRTCとIMという2つの基本PaaSサービスを統合したもののため、TRTCの無料利用枠（10000分）が期限切れまたは使用済みの場合、このサービスをアクティブ化しようとすると失敗します。その場合は、TRTCコンソールをクリックし、SDKAppIDに対応するアプリケーション管理ページを見つけ、図のように後払い機能をアクティブ化してから**アプリケーションを有効にする**ことで、オーディオビデオ通話機能を正常に体験することができるようになります。

ステップ2：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します。具体的な手順は次のとおりです。

1. `Podfile` ファイルに次の依存を追加します。

```
pod 'TUICallKit'
```

2. 次のコマンドを実行し、コンポーネントをインストールします。

```
pod install
```

TUICallKitの最新バージョンをインストールできない場合は、次のコマンドを実行して、ローカルのCocoaPodsリポジトリリストを更新します。

```
pod repo update
```

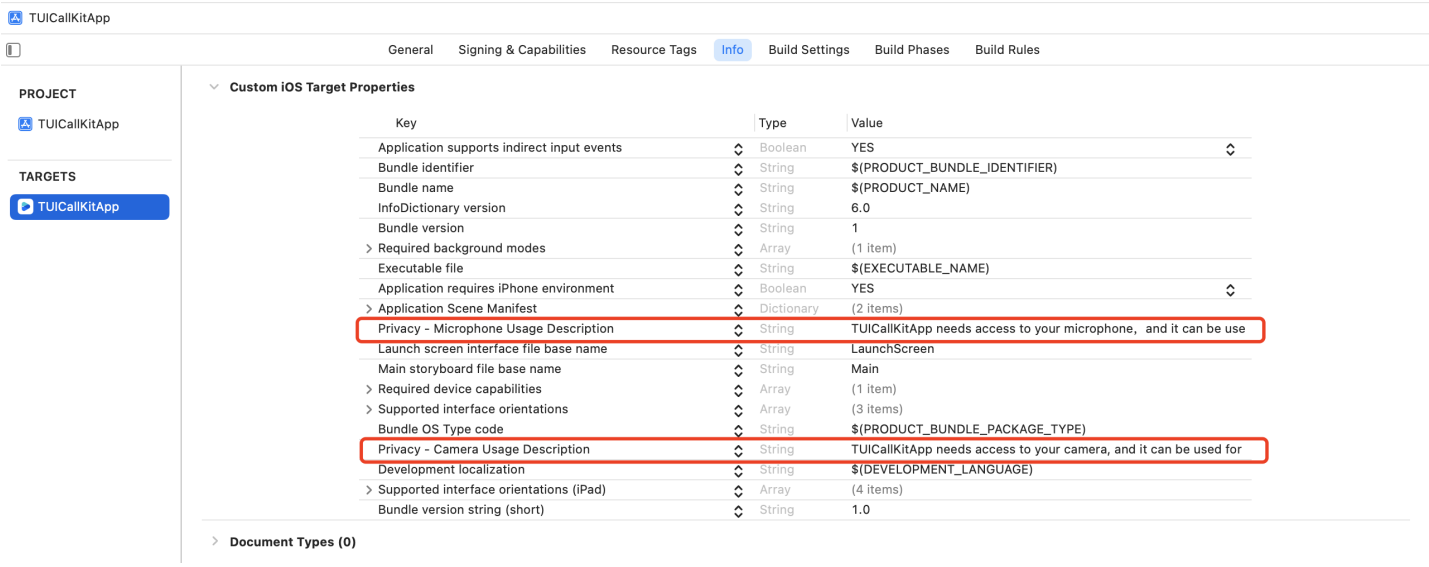
ステップ3：プロジェクト設定の完了

オーディオビデオ機能を使用するには、マイクおよびカメラの使用権限を付与する必要があります。Appの Info.plistで以下の2項を追加します。これらはシステムが権限付与ダイアログボックスをポップアップする際に表示される、マイクとカメラのメッセージにそれぞれ対応します。

```
<key>NSCameraUsageDescription</key>
<string>CallingAppはカメラへのアクセス権限が必要です。有効にした後にレコーディングしたビデオでなければ画面は出ません</string>
```

©2013-2022 Tencent Cloud. All rights reserved.

```
<key>NSMicrophoneUsageDescription</key>
<string>CallingAppはマイクへのアクセス権限が必要です。有効にした後にレコーディングしたビデオでなければ音声は出ません</string>
```



ステップ4：TUIコンポーネントへのログイン

プロジェクトに次のコードを追加します。この役割は、TUICore内の関連インターフェースを呼び出して、TUIコンポーネントへのログインを完了することです。ログインに成功しなければTUICallKitの各機能を正常に使用できないため、この手順に異常がないことが重要です。関連のパラメータが正しく設定されているかどうかを注意深く確認してください。

- Objective-C Objectivec
- Swift

```
// コンポーネントのログイン
[TUILogin login:1400000001 // ステップ1で取得したSDKAppIDに置き換えてください
userID:@"denny" // ご自身のUserIDに置き換えてください
userSig:@"xxxxxxxxxxx" // コンソールでUserSigを計算し、この位置に入力することができます
succ:^{
NSLog(@"login success");
} fail:^(int code, NSString *msg) {
NSLog(@"login failed, code: %d, error: %@", code, msg);
}
```

パラメータの説明：

ここではlogin関数内で使用するいくつかの重要パラメータについて詳しくご説明します。

©2013-2022 Tencent Cloud. All rights reserved.

- sdkAppId：ステップ1の最後の段階で取得済みです。ここでは説明を省略します。
- userId：現在のユーザーIDです。文字列タイプでは、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：ステップ3で取得したSecretKeyを使用してSDKAppID、UserIDなどの情報を暗号化し、UserSigを取得します。これは認証用の証明書であり、現在のユーザーがTRTCサービスを使用できるかどうかをTencent Cloudが識別するために用いられます。コンソールのUserSigの生成ボタンで、一時的に使用可能なUserSigを生成することができます。
  その他の情報については、UserSigの計算、使用方法をご参照ください。

注意：

- **この手順は、現在開発者から最も多くのフィードバックが寄せられる手順でもあります。よくみられる問題には次のようなものがあります。**
  - sdkAppIdの設定に誤りがある。国内サイトのSDKAppIDは一般的に140で始まる10桁の整数です。
  - userSigを誤って暗号化鍵（Secretkey）に設定している。userSigはSecretKeyを使用して、sdkAppId、userIdおよび有効期限などの情報を暗号化するためのものであり、Secretkeyは直接userSigに設定するものではありません。
  - userIdが「1」、「123」、「111」などの簡単な文字列で設定されている。**TRTCは同一のUserIDによる複数端末へのログインをサポートしていない**ため、複数人によるコラボレーション開発の場合、「1」、「123」、「111」のようなuserIdは同僚に占有されていることが多く、ログイン失敗につながりやすいです。このため、デバッグの際に、識別度の高いuserIdを設定することをお勧めします。
- GithubのサンプルコードでgenTestUserSig関数を使用してローカルでuserSigを計算すると、現在のアクセスフローをさらに速くスタートさせることができますが、この方法ではSecretKeyがAppのコード内で開示されてしまい、その後のアップグレードおよびSecretKeyの保護にとって不利益になります。そのため、userSigの計算ロジックはサーバーに置いて実行し、またAppがTUICallKitコンポーネントを使用するたびに、リアルタイムに計算したuserSigをサーバーにリクエストするようにすることを強く推奨します。

ステップ5：通話する

1対1ビデオ通話

TUICallKitのcall関数を呼び出し、 通話タイプと着呼者のuserIdを指定すると、音声またはビデオ通話を開始することができます。

- Objective-C Objectivec
- Swift

©2013-2022 Tencent Cloud. All rights reserved.

```
// 1対1ビデオ通話を開始します(userIdはmikeとします)
[[TUICallKit createInstance] call:@"mike" callMediaType:TUICallMediaTypeVideo];
```

パラメータ	タイプ	意味
userId	String	ターゲットユーザーのUserID： `"mike"`
callMediaType	TUICallMediaType	通話のメディアタイプ。例： `TUICallMediaTypeVideo`

グループ内ビデオ通話

TUICallKitのgroupCall関数を呼び出し、 通話タイプと着呼者のUserIDリストを指定すると、グループ内で音声またはビデオ通話を開始することができます。

- Objective-C Objectivec
- Swift

```
[[TUICallKit createInstance] groupCall:@"12345678" userIdList:@[@"denny", @"mike", @"tommy"] callMediaType:TUICallMediaTypeVideo];
```

パラメータ	タイプ	意味
groupId	String	グループID。例： `@"12345678"`
userIdList	Array	ターゲットユーザーのuserIdリスト。例： `@[@"denny", @"mike", @"tommy"]`
callMediaType	TUICallMediaType	通話のメディアタイプ。例： `TUICallMediaTypeVideo`

説明：

- グループの作成の詳細についてはIMグループ管理をご参照ください。もしくはIM TUIKitを直接使用し、チャット、通話などのシナリオをワンストップで統合することもできます。
- TUICallKitでは現在、グループ以外の多人数ビデオ通話はサポートしていません。もし必要な場合は、colleenyu@tencent.comまでぜひフィードバックをお寄せください。

ステップ6：通話に応答する

©2013-2022 Tencent Cloud. All rights reserved.

ステップ4が完了し、通話リクエストを受信すると、TUICallKitコンポーネントは対応する応答画面を自動的に起動します。

ステップ7：その他の特徴

1、ニックネーム&プロフィール画像の設定

カスタムニックネームまたはプロフィール画像を設定したい場合は、次のインターフェースを使用して更新できます。

- Objective-C Objectivec
- Swift

```
[[TUICallKit createInstance] setSelfInfo:@"ニックネーム" avatar:@"プロフィール画像URL" succ:^{

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

}];
```

> 注意：
> ユーザーのプライバシー上の制限により、フレンド以外との通話では、呼ばれるニックネームとプロフィール画像の更新が遅れる可能性があります。一度通話が成功するとスムーズに更新されるようになります。

2、オフライン通知

上記の手順が完了すると、オーディオビデオ通話の発信と接続が実現できますが、業務上「Appのプロセスが強制終了された後」または「APPがバックエンドに戻った後」もオーディオビデオ通話リクエストを正常に受信できる必要がある場合は、オフライン通知機能を追加する必要があります。詳細については**オフライン通知（iOS）**をご参照ください。

3、フローティングウィンドウ機能

業務上、フローティングウィンドウ機能を有効にする必要がある場合は、TUICallKitコンポーネントの初期化の際に次のインターフェースを呼び出してこの機能を有効化することができます。

- Objective-C Objectivec
- Swift

©2013-2022 Tencent Cloud. All rights reserved.

```
[[TUICallKit createInstance] enableFloatWindow:YES];
```

4. 通話ステータスの監視

業務上、通話の開始、終了、通話中のネットワーク品質などの**通話ステータスの監視**が必要な場合は、次のイベントを監視することができます。

- Objective-C Objectivec
- Swift

```
[[TUICallEngine createInstance] addObserver:self];

- (void)onCallBegin:(TUIRoomId *)roomId callMediaType:(TUICallMediaType)callMediaType callRole:(TUICallRole)callRole {


}
- (void)onCallEnd:(TUIRoomId *)roomId callMediaType:(TUICallMediaType)callMediaType callRole:(TUICallRole)callRole totalTime:(float)totalTime {


}
- (void)onUserNetworkQualityChanged:(NSArray<tuinetworkqualityinfo *=""> *)networkQualityList {


}
```

5、カスタム着信音

着信音をカスタマイズしたい場合は、次のインターフェースによって設定できます。

- Objective-C Objectivec
- Swift

```
[[TUICallKit createInstance] setCallingBell:filePath];
```

よくあるご質問

「The package you purchased does not support this ability」というエラーが表示されました。

©2013-2022 Tencent Cloud. All rights reserved.

上記のエラーが表示された場合は、現在のアプリケーションのオーディオビデオ通話機能パッケージが期限切れまたはアクティブ化されていないことを表します。ステップ1を参照してオーディオビデオ通話機能を取得またはアクティブ化し、引き続きTUICallKitコンポーネントをご利用ください。

ご意見とフィードバック

ご利用中にご提案やご意見などがございましたら、colleenyu@tencent.comまでフィードバックをお願いいたします。

©2013-2022 Tencent Cloud. All rights reserved.

Web

最終更新日：：2023-02-17 16:32:51

TUICallKitコンポーネントのクイックアクセス

ここでは、最短の時間でTUICallKitコンポーネントの統合を完了する方法についてご説明します。このドキュメントに沿って操作することで、次の重要手順を完了し、最終的にUIを完備したビデオ通話機能を手にすることができます。
コンポーネントの効果を体験したい場合は、TUICallKit demoクイックスタートをご参照ください。
アクセス前にデスクトップブラウザがオーディオビデオサービスをサポートしているかどうかを確認してください。詳しい要件は環境要件をご参照ください。

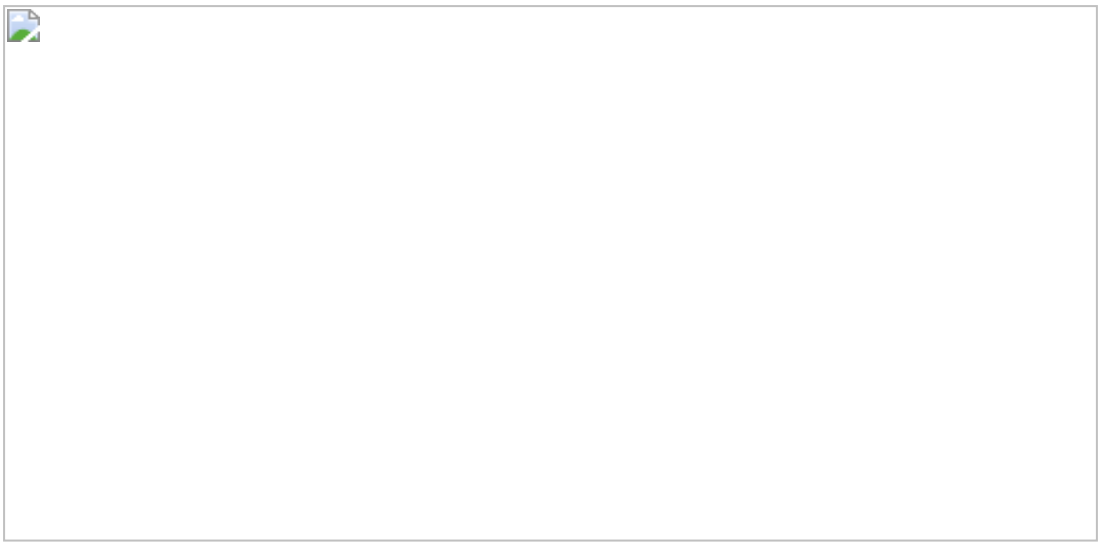
目次

ステップ1：サービスをアクティブにする
ステップ2：TUICallKitコンポーネントをインポートする
ステップ3：TUICallKitコンポーネントを呼び出す
その他のドキュメント
よくあるご質問
1. どのようにUserSigを生成すればよいですか
2. viteのインポートの問題
3. 環境要件
(1) ブラウザのバージョン要件
(2) ネットワーク環境の要件
(3) ウェブサイトドメイン名プロトコルの要件

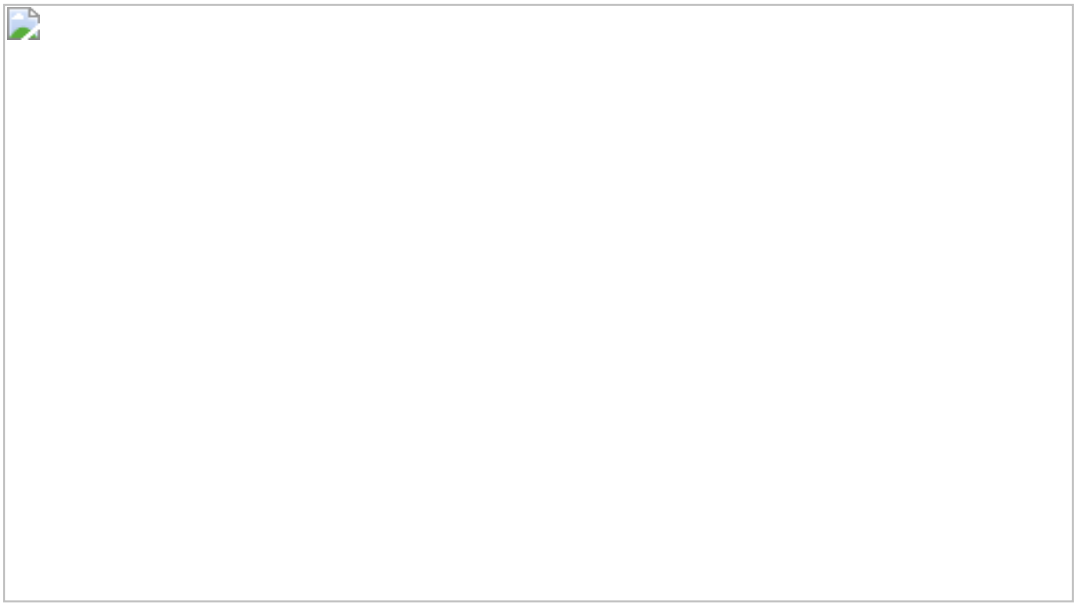
ステップ1：サービスのアクティブ化

TUICallKitはTencent CloudのIMと、TRTCという2つの有料PaaSサービスをベースに構築したオーディオビデオ通信コンポーネントです。以下の手順で関連のサービスをアクティブ化し、7日間の無料トライアルサービスを体験することができます。
1. IMコンソールにログインし、**新しいアプリケーションの作成**をクリックし、ポップアップしたダイアログボックスにアプリケーション名を入力して**OK**をクリックします。

©2013-2022 Tencent Cloud. All rights reserved.

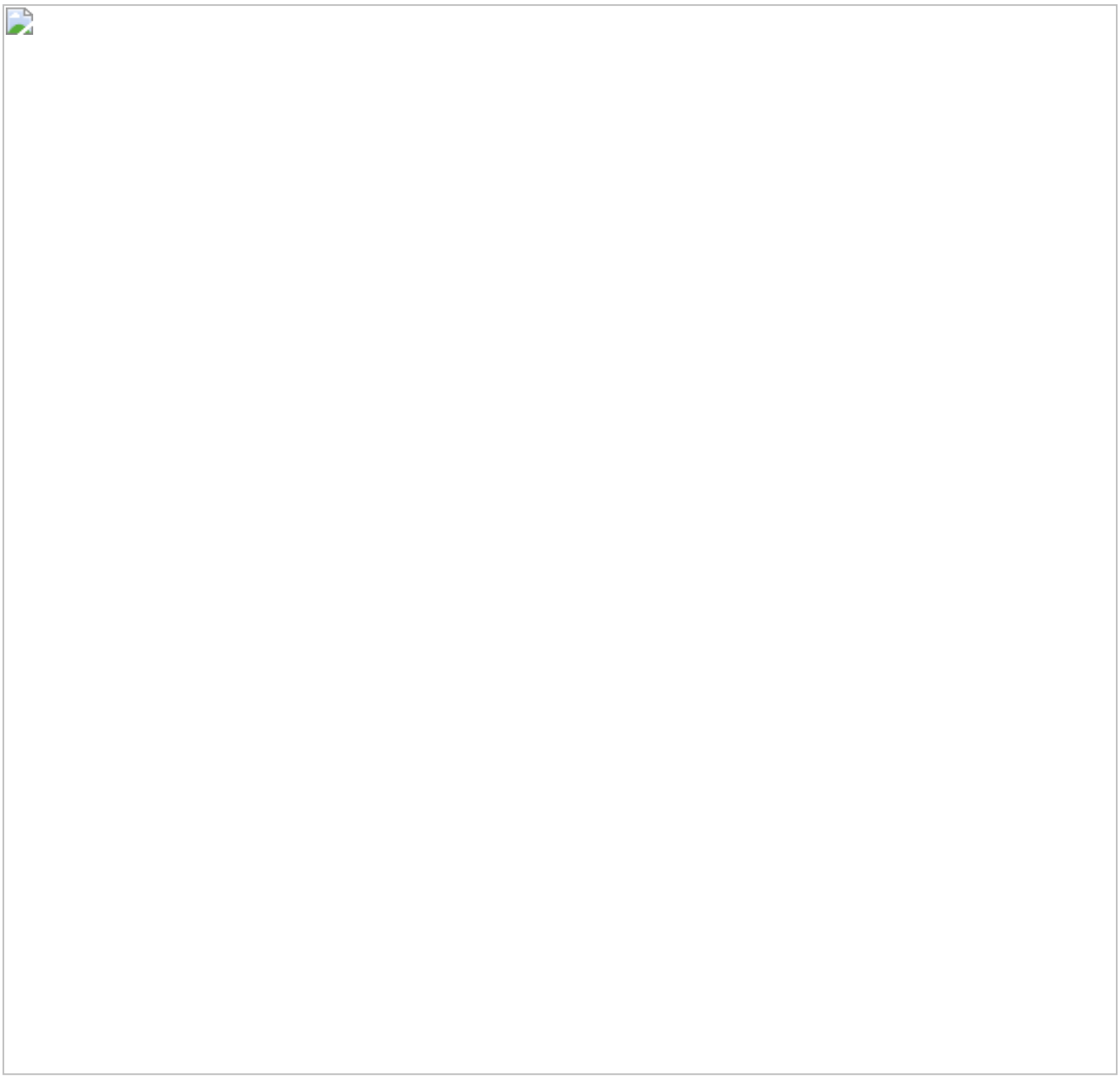


2. 作成したアプリケーションをクリックし、**基本設定**ページに進み、ページ右下隅の**TRTCサービスのアクティブ化**機能エリアで**無料体験**をクリックすると、TUICallKitの7日間無料トライアルサービスをアクティブ化することができます。



3. 同じページで**SDKAppID**と**キー**を見つけて記録します。これらは後ほど使用します。

©2013-2022 Tencent Cloud. All rights reserved.



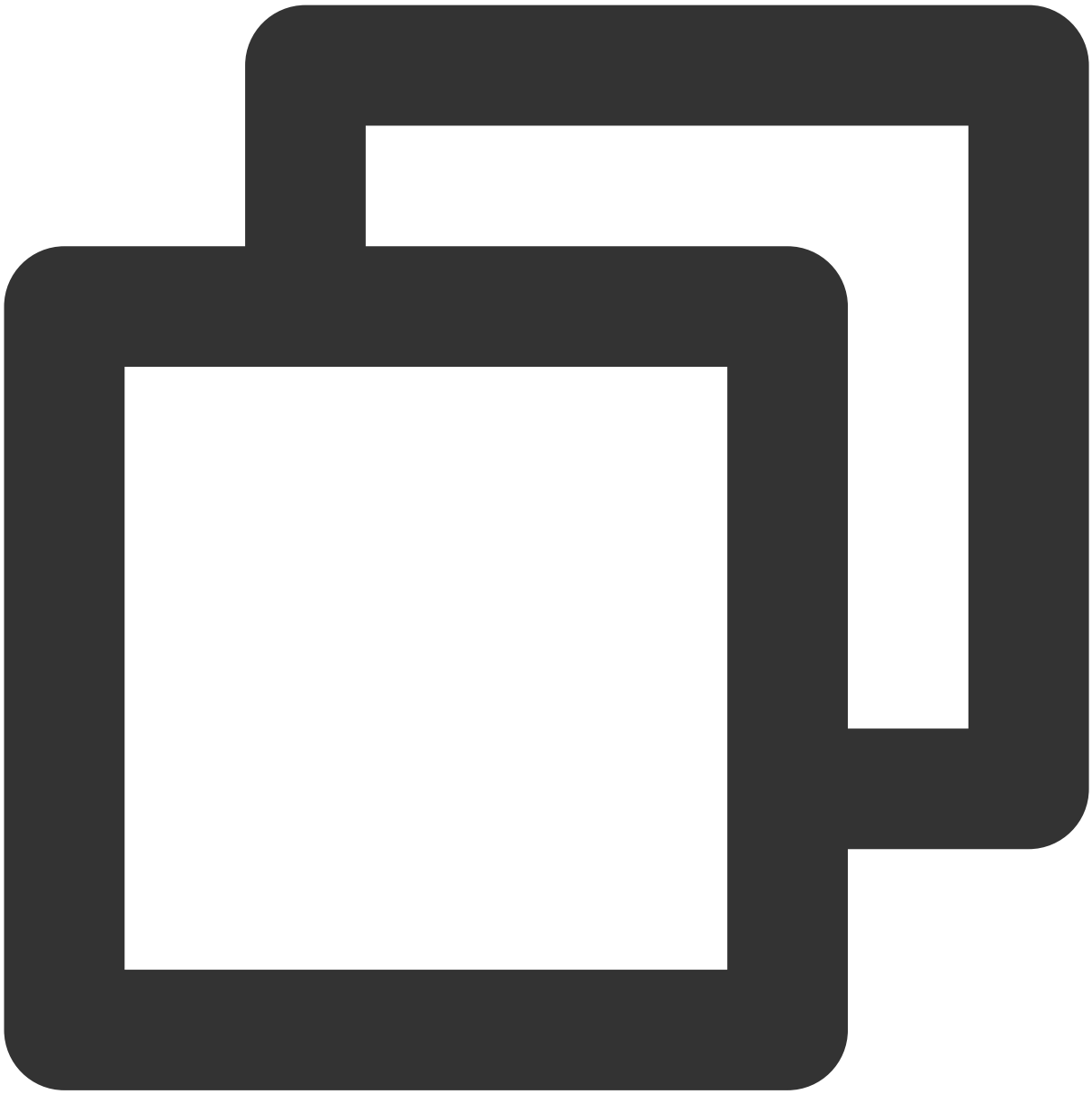
SDKAppID：IMのアプリケーションIDです。業務の分離、すなわち異なるSDKAppIDの通話が互いに接続できないようにするために用いられます。
Secretkey：IMのアプリケーションキーです。SDKAppIDとペアで使用する必要があり、IMサービスを正当に使用するための認証用証明書UserSigの発行に用いられます。このKeyは次のステップ5で使用します。

ステップ2：TUICallKitコンポーネントのインポート

方式1：npm統合

©2013-2022 Tencent Cloud. All rights reserved.

npm方式でTUICallKitコンポーネントをダウンロードします。その後の拡張をしやすくするために、TUICallKitコンポーネントをプロジェクトの `src/components/` ディレクトリにコピーすることをお勧めします。



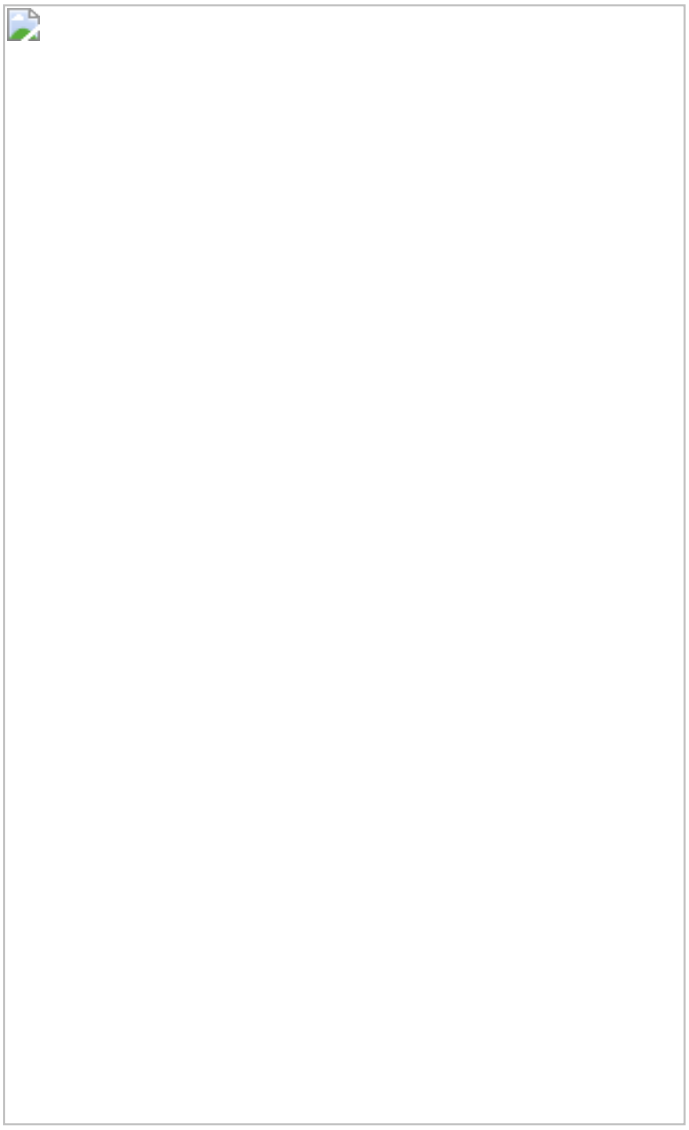
```
# macOS
yarn add @tencentcloud/call-uikit-vue  # yarnがインストールされていない場合、まず以下を実行
mkdir -p ./src/components/TUICallKit/Web && cp -r ./node_modules/@tencentcloud/call

# windows
yarn add @tencentcloud/call-uikit-vue  # yarnがインストールされていない場合、まず以下を実行
xcopy .\\node_modules\\@tencentcloud\\call-uikit-vue  .\\src\\components\\TUICallKi
```

©2013-2022 Tencent Cloud. All rights reserved.

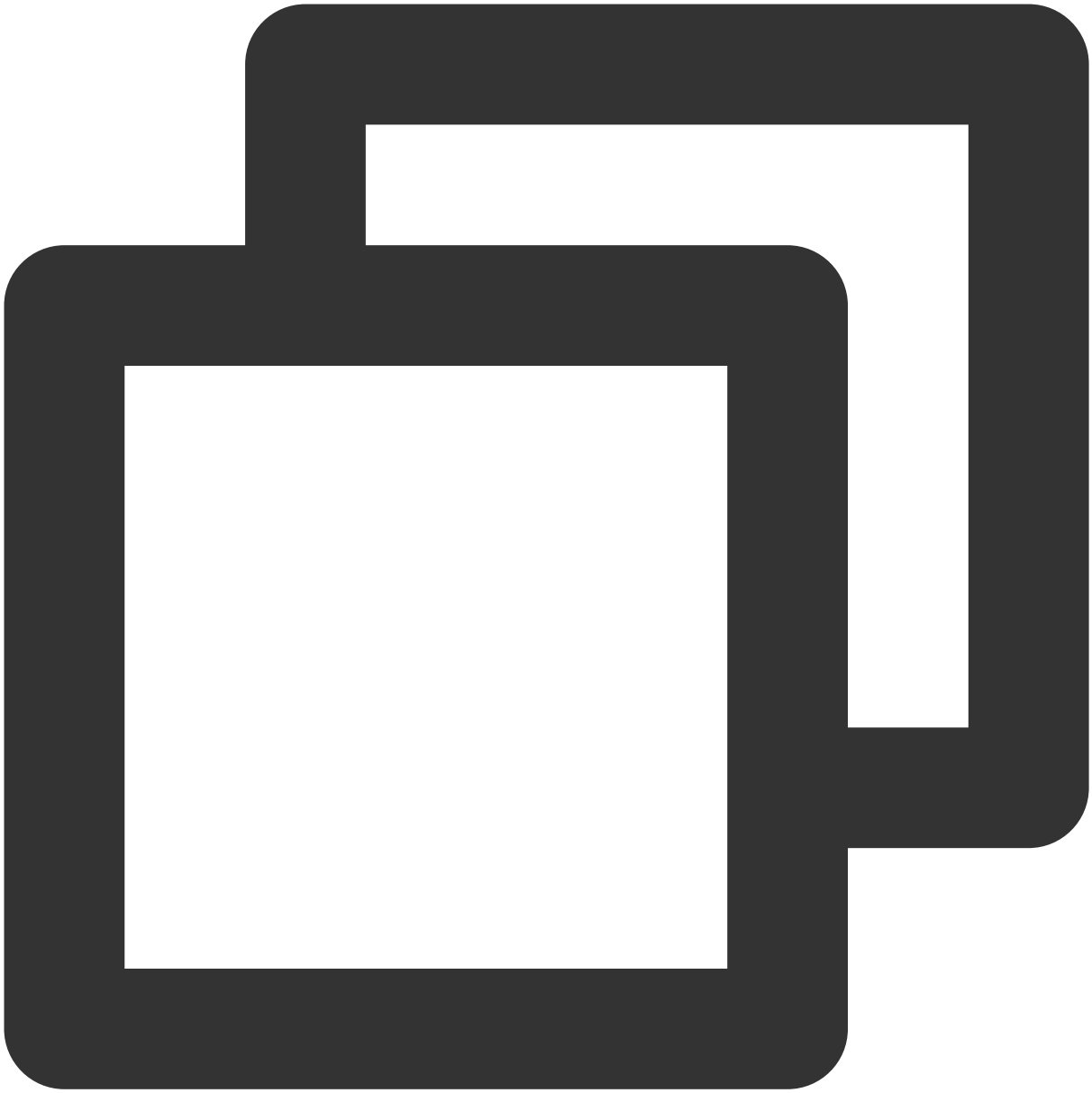
方式2：ソースコードの統合

1. GitHubからTUICallKitソースコードをダウンロードします。 `TUICallKit/Web` フォルダをコピーして自身のプロジェクトの `src/components` フォルダに置きます。例えば次のようになります。



2. このフォルダに進み、実行に必要な依存をインストールします

©2013-2022 Tencent Cloud. All rights reserved.

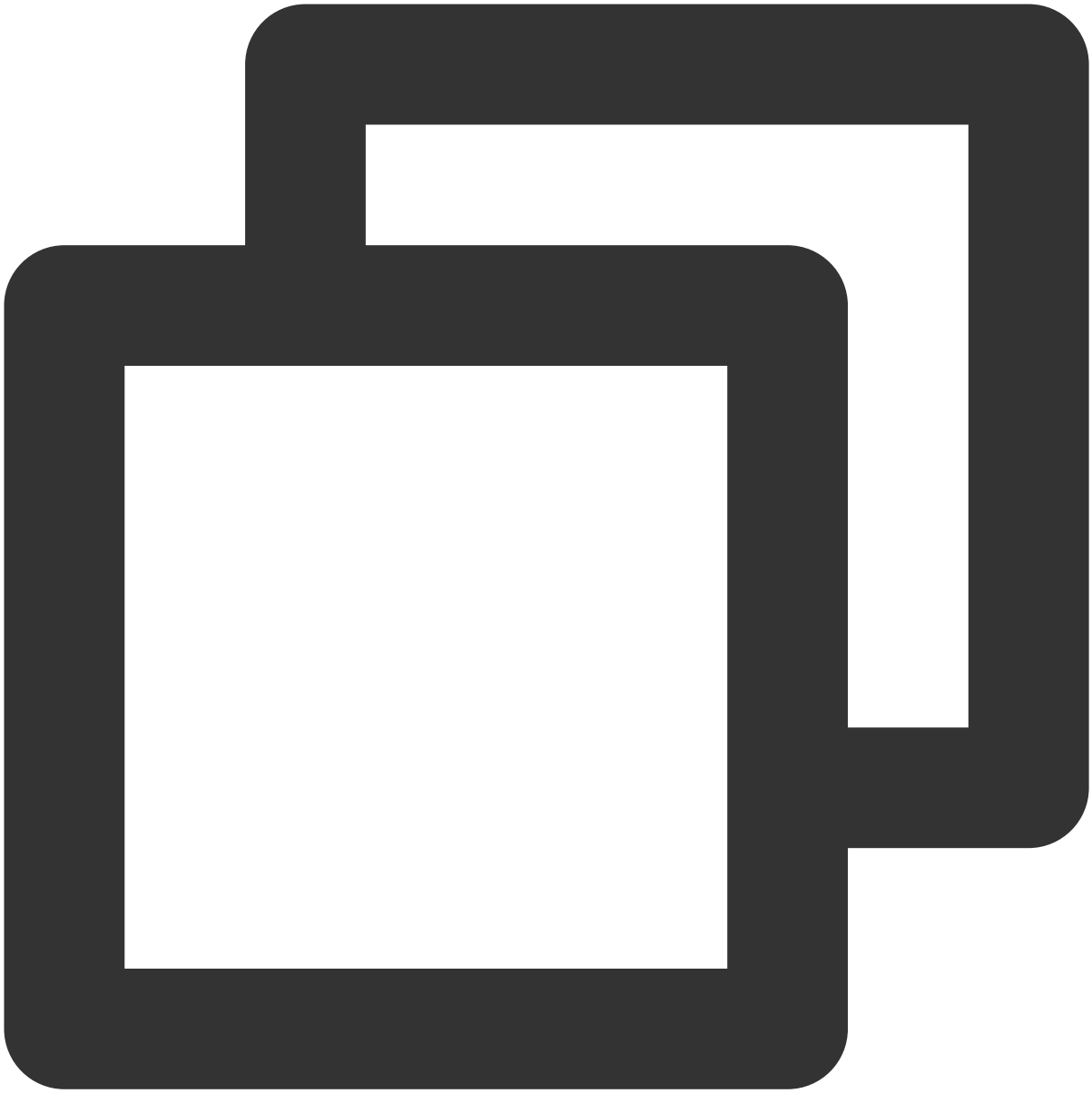


```
cd ./src/components/TUICallKit/Web
yarn          # yarnをインストールしていない場合、先に以下を実行できます。 npm install -g yar
```

ステップ3：TUICallKitコンポーネントの呼び出し

表示したいページでTUICallKitのコンポーネントを呼び出すと、通話ページを表示することができます。
1. 次のようにTUICallKit UIをインポートします。

©2013-2022 Tencent Cloud. All rights reserved.



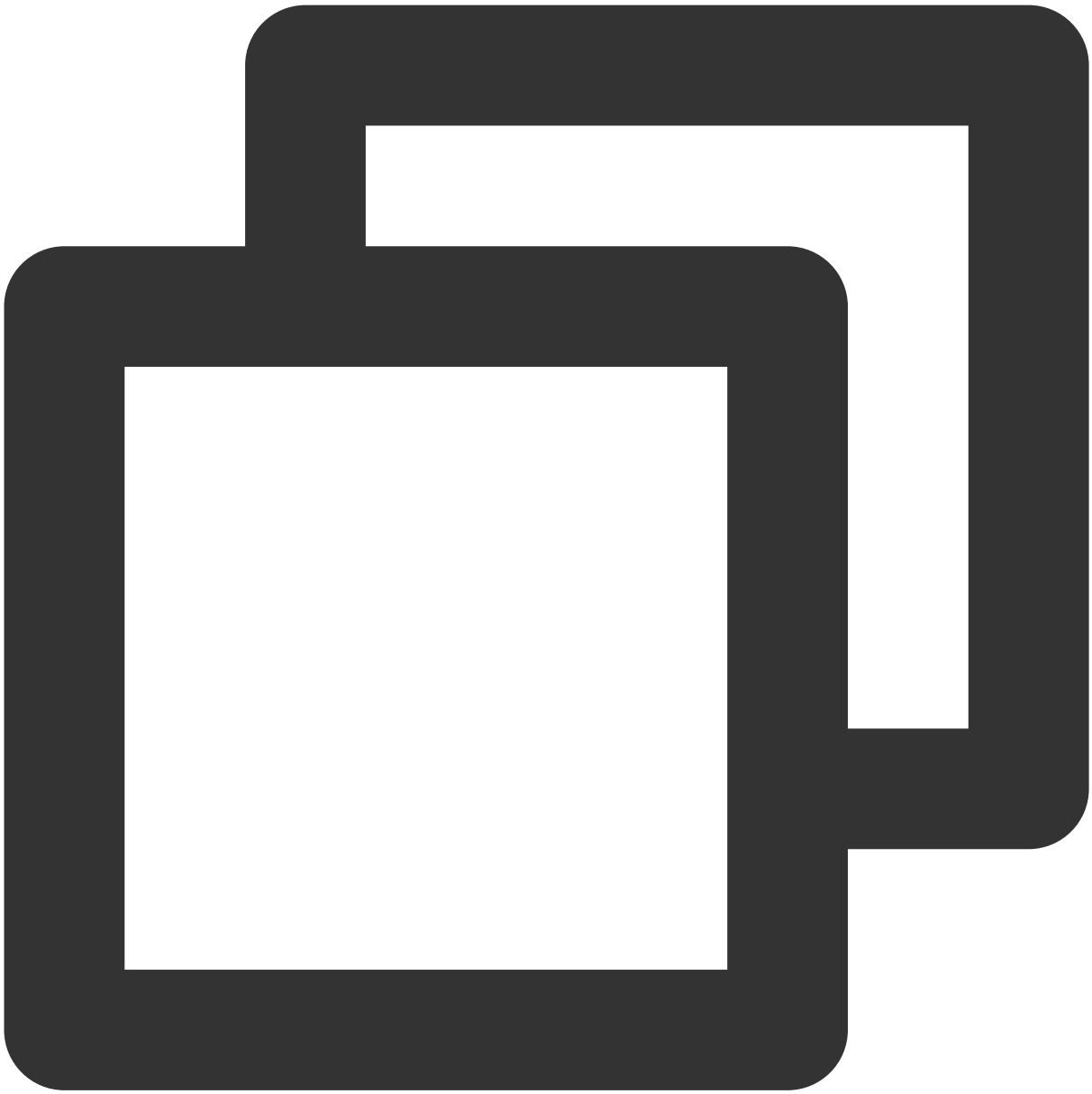
```
<script lang="ts" setup>
import { TUICallKit } from "./components/TUICallKit/Web";
</script>

<template>
  <TUICallKit />
</template>
```

2. ユーザーログインし、電話をかけます

©2013-2022 Tencent Cloud. All rights reserved.

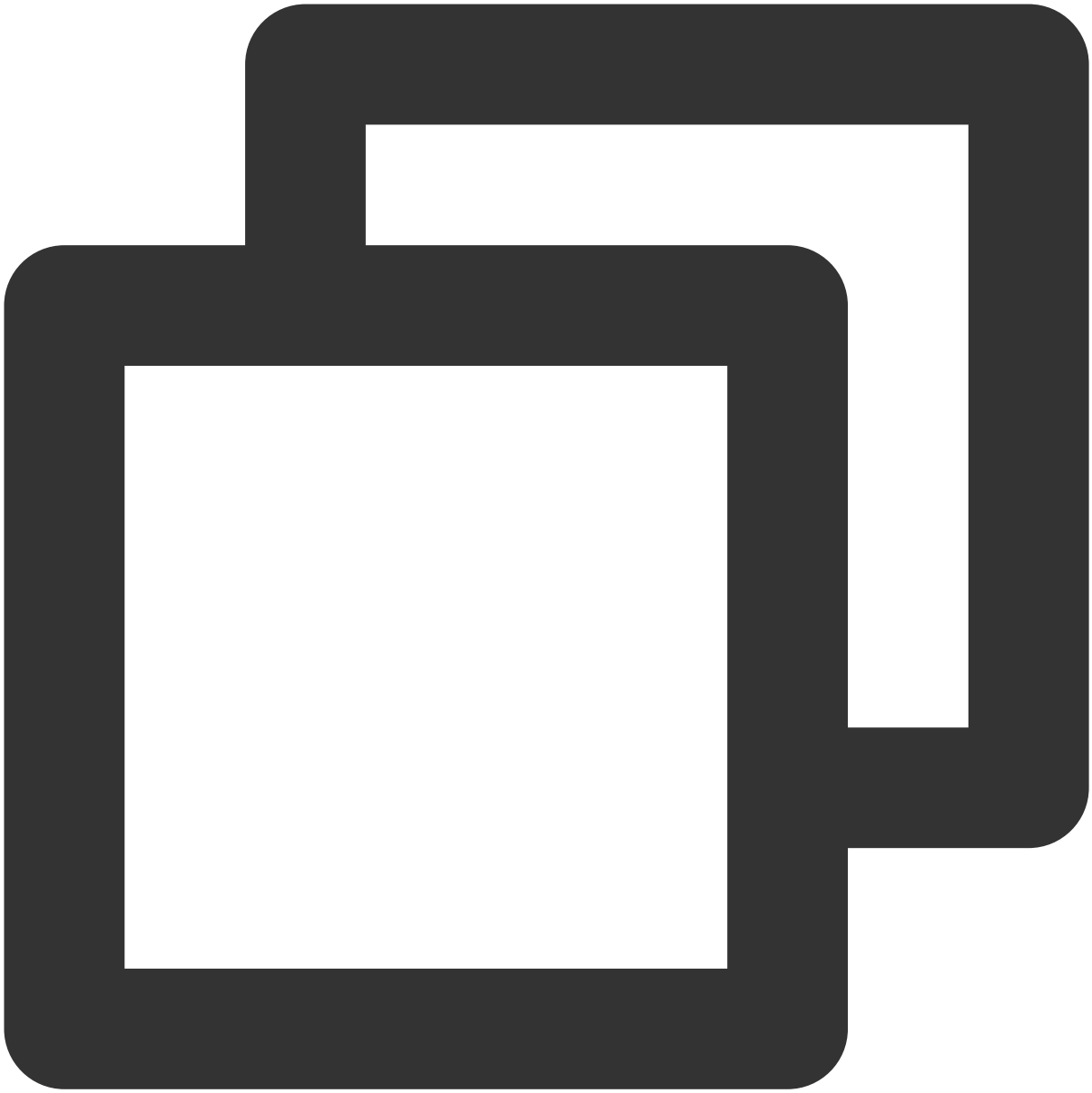
2.1 TUIKitのキットをすでに使用している場合は下記のコードをインポートし、TUICallKitがプラグインであることを宣言する必要があります。使用していない場合、このステップは2.2までスキップすることができます。



```
import { TUICallKit } from './components/TUICallKit/Web';
TUIKit.use(TUICallKit);
```

2.2 電話をかけるには初期化コンポーネントを保持する必要があります

©2013-2022 Tencent Cloud. All rights reserved.



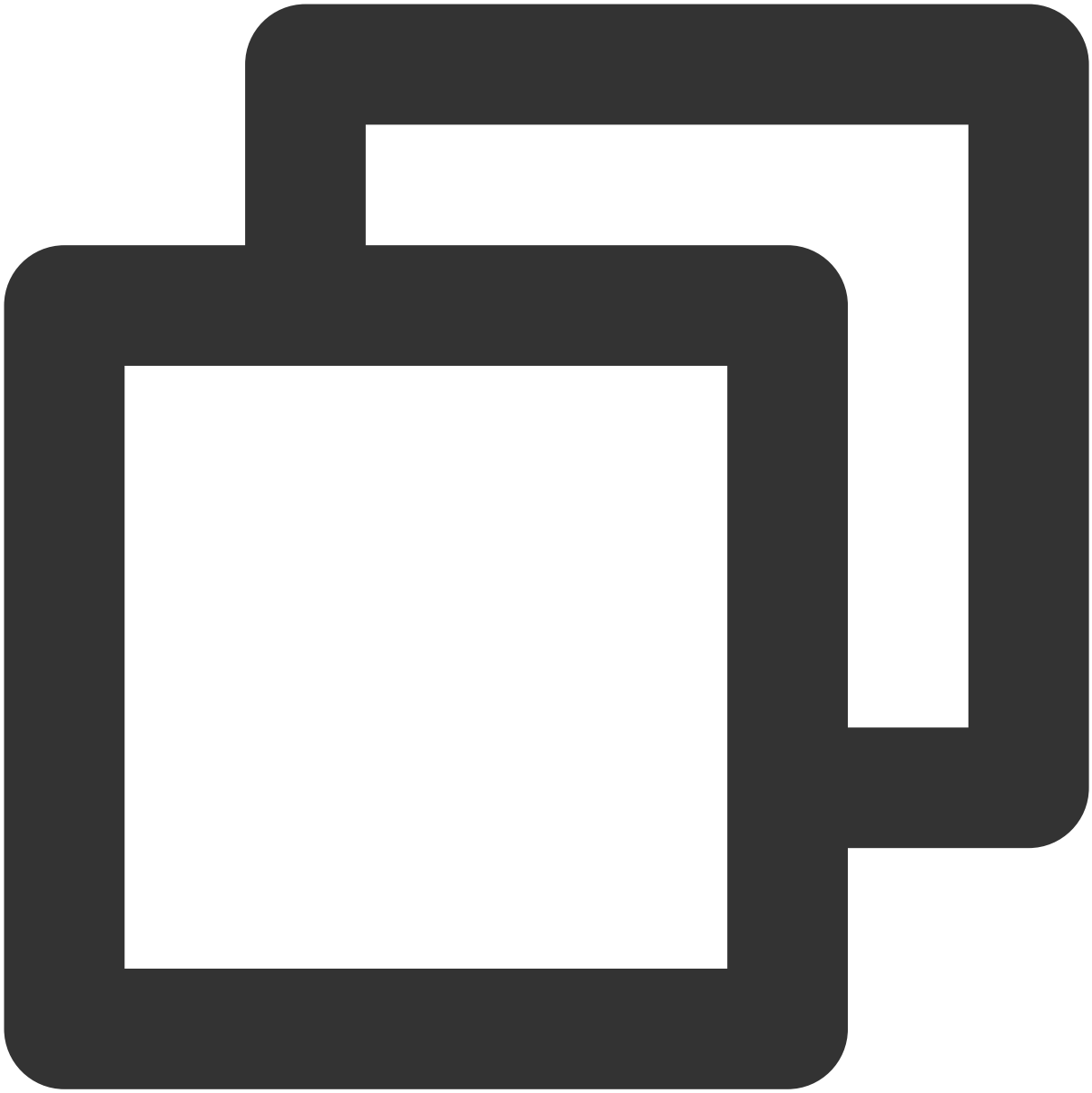
```
import { TUICallKitServer } from './components/TUICallKit/Web';
TUICallKitServer.init({ SDKAppID, userID, userSig });
```

explain

SDKAppID、 SecretKeyの取得はステップ1をご参照ください

userSigは**一時的に** `GenerateTestUserSig.js` の `genTestUserSig(userID)` 関数を使用して計算することができます。例えば次のようになります。

©2013-2022 Tencent Cloud. All rights reserved.



```
import * as GenerateTestUserSig from "./components/TUICallKit/Web/demos/basic/publi
const { userSig } = GenerateTestUserSig.genTestUserSig(userID, SDKAppID, SecretKey)
```

viteを起動ツールとして使用する場合、viteのインポートの問題に注意してください

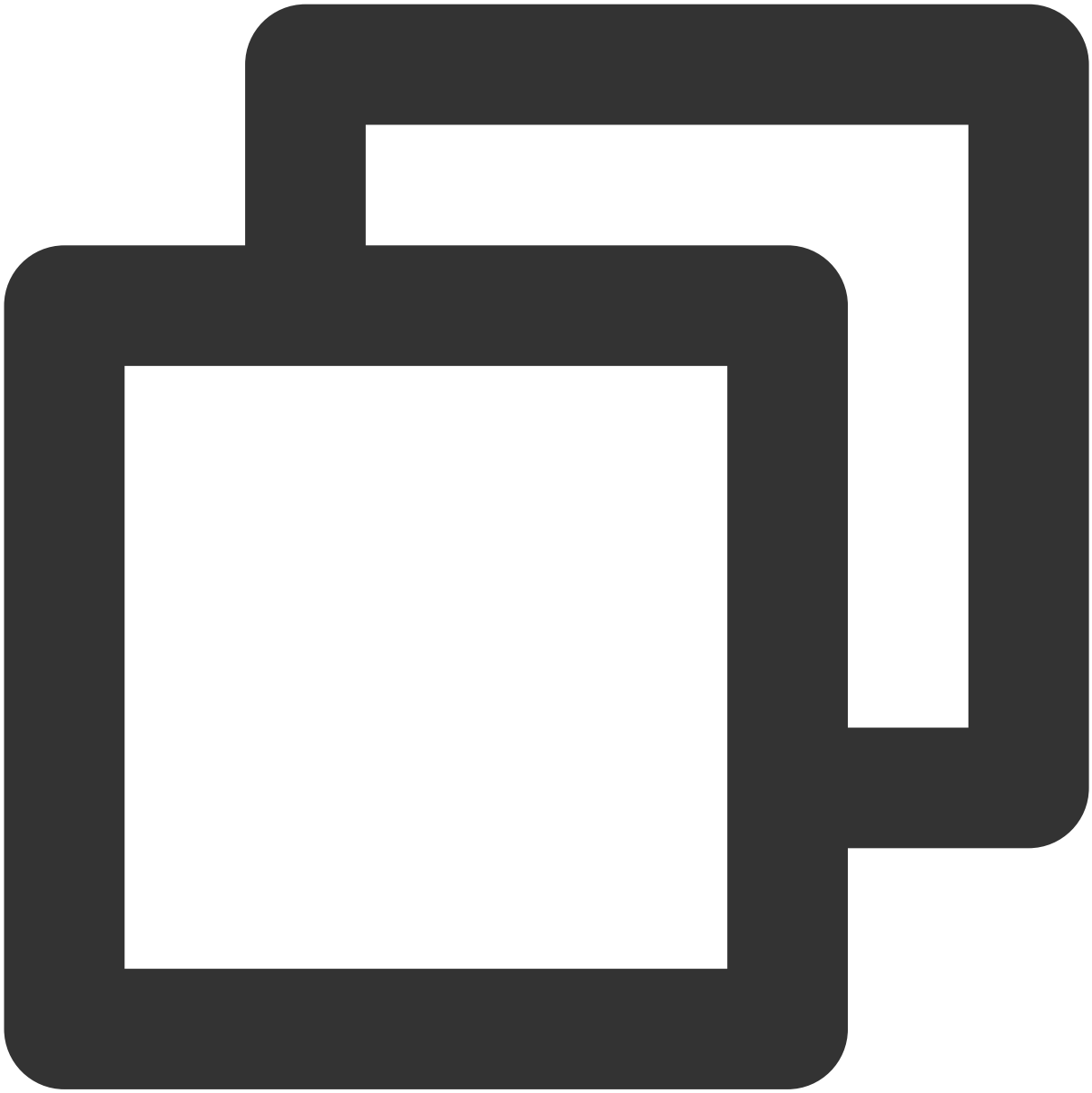
notice

ここで言及するUserSigの取得方法は、フロントエンドコードに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、ローカルのクイックスタート機能の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

©2013-2022 Tencent Cloud. All rights reserved.

ーフェース向けに提供することです。 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。詳細については、サーバーでのUserSig新規作成をご参照ください。

2.3 電話をかけたい場所で、次を実行します。



```
import { TUICallKitServer } from './components/TUICallKit/Web';
TUICallKitServer.call({ userID: "123", type: 2 }); // 2人通話
TUICallKitServer.groupCall({ userIDList: ["xxx"], groupID: "xxx", type: 2 }); // 多
```

その後すぐに最初の電話をかけることができます。より詳細なインターフェースパラメータについては、インターフェースドキュメントをご参照ください

©2013-2022 Tencent Cloud. All rights reserved.

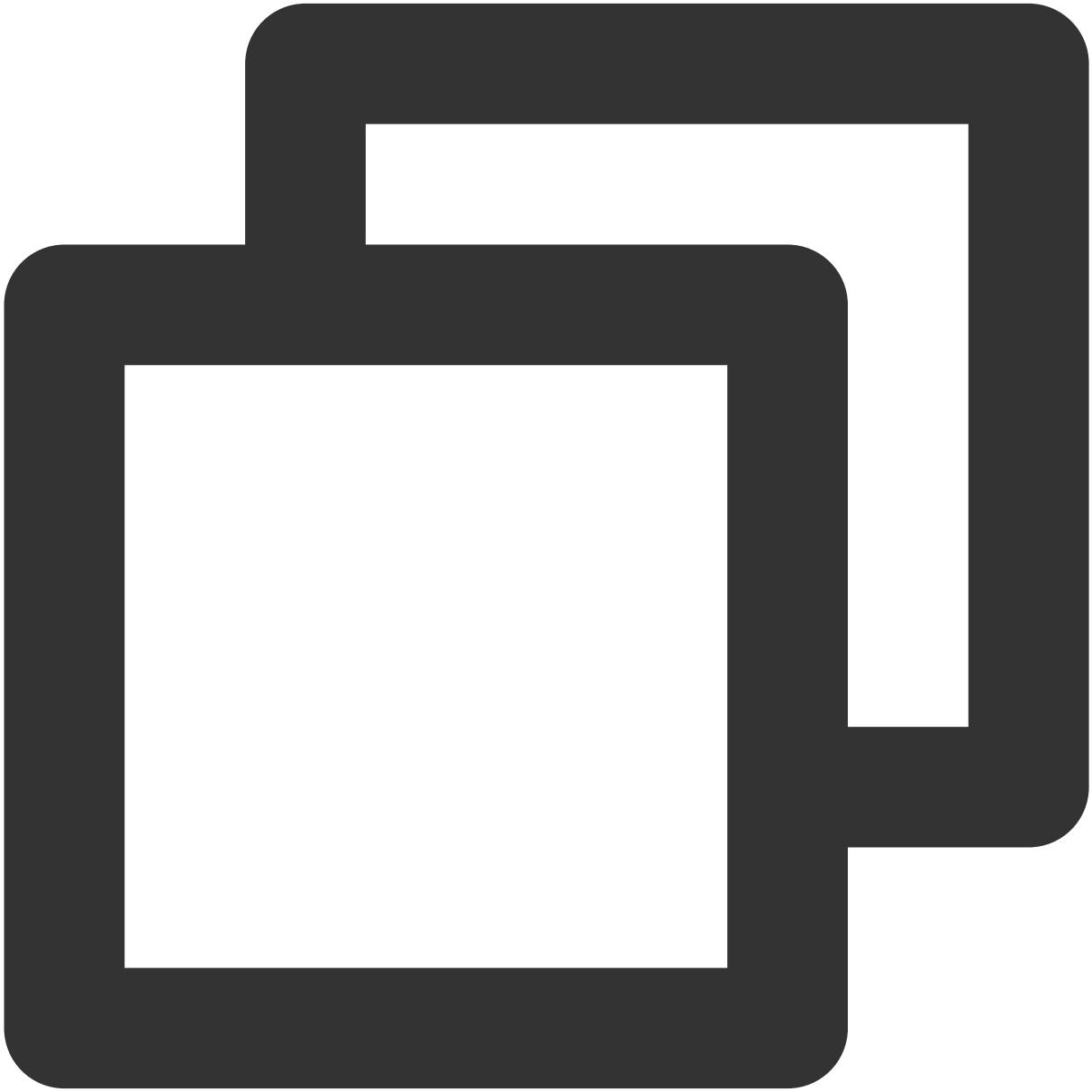
3. 発展インターフェース

このコンポーネントでは `beforeCalling` および `afterCalling` という2つのコールバックを提供しており、業務側に現在の通話状態を通知するために用いられます。

`beforeCalling` : 通話前に実行します

`afterCalling` : 通話後に実行します

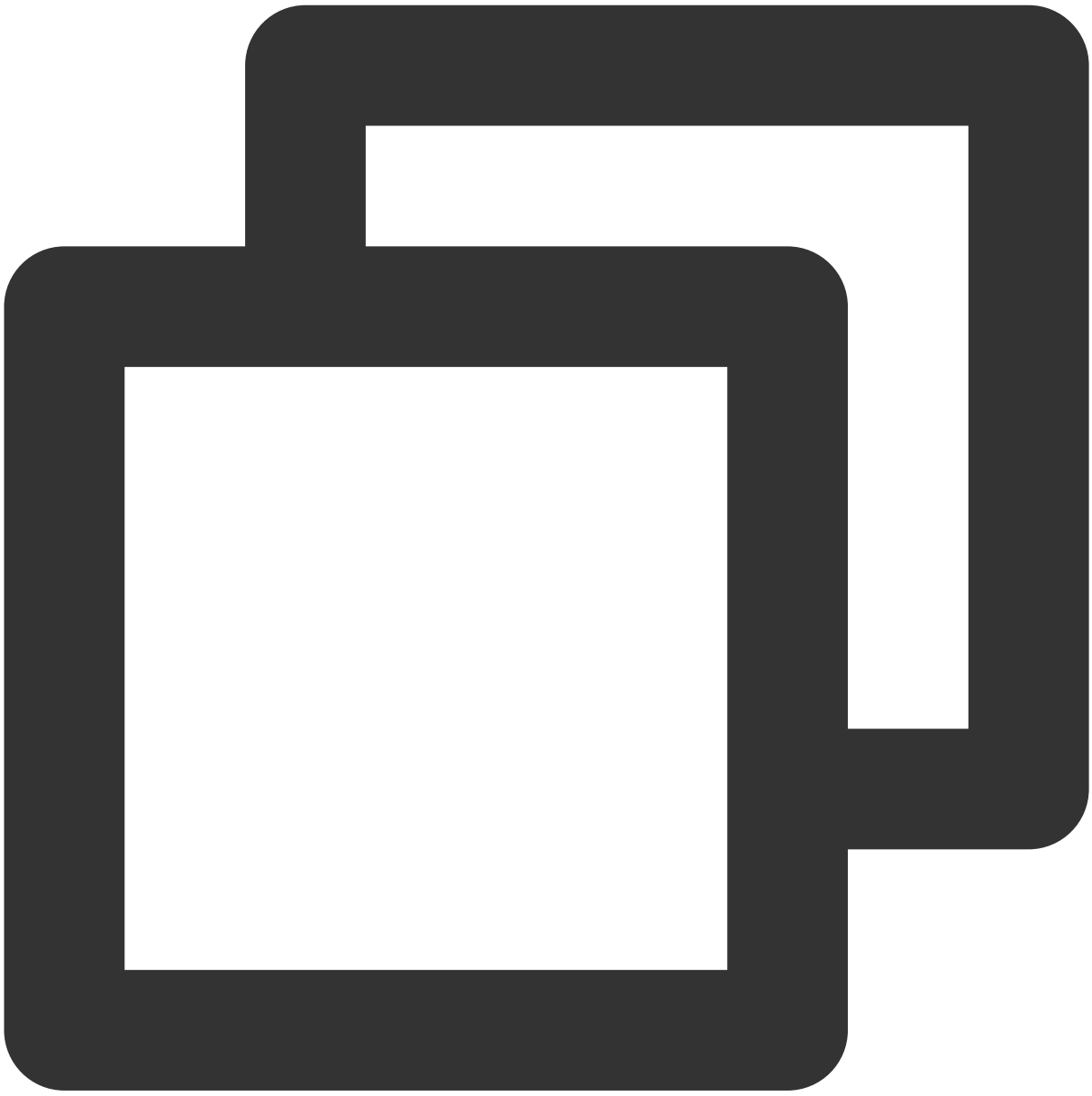
例えばサンプルdemoのように、 `<TUICallKit />` コンポーネントの表示と折りたたみに用いることができます。



```
function beforeCalling() {
  console.log("通話前にこの関数を実行");
```

©2013-2022 Tencent Cloud. All rights reserved.

```
}
function afterCalling() {
  console.log("通話後にこの関数を実行");
}
```



```
<TUICallKit :beforeCalling="beforeCalling" :afterCalling="afterCalling"/>
```

その他のドキュメント

©2013-2022 Tencent Cloud. All rights reserved.

TUICallKit API
TUICallKit demoクイックスタート
TUICallKitインターフェースのカスタマイズガイド
TUICallKit (Web) に関するよくあるご質問

よくあるご質問

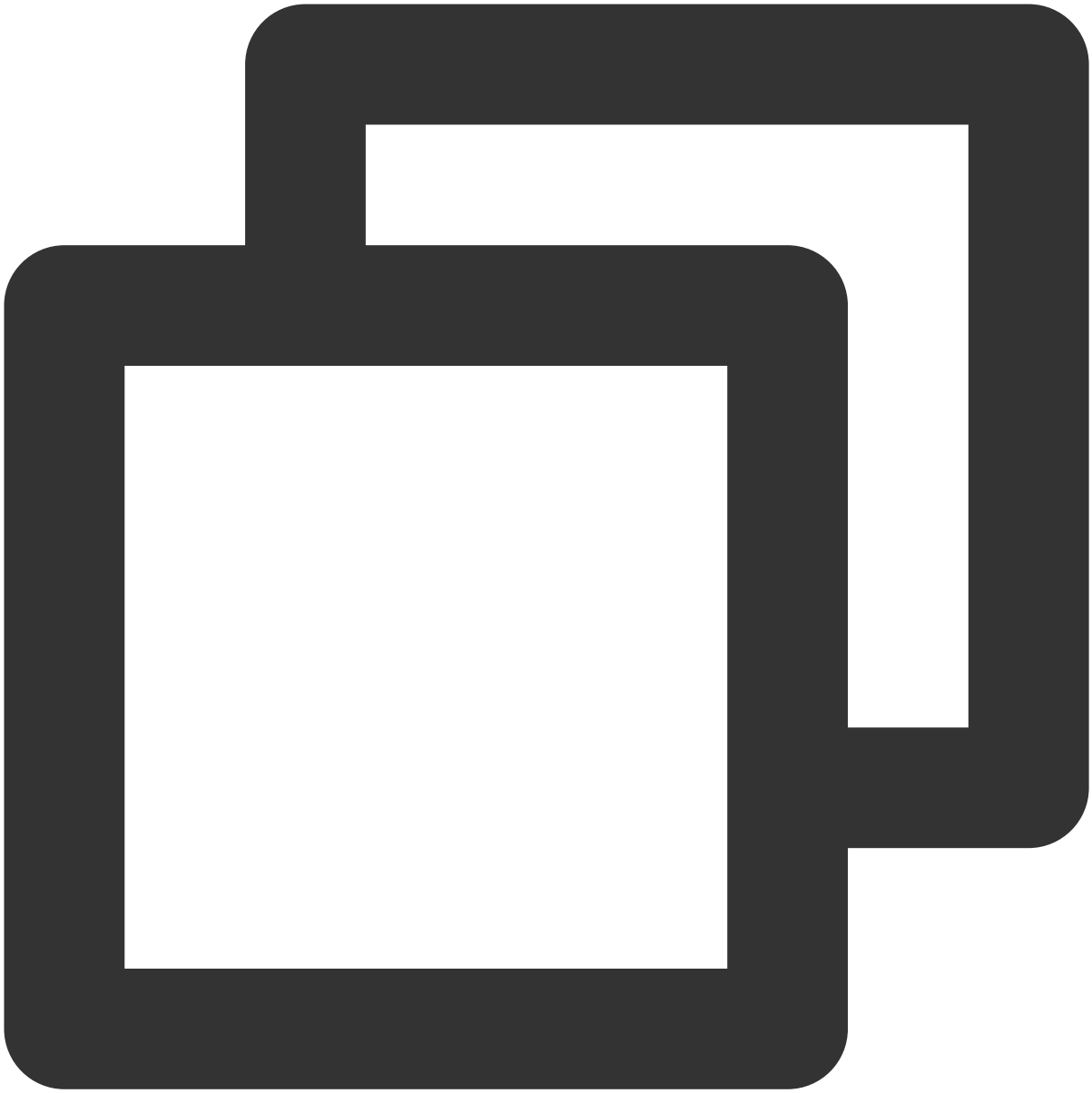
1. UserSigはどのように生成しますか。

UserSigの発行方法は、UserSigの計算コードをサーバーに統合し、プロジェクトのインターフェース向けに提供する方法となります。UserSigが必要な時は、プロジェクト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。詳細については、サーバーでのUserSig新規作成をご参照ください。

2. viteのインポートの問題

プロジェクトをviteで作成した場合は、ファイルパッケージの違いにより、 `index.html` に `lib-generate-test-usersig.min.js` をインポートする必要があります

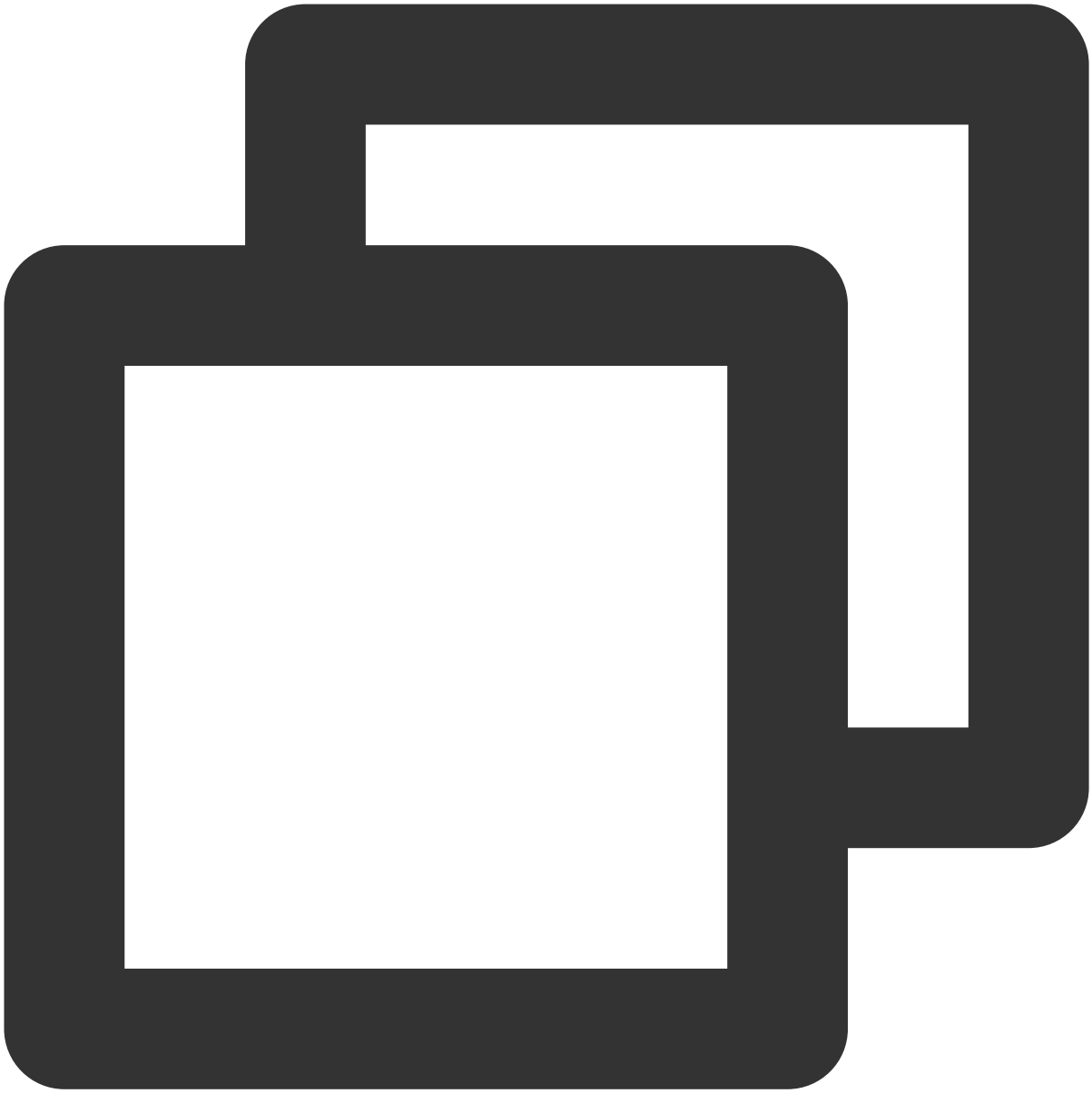
©2013-2022 Tencent Cloud. All rights reserved.



```
// index.html
<script src="./src/components/TUICallKit/Web/demos/basic/public/debug/lib-generate-
```

また、元の `GenerateTestUserSig.js` にimportしたメソッドのアノテーションを行います

©2013-2022 Tencent Cloud. All rights reserved.



```
// import * as LibGenerateTestUserSig from './lib-generate-test-usersig.min.js'
```

3. 環境要件

(1) ブラウザのバージョン要件

最新バージョンのChromeブラウザを使用して体験してください。このドキュメントで連結するコンポーネントの、現在の主流デスクトップブラウザに対するサポート状況は次のとおりです。

オペレーティングシステム	ブラウザタイプ	ブラウザの最小バージョン要件

©2013-2022 Tencent Cloud. All rights reserved.

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+
Windows	デスクトップ版QQブラウザ（クイックコア）	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+

explain

詳細な互換性の照会については、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。

(2) ネットワーク環境の要件

TUICallKitを使用する際、ユーザーはファイアウォールの制限によって正常にオーディオビデオ通話が行えない可能性があります。ファイアウォール制限の対応関連 をご参照の上、対応するポートおよびドメイン名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。

(3) ウェブサイトドメイン名プロトコルの要件

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、ブラウザは本ドキュメントの連結コンポーネントの全機能をHTTPSプロトコルでしか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製品機能をスムーズに体験できるようにするため、ウェブサイトはhttps:// プロトコルのドメイン名にデプロイしてください。

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	サポートしています	サポートしています	サポートしています	推奨
本番環境	HTTPプロトコル	サポートしています	サポートしていません	サポートしていません	-
ローカル開発環境	http://localhost	サポートしています	サポートしています	サポートしています	推奨

©2013-2022 Tencent Cloud. All rights reserved.

ローカル開発環境	http://127.0.0.1	サポートしています	サポートしています	サポートしています	-
ローカル開発環境	http://[ローカルマシンIP]	サポートしています	サポートしていません	サポートしていません	-
ローカル開発環境	file:///	サポートしています	サポートしています	サポートしています	-

©2013-2022 Tencent Cloud. All rights reserved.

インターフェースのカスタマイズ（TUICallKit）

Android

最終更新日：：2022-11-21 16:34:34

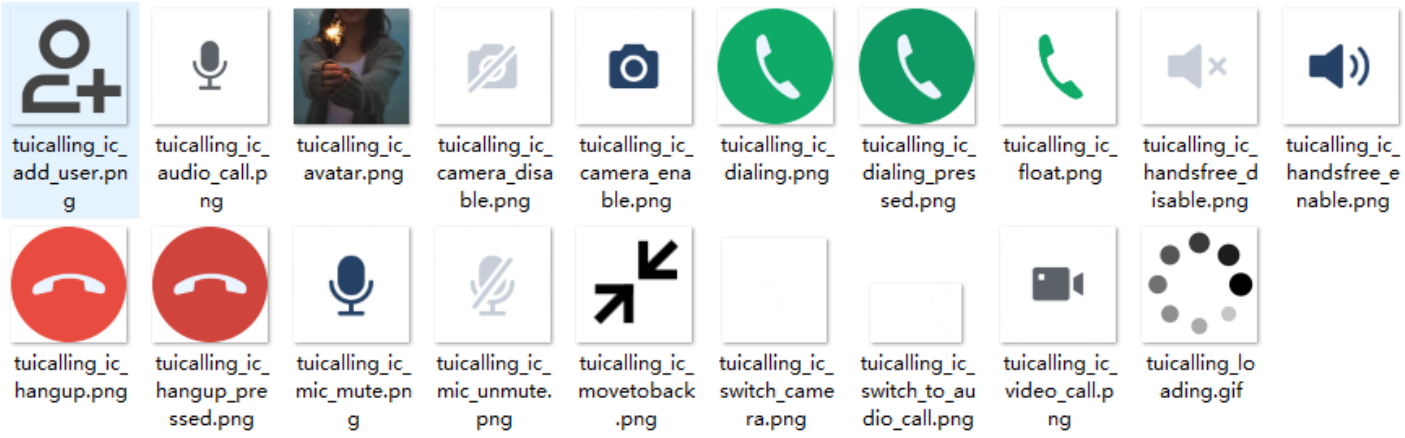
ここではTUICallKitのユーザーインターフェースをカスタマイズする方法についてご説明します。**インターフェース微調整**と**自身でのUI実装**という2つのソリューションをご用意しています。

方法1：インターフェース微調整ソリューション

ご提供するUIソースコードを直接変更することで、TUICallKitのユーザーインターフェースを調整します。
TUICallKitのインターフェースソースコードはGithubの `Android/tuicallkit` フォルダにあります。

アイコンの置き換え

`res\drawable-xxhdpi` フォルダ内のアイコンを直接置き換え、App全体のアイコンの色やスタイルに統一性を持たせることができます。置き換える際にアイコンファイルの名前を変更しないようにしてください。



着信音の置き換え

`res\raw` フォルダ内の3つのオーディオファイルを置き換えることで、着信音の置き換えができます。

ファイル名	用途
phone_dialing.mp3	呼び出し開始時の音声

©2013-2022 Tencent Cloud. All rights reserved.

ファイル名	用途
phone_hangup.mp3	通話終了の音声
phone_ringing.mp3	呼び出し応答時の音声

テキストの置き換え

values-zhおよびvalues-en内の `strings.xml` ファイルを変更することで、ビデオ通話画面内の文字列の内容を変更できます。

方法2：自身でのUI実装ソリューション

TUICallKitの通話機能全体はTUICallEngineというUIレスコンポーネントをベースにして実装されています。tuicallkitフォルダを削除し、TUICallEngineを完全にベースにしてご自身のUIを実装することが可能です。

TUICallEngine

TUICallEngine APIは通話コンポーネント全体の基盤インターフェースです。主に1対1オーディオビデオ通話およびグループ内通話の開始、応答、拒否、終了、デバイス操作などの重要なインターフェースを提供します。

API	説明
createInstance	TUICallEngineインスタンスの作成（シングルトンモード）
destroyInstance	TUICallEngineインスタンスの破棄（シングルトンモード）
init	オーディオビデオ通話基本機能の認証完了
addObserver	イベントコールバックの追加
removeObserver	コールバックインターフェースの削除
call	1v1通話の開始
groupCall	グループ通話の開始
accept	通話応答
reject	通話拒否
hangup	通話終了
ignore	通話を無視

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
inviteUser	グループ通話中に他の人を招待
joinInGroupCall	現在のグループ通話に自主的に参加
switchCallMediaType	通話メディアタイプの切り替え。ビデオ通話からオーディオ通話への切り替えなど
startRemoteView	リモートユーザービデオストリームのサブスクリプション開始
stopRemoteView	リモートユーザービデオストリームのサブスクリプション停止
openCamera	カメラの起動
closeCamera	カメラの終了
switchCamera	フロント/リアカメラの切り替え
openMicrophone	マイクをオンにする
closeMicrophone	マイクをオフにする
selectAudioPlaybackDevice	オーディオ再生デバイスの選択（ヘッドホン/スピーカー）
setSelfInfo	ユーザーのニックネーム、プロフィール画像の設定
enableMultiDeviceAbility	TUICallEngineのマルチデバイスログインモードのオン/オフ （プレミアム版パッケージのみサポート）

TUICallObserver

TUICallObserverはTUICallEngineに対応するコールバックイベントクラスです。このコールバックによって、関心のあるコールバックイベントを監視することができます。

API	説明
onError	通話中のエラーコールバック
onCallReceived	通話リクエストのコールバック
onCallCancelled	通話キャンセルのコールバック
onCallBegin	通話接続のコールバック
onCallEnd	通話終了のコールバック
onCallMediaTypeChanged	通話メディアタイプ変更発生のコールバック

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
onUserReject	xxxxユーザーによる通話拒否のコールバック
onUserNoResponse	xxxxユーザーの応答なしのコールバック
onUserLineBusy	xxxxユーザーが通話中である場合のコールバック
onUserJoin	xxxxユーザーの通話参加のコールバック
onUserLeave	xxxxユーザーの通話からの退出のコールバック
onUserVideoAvailable	xxxユーザーのビデオストリームの有無のコールバック
onUserAudioAvailable	xxxユーザーのオーディオストリームの有無のコールバック
onUserVoiceVolumeChanged	全ユーザーの音量レベルフィードバックのコールバック
onUserNetworkQualityChanged	全ユーザーのネットワーク品質フィードバックのコールバック

主要なタイプの定義

API	説明
TUICallDefine.MediaType	通話のメディアタイプ。列挙タイプ：ビデオ通話、音声通話
TUICallDefine.Role	通話のロール。列挙タイプ：発呼側、着呼側
TUICallDefine.Status	通話の状態。列挙タイプ：アイドル状態、応答待ち、応答中
TUICommonDefine.RoomId	オーディオビデオルームID。数字、文字列の2種類をサポートしています
TUICommonDefine.Camera	カメラIDパラメータ。列挙タイプ：フロントカメラ、リアカメラ
TUICommonDefine.AudioPlaybackDevice	音声再生デバイス。列挙タイプ：スピーカー、ヘッドホン
TUICommonDefine.NetworkQualityInfo	現在のネットワーク品質情報

©2013-2022 Tencent Cloud. All rights reserved.

iOS

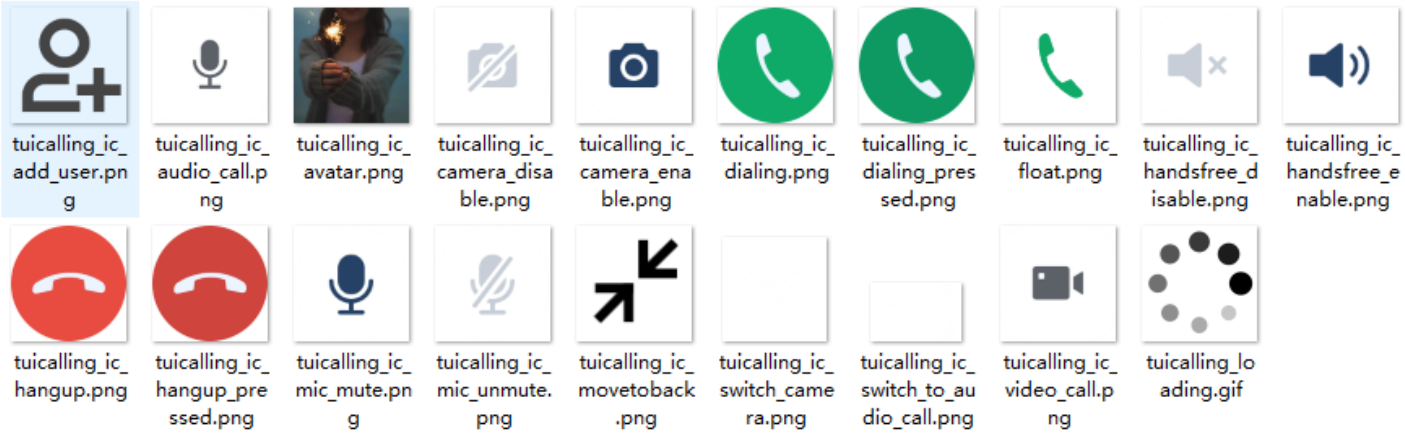
最終更新日：：2022-11-21 16:34:35

ここではTUICallKitのユーザーインターフェースをカスタマイズする方法についてご説明します。**インターフェース微調整**と**自身でのUI実装**という2つのソリューションをご用意しています。

方法1：インターフェース微調整ソリューション

ご提供するUIソースコードを直接変更することで、TUICallKitのユーザーインターフェースを調整します。
TUICallKitのインターフェースソースコードはGithubの `iOS/TUICallKit` フォルダにあります。

アイコンの置き換え `Resources\Calling.xcassets` フォルダ内のアイコンを直接置き換え、app全体のアイコンの色やスタイルに統一性を持たせることができます。置き換える際にアイコンファイルの名前を変更しないようにしてください。



着信音の置き換え `Resources\AudioFile` フォルダ内の3つのオーディオファイルを置き換えることで、着信音の置き換えができます。

ファイル名	用途
phone_dialing.m4a	呼び出し開始時の音声
phone_hangup.mp3	通話終了の音声
phone_ringing.flac	呼び出し応答時の音声

©2013-2022 Tencent Cloud. All rights reserved.

テキストの置き換え zh-Hans.lprojおよびen.lproj内の `CallingLocalized.strings` ファイルを変更することで、ビデオ通話画面内の文字列の内容を変更できます。

方法2：自身でのUI実装ソリューション

TUICallKitの通話機能全体はTUICallEngineというUIレスコンポーネントをベースにして実装されています。tuicallkitフォルダを削除し、TUICallEngineを完全にベースにしてご自身のUIを実装することが可能です。

TUICallEngine

TUICallEngine APIは通話コンポーネント全体の基盤インターフェースです。主に1対1オーディオビデオ通話およびグループ内通話の開始、応答、拒否、終了、デバイス操作などの重要なインターフェースを提供します。

API	説明
createInstance	TUICallEngineインスタンスの作成（シングルトンモード）
destroyInstance	TUICallEngineインスタンスの破棄（シングルトンモード）
init	オーディオビデオ通話基本機能の認証完了
addObserver	イベントコールバックの追加
removeObserver	コールバックインターフェースの削除
call	1v1通話の開始
groupCall	グループ通話の開始
accept	通話応答
reject	通話拒否
hangup	通話終了
ignore	通話を無視
inviteUser	グループ通話中に他の人を招待
joinInGroupCall	現在のグループ通話に自主的に参加
switchCallMediaType	通話メディアタイプの切り替え。ビデオ通話からオーディオ通話への切り替えなど
startRemoteView	リモートユーザービデオストリームのサブスクリプション開始

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
stopRemoteView	リモートユーザービデオストリームのサブスクリプション停止
openCamera	カメラの起動
closeCamera	カメラの終了
switchCamera	フロント/リアカメラの切り替え
openMicrophone	マイクをオンにする
closeMicrophone	マイクをオフにする
selectAudioPlaybackDevice	オーディオ再生デバイスの選択（ヘッドホン/スピーカー）
setSelfInfo	ユーザーのニックネーム、プロフィール画像の設定
enableMultiDeviceAbility	TUICallEngineのマルチデバイスログインモードのオン/オフ （プレミアム版パッケージのみサポート）

TUICallObserver

TUICallObserverはTUICallEngineに対応するコールバックイベントクラスです。このコールバックによって、関心のあるコールバックイベントを監視することができます。

API	説明
onError	通話中のエラーコールバック
onCallReceived	通話リクエストのコールバック
onCallCancelled	通話キャンセルのコールバック
onCallBegin	通話接続のコールバック
onCallEnd	通話終了のコールバック
onCallMediaTypeChanged	通話メディアタイプ変更発生のコールバック
onUserReject	xxxxユーザーによる通話拒否のコールバック
onUserNoResponse	xxxxユーザーの応答なしのコールバック
onUserLineBusy	xxxxユーザーが通話中である場合のコールバック
onUserJoin	xxxxユーザーの通話参加のコールバック

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
onUserLeave	xxxxユーザーの通話からの退出のコールバック
onUserVideoAvailable	xxxユーザーのビデオストリームの有無のコールバック
onUserAudioAvailable	xxxユーザーのオーディオストリームの有無のコールバック
onUserVoiceVolumeChanged	全ユーザーの音量レベルフィードバックのコールバック
onUserNetworkQualityChanged	全ユーザーのネットワーク品質フィードバックのコールバック

主要なタイプの定義

API	説明
TUICallMediaType	通話のメディアタイプ。列挙タイプ：ビデオ通話、音声通話
TUICallRole	通話のロール。列挙タイプ：発呼側、着呼側
TUICallStatus	通話の状態。列挙タイプ：アイドル状態、応答待ち、応答中
TUIRoomId	オーディオビデオルームID。数字、文字列の2種類をサポートしています
TUICallCamera	カメラIDパラメータ。列挙タイプ：フロントカメラ、リアカメラ
TUIAudioPlaybackDevice	音声再生デバイス。列挙タイプ：スピーカー、ヘッドホン
TUINetworkQualityInfo	現在のネットワーク品質情報

©2013-2022 Tencent Cloud. All rights reserved.

Web

最終更新日：：2023-02-10 18:25:15

TUICallKitインターフェースのカスタマイズガイド

ここではTUICallKitのユーザーインターフェースをカスタマイズする方法についてご説明します。インターフェース微調整と自身でのUI実装という2つのソリューションをご用意しています。

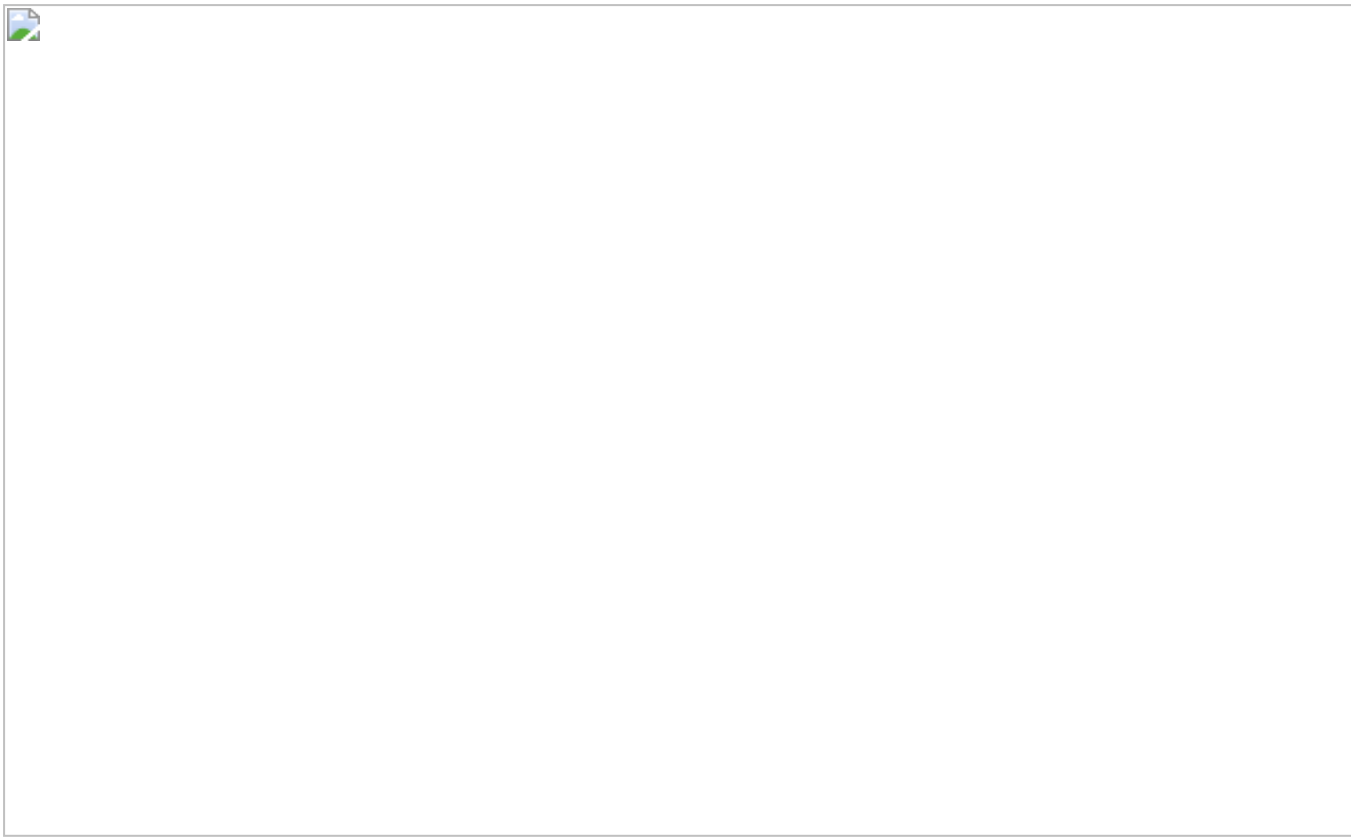
方法1：インターフェース微調整ソリューション

ご提供するUIソースコードを直接変更することで、TUICallKitのユーザーインターフェースを調整します。
TUICallKitのインターフェースソースコードはGithubの `Web/` フォルダにあります。

アイコンの置き換え

`src/icons` フォルダ内のアイコンコンポーネントを直接変更し、app全体のアイコンの色やスタイルに統一性を持たせることができます。置き換える際にアイコンファイルの名前を変更しないようにしてください。アイコンのプレビューは `src/assets` で参照できます。

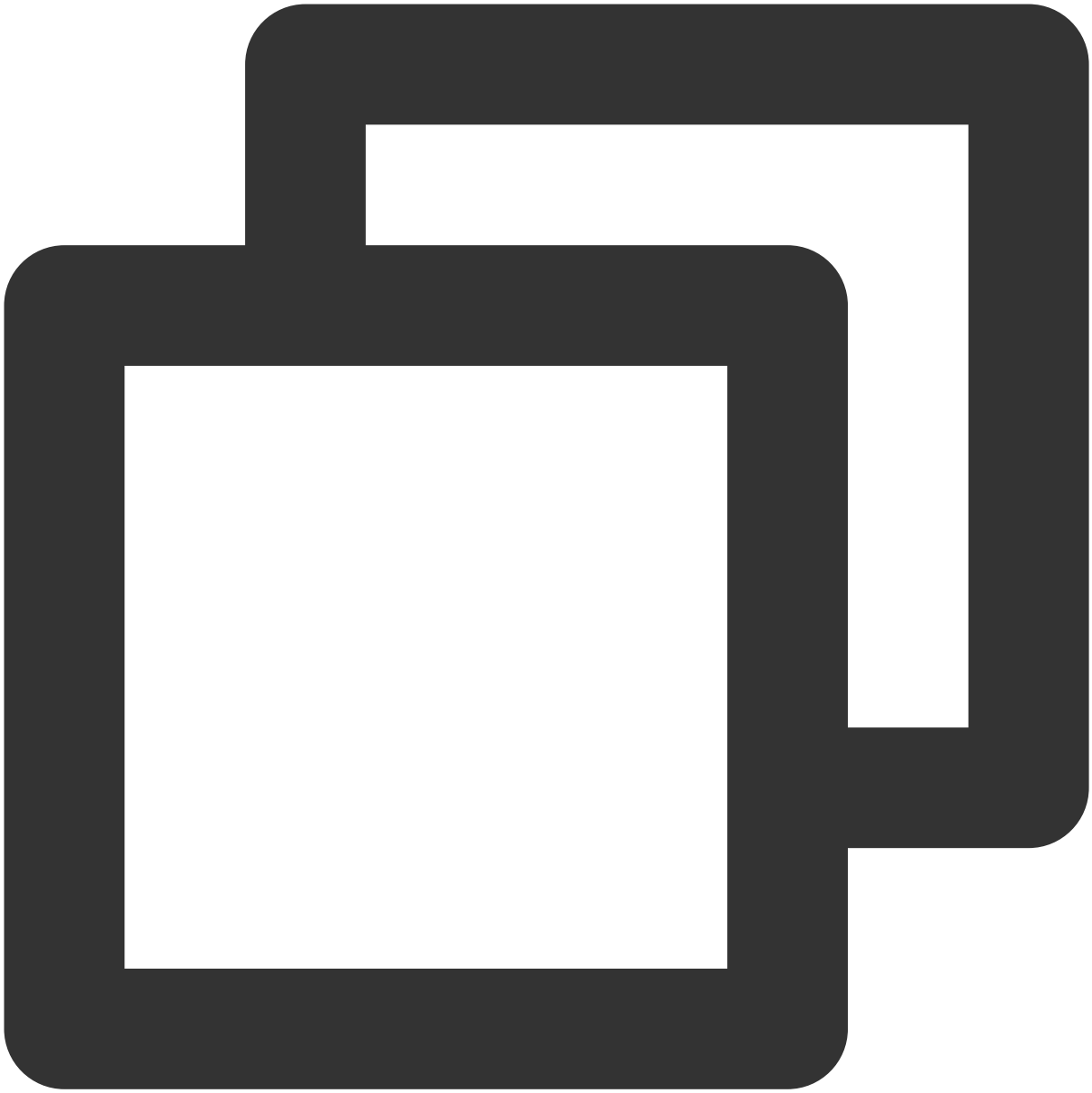
©2013-2022 Tencent Cloud. All rights reserved.



UIレイアウトの調整

src/components/ファイルのそれぞれのページを変更することで、オーディオビデオ通話インターフェースを変更できます

©2013-2022 Tencent Cloud. All rights reserved.



```
- components/
    - Calling-C2CVideo.vue          2人ビデオ通話
    - Calling-Group.Vue             多人数オーディオ・ビデオ通話
    - ControlPanel.vue              コントロールパネル
    - ControlPanelItem.vue          コントロールパネルサブ項目
    - Dialing.vue                   電話発信ページ、着信ページ、2人オーディオ通話
    - MicrophoneIcon.vue            音量の変化を表示できるマイクIcon
    - TUICallKit.vue                TUICallKit全体コンポーネント
```

スタイルの変更

©2013-2022 Tencent Cloud. All rights reserved.

スタイルファイルは `src/style.css` にあります。UIのスタイルに合わせてご自身で調整できます。

方法2：自身でのUI実装ソリューション

TUICallKitのすべての通話機能は、TUICallEngineというUIレスSDKをベースにして実装されており、TUICallEngineを完全にベースにしてご自身のUIを実装することが可能です。詳細については以下をご参照ください

TUICallEngineアクセスガイド

TUICallEngine APIインターフェースアドレス

©2013-2022 Tencent Cloud. All rights reserved.

オフライン通知（TUICallKit）
Android

最終更新日：：2022-11-21 16:34:35

オフライン通知機能は、アプリケーションが「バックエンドで実行中」または「オフライン状態」の場合でも、オーディオビデオ通話の着信を受信できる機能です。TUICallKitはTUIOfflinePushコンポーネントを統合することでオフライン通知機能を実現しています。ここでは、オーディオビデオ通話プロジェクトで `TUIOfflinePush` コンポーネントを統合する方法についてご説明します。

事前準備

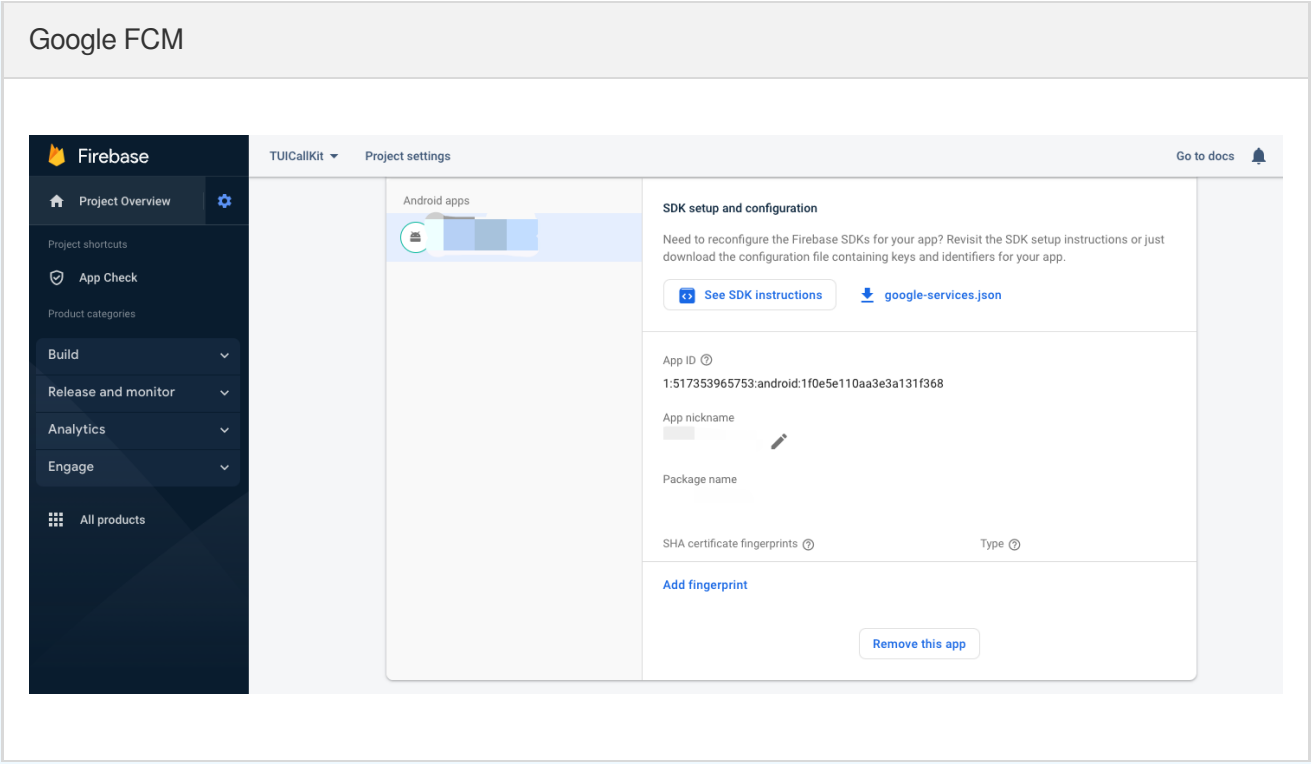
1. **アプリケーションをメーカーのプッシュプラットフォームに登録する** オフラインプッシュ機能はメーカーオリジナルのチャネルに依存します。ご自身のアプリケーションを各メーカーのプッシュプラットフォームに登録し、APPIDおよびAPPKEYなどのパラメータを取得する必要があります。

注意：
次の2つのファイルはこの後の手順で使用します

- Huaweiプラットフォームに登録する場合は、 `agconnect-services.json` ファイルをダウンロードして保存します。
- Googleプラットフォームに登録する場合は、 `google-services.json` ファイルをダウンロードして保存します。

  Google FCM

©2013-2022 Tencent Cloud. All rights reserved.



2. **IMコンソールで設定を行う**

メーカーのチャネルに登録するにはご自身のパッケージ名を渡す必要があります。メッセージの相互運用のため、各メーカーで入力するパッケージ名は一致させる必要があります。生成したID、APPIDおよびAPPKEYを記録します。これらはこの後の手順で使用します。

ステップ1：コンポーネントのダウンロードとインポート

1. Githubでコードをクローン/ダウンロードした後、下図のように、Androidディレクトリ下のtuiofflinepushサブディレクトリを現在のプロジェクトのappの同階層のディレクトリにコピーします。

©2013-2022 Tencent Cloud. All rights reserved.



2. プロジェクトのルートディレクトリ下で `setting.gradle` ファイルを見つけ、その中に次のコードを追加します。

```
include ':tuiofflinepush'
```

3. appディレクトリ下で `build.gradle` ファイルを見つけ、その中に次のコードを追加します。その役割は、現在のappの新たに追加したtuiofflinepushコンポーネントへの依存を宣言するものです。

```
api project(':tuiofflinepush')
```

ステップ2：プロジェクト設定の完了

1. appディレクトリ下で `build.gradle` ファイルを見つけ、アプリケーションパッケージ名をご自身のパッケージ名に変更します。

```
applicationId 'com.****.trtc'
```

2. appディレクトリ下で `build.gradle` ファイルを見つけ、 `ViVo` アクセスパラメータの `VIVO_APPKEY` 、 `VIVO_APPID` および `HONOR_APPID` を設定し、コンパイルまたは実行エラーを避けます。

©2013-2022 Tencent Cloud. All rights reserved.

```
manifestPlaceholders = [
"VIVO_APPKEY": "PLACEHOLDER",
"VIVO_APPID" : "PLACEHOLDER",
"HONOR_APPID": "PLACEHOLDER"
]
```

3. **Huawei**および**Google**ファイルの設定：appディレクトリ下で `google-services.json` ファイルを置き換えます。このファイルは事前準備でGoogleプラットフォームに登録した際に保存したファイルです。 appディレクトリ下に `agconnect-services.json` ファイルを追加します。このファイルは事前準備でHuaweiプラットフォームに登録した際に保存したファイルです。

4. 事前準備で記録したID、APPID、APPKEYを `PrivateConstants` ファイルに入力し、パラメータの設定が正しいかどうかを確認します。**入力するパラメータは次のとおりです**。

```
public class PrivateConstants {
/****** Xiaomiオフラインプッシュパラメータstart ******/
// Tencent Cloudコンソールで、サードパーティの証明書プッシュ後に割り当てられた証明書IDをアップロードします
public static final long XM_PUSH_BUZID = アプリケーションが割り当てた証明書ID;
// Xiaomiオープンプラットフォームが割り当てたアプリケーションAPPIDおよびAPPKEY
public static final String XM_PUSH_APPID = "アプリケーションが割り当てたAPPID";
public static final String XM_PUSH_APPKEY = "アプリケーションが割り当てたAPPKEY";
/****** Xiaomiオフラインプッシュパラメータend ******/
}
```

> 注意：
> この手順は非常に重要です。パラメータが正しく設定されているかどうかをよく確認してください。

上記の手順が完了すると、プロジェクトに `TUICallKit` のオフライン通知機能が実装されます。

ステップ3：オフライン通知内容のカスタマイズ

TUICallKitはデフォルトの通知スタイルを提供していますが、通知内容をカスタマイズしたい場合は、OfflinePushInfoConfig.javaファイルを変更してください。

©2013-2022 Tencent Cloud. All rights reserved.

```
public static TUIOfflinePushInfo createOfflinePushInfo(Context context) {
TUIOfflinePushInfo pushInfo = new TUIOfflinePushInfo();
pushInfo.setTitle("mike");
pushInfo.setDesc("You have receive a new call");
// OPPOはChannelIDを設定するとプッシュメッセージを受信することができます。このchannelIDはコンソールと一致している必要があります
// OPPO must set a ChannelID to receive push messages. This channelID needs to be the same as the console.
pushInfo.setAndroidOPPOChannelID("tuikit");
pushInfo.setIgnoreIOSBadge(false);
pushInfo.setIOSSound("phone_ringing.mp3");
return pushInfo;
}
```

よくあるご質問

オフラインプッシュ通知が受信できない場合は、お調べしますのでお問い合わせください。

1、通知が受信できない

メーカーのコンソールを使用してプッシュテストを行い、成功した場合はメーカーのチャネルに問題がないことがわかります。さらにTUIOfflinePushコンソールのメーカーパラメータ設定が正しいかどうかを確認し、要件に従って入力します。（テストの結果、vivo x9はコンソールでメッセージのカテゴリーを設定する必要があります）。

一部のスマートフォンでは通知が「重要でない通知」に入ることがあります。ステータスバーをプルダウンし、「重要でない通知」に分類されていないかを確認してください。

TUIOfflinePushの登録が正常に行われているかどうかを確認します。次のログをフィルタリングします。

```
TUIOfflinePush
```

2、画面ロック時にディスプレイが点灯しない

Androidスマートフォンはメーカーとプラットフォームの制限により、画面ロック状態での必要な権限が異なります。状況に応じて以下のトラブルシューティングを行ってください。

メーカーのロック画面通知権限がオンになっているかを確認する 一部のメーカーではルールの統一化を行っています。例えばXiaomiではロック画面中のオフライン通知到着時にディスプレイが点灯しない場合、**設定 > ロック画面**で、**ロック画面中の通知受信時にディスプレイを点灯**スイッチをクリックしてオンにします。

©2013-2022 Tencent Cloud. All rights reserved.

アプリケーションのロック画面通知権限がオンになっているかを確認する 例：Xiaomiではロック画面表示権限が必要です。

> 説明：
> この問題が起こった場合は互換性の処理が必要です。TencentのQQグループ（592465424）に参加すると、問い合わせとフィードバックを行うことができます。

3、オフラインプッシュ通知をクリックしても通話画面が開かない

通話リクエストが見つかるかどうかを確認する必要があります。次のログをフィルタリングできます。

```
onReceiveNewInvitation
```

4、アプリケーションがバックエンドにあるとき、通話画面をフロントエンドに自動的にプルできない

アプリケーションをバックエンドからフロントエンドに自動的にプルする場合、Appが「バックエンド自動起動」または「フローティングウィンドウ」権限を有効にしているかどうかを確認する必要があります。

メーカーが異なる場合や、同じメーカーであってもAndroidのバージョンが異なる場合は、アプリケーションに許可する権限や権限名が一致しないことがありますのでご注意ください。例えば、Xiaomi 6では**バックエンドの画面ポップアップ**権限を有効にすればよいだけですが、Redmiでは**バックエンドの画面ポップアップ**と**フローティングウィンドウの表示**の権限を同時に有効にする必要があります。

> 説明：
> 手動ですべての権限を有効化しても、通話画面のフロントエンドへの自動プルができないことをテスト中に発見した場合は、互換性の処理が必要です。

©2013-2022 Tencent Cloud. All rights reserved.

iOS

最終更新日：：2022-11-21 16:34:35

オフライン通知機能は、Appがバックエンドで実行中またはオフライン状態の場合でも、オーディオビデオ通話の着信を受信できる機能です。TUICallKitはAppleが提供するシステムレベルのプッシュチャネル（APNs）を使用してメッセージの通知を行います。

ステップ1：オフラインプッシュの設定

1. **APNs証明書の申請**：具体的にはオフラインプッシュ(iOS)のステップ1をご参照ください。

2. **IMコンソール設定**：具体的にはオフラインプッシュ(iOS)のステップ2をご参照ください。

3. **Appへの毎回のログイン時に、AppleからdeviceTokenを取得します**。具体的にはオフラインプッシュ(iOS)のステップ3をご参照ください。

4. **IM SDKにログイン後、deviceTokenをTencent Cloudにアップロードします**。具体的にはオフラインプッシュ(iOS)のステップ4をご参照ください。

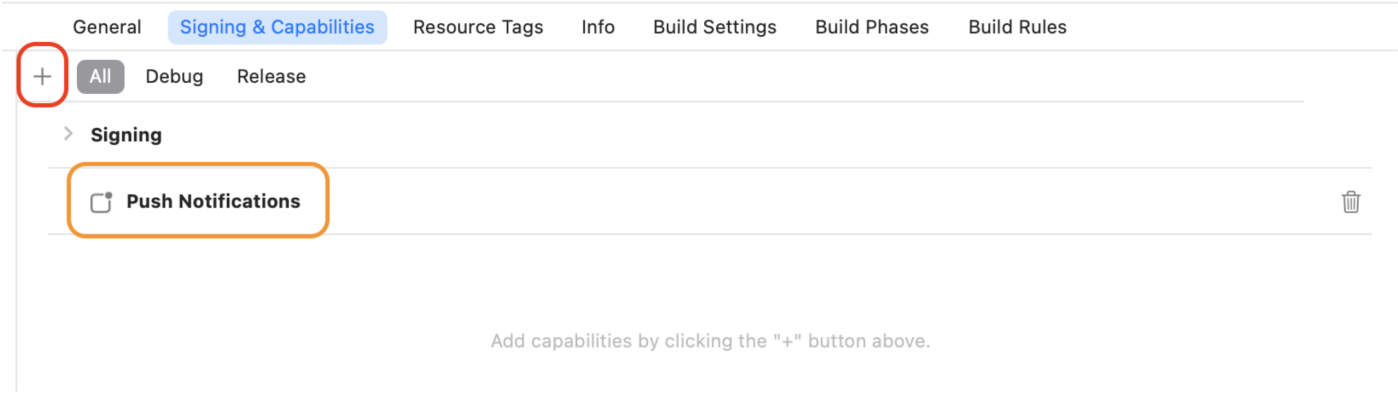
> 説明：
> その後の手順はTUICallKitがすでに完了していますので、オフラインプッシュ(iOS)のその他の手順を参照する必要はありません。

ステップ2：プロジェクトの設定

アプリケーションプログラムに必要な権限を追加する必要があります。Xcodeプロジェクトでプッシュ通知機能を有効にしてください。

Xcodeプロジェクトを開き、**Project** > **Target** > **Capabilities**ページで赤枠内の＋ボタンをクリックし、**Push Notifications**を選択して追加します。追加後の結果は図の黄色枠のようになります。

©2013-2022 Tencent Cloud. All rights reserved.



上記の手順が完了し、プロジェクトを実行すると、 `TUICallKit` のオフライン通知機能を体験できるようになります。

よくあるご質問

1、プッシュ通知を受信できず、バックエンドでbad devicetokenというエラーが表示されました。

AppleのdeviceTokenは現在のコンパイル環境に関連しています。IMSDKにログイン後にdeviceTokenをTencent Cloudにアップロードした際に使用した証明書IDがtokenと一致しなかった場合、エラーが表示されます。

Release環境を使用してコンパイルした場合、 `-application:didRegisterForRemoteNotificationsWithDeviceToken:` のコールバックで返されるのはリリース環境のtokenです。この場合、businessIDには本番環境の証明書IDを設定する必要があります。

Debug環境を使用してコンパイルした場合、 `-application:didRegisterForRemoteNotificationsWithDeviceToken:` のコールバックで返されるのは開発環境のtokenです。この場合、businessIDには開発環境の証明書IDを設定する必要があります。

```
V2TIMAPNSConfig *confg = [[V2TIMAPNSConfig alloc] init];
/* ユーザーは自身でAppleに開発者証明書を登録します。開発者アカウントに証明書(p12ファイル)をダウンロードして生成し、生成したp12ファイルをTencent証明書管理コンソールに渡すと、コンソールは証明書IDを自動生成し、証明書IDを以下のbusiIdパラメータに渡します。*/
//証明書IDをプッシュします
confg.businessID = sdkBusiId;
confg.token = self.deviceToken;
[[V2TIMManager sharedInstance] setAPNS:confg succ:^{
} fail:^(int code, NSString *msg) {
}];
```

©2013-2022 Tencent Cloud. All rights reserved.

2、iOS開発環境で、登録してもdeviceTokenが返されないことや、APNsのtokenリクエストに失敗したと表示されることがあります。

この問題の現象はAPNsサービスが不安定なために発生します。次の方法で解決を試みることができます。

1. スマートフォンにSIMカードを挿入し、4Gネットワークを使用してテストを行う。

2. アンインストールして再インストール、Appの再起動、シャットダウンして再起動を行ってからテストを行う。

3. 本番環境のパッケージを作成してテストを行う。

4. 他のiOSシステムのスマートフォンに代えてテストを行う。

©2013-2022 Tencent Cloud. All rights reserved.

クラウドレコーディング（TUICallKit）

最終更新日：：2022-11-21 16:34:35

ここではTUICallKitのクラウドレコーディングを起動し、重要な通話の保存、審査などに役立てる方法についてご説明します。自動レコーディングソリューションとREST APIレコーディングソリューションという2つのソリューションをご用意しています。

> 説明：
> TUICallKitはTencent Cloudの基本的なPaaSサービスを複数統合しています。このうち、オーディオビデオ関連機能はTRTCに依存しています。このためTUICallKitのクラウドレコーディング機能はTRTCコンソールで設定する必要があります。

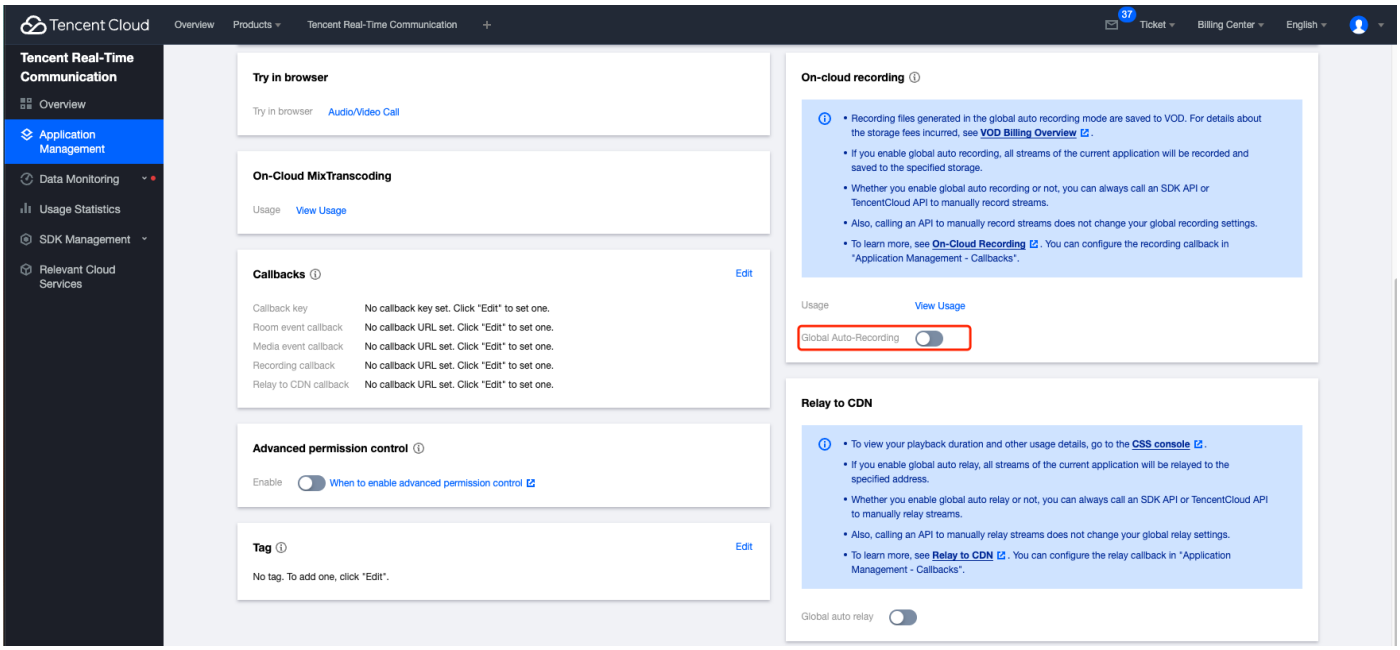
方法1：自動レコーディングソリューション（推奨）

自動レコーディングソリューションの使用を推奨します。業務側でレコーディングを起動および停止する必要がなく、レコーディングタスクはTencent Cloud TRTCバックエンドが管理します。通話中にオーディオビデオストリームのアップストリームがあれば自動的にレコーディングし、**スピーディーかつ簡単にアクセスできます**。次のいくつかの手順で完成させることができます。

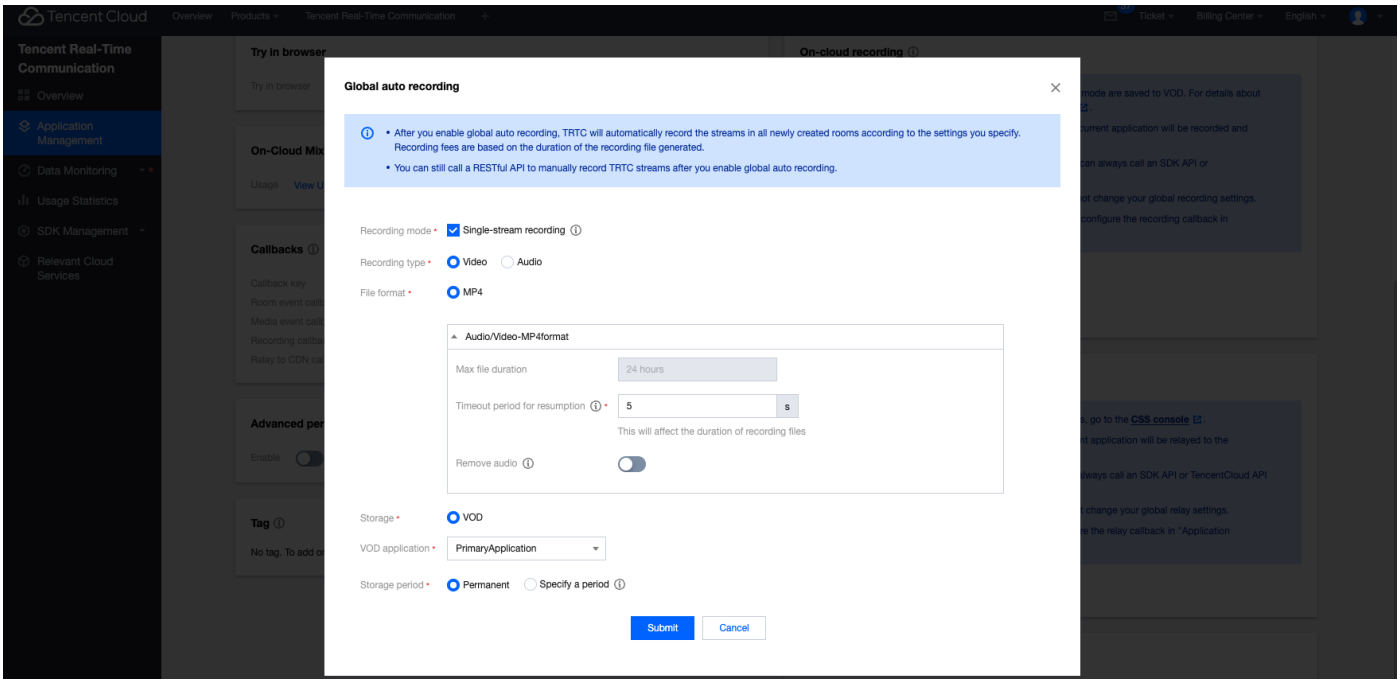
1. TRTCコンソール > アプリケーション管理でSDKAppIdに対応するアプリケーションを見つけ、機能設定ページに進みます。

2. 機能設定ページでクラウドレコーディング設定のカードを確認できます。**クラウドレコーディング機能を有効にした後、Global Auto-Recordingテンプレートの作成**をクリックします。

©2013-2022 Tencent Cloud. All rights reserved.



3. オーディオビデオ通話の業務シナリオ（1v1通話、グループ通話）に応じて、次のようにパラメータを設定することを推奨しますが、ご自身の業務ニーズに応じてカスタムレコーディングテンプレートを設定することも可能です。



注意：

グローバルレコーディングがサポートする最大ミクスストリーミング人数は8人です。通話人数が8人を超える場合（自分を含めて）、最後のユーザーのストリームはレコーディングされません。

Global Auto-Recording機能を有効にすると、通話応答後かつオーディオビデオアップストリームが存在する場合にレコーディングタスクの自動起動がトリガーされ、通話終了後に自動的にレコーディングが停止します。ネットワークまたはその他の異常によって退室になった場合、ご自身の設定したMaxIdleTime値（アイドル状態の待機時間。デフォルトでは5秒）に基づいてバックエンドが自動的にレコーディングタスクを停止し、それ以上の料金ロスが発生しないようにします。

4. **テンプレートの作成が完了した後、**Global Auto-Recordingにチェックを入れれば完了です。

方法2：REST APIレコーディングソリューション

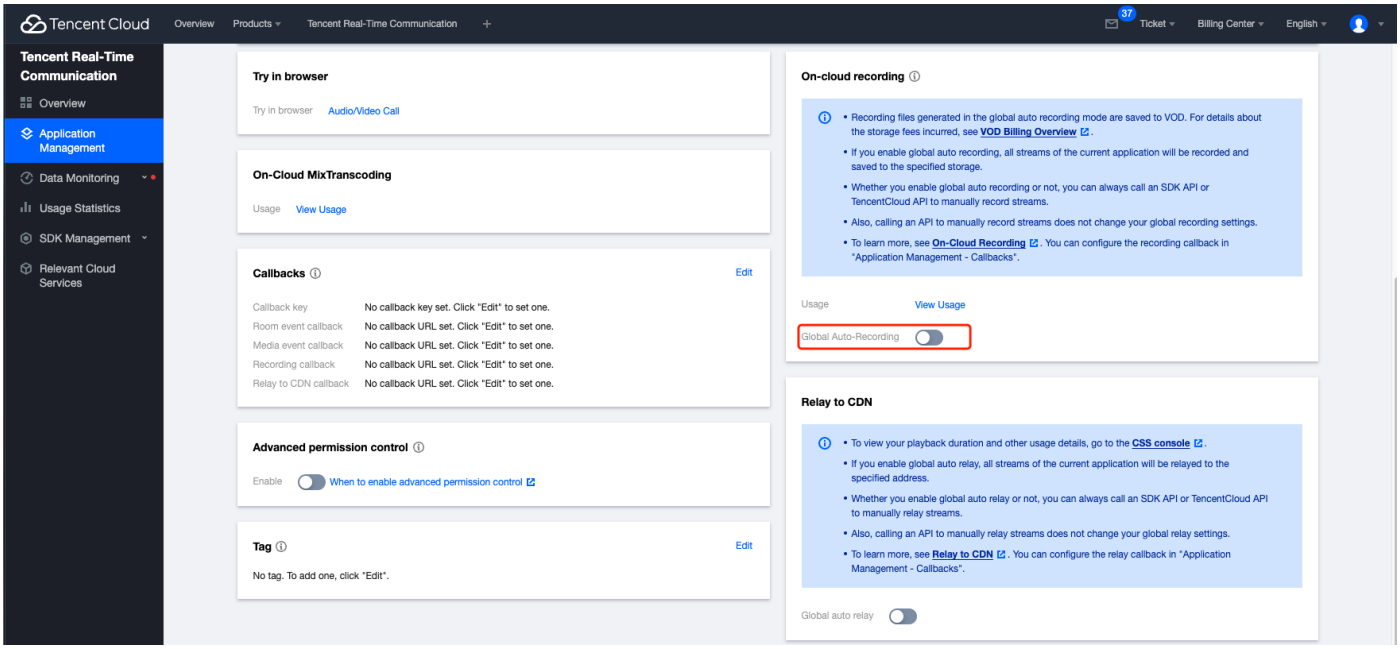
自動レコーディングソリューションで業務ニーズを満たせない場合は、よりフレキシブルなREST APIレコーディングソリューションを使用することもできます。レコーディングサブスクリプションルーム内のキャスターの指定、ミクスストリーミングレイアウトのカスタマイズ、レコーディング中のレイアウトおよびサブスクリプション更新などが可能ですが、**業務バックエンドサービスとの組み合わせが必要で、接続がより複雑になり、機能は強力になります。重要な手順は次のとおりです。**

1. TRTCコンソール > アプリケーション管理でSDKAppIdに対応するアプリケーションを見つけ、機能設定ページに進みます。

2. 機能設定ページでクラウドレコーディング設定のカードを確認できます。**クラウドレコーディング機能を有効にする**と、この時点で**レコーディングを手動でカスタマイズ**、すなわちREST APIモードにデフォルトでチェッ

©2013-2022 Tencent Cloud. All rights reserved.

クが入っています。



3. その後、REST API（CreateCloudRecording）を呼び出してクラウドレコーディングを起動することができます。ここではTUICallObserverの通知イベントを監視することで、オーディオビデオ通話の開始時にレコーディングを起動できるようにすることをお勧めします。Javaコードの例を挙げます。

```
TUICallEngine.createInstance(context).addObserver(new TUICallObserver() {
@Override
public void onCallBegin(TUICommonDefine.RoomId roomId, TUICallDefine.MediaType ca
llMediaType, TUICallDefine.Role callRole) {
// REST APIを使用してレコーディングタスクを起動することを業務バックエンドに通知します。
}
});
```

4. クライアントにネットワーク不良、プロセスキルなどが発生して通話が異常終了する可能性を考慮し、レコーディング終了の方法については、TRTCルームステータスのコールバック（詳細についてはサーバーイベントコールバックの監視をご参照ください）をサブスクライブし、TRTCルームのステータスが解散となったコールバックを受信した時点でREST API （DeleteCloudRecording）を呼び出してクラウドのレコーディングタスクを停止するようにすることをお勧めします。

よくあるご質問

1. レコーディング時間の明細を確認するにはどうすればよいですか。

©2013-2022 Tencent Cloud. All rights reserved.

TRTCコンソール > クラウドレコーディングでレコーディングの時間明細を確認できます。

2. レコーディングしたファイルを確認するにはどうすればよいですか。

VODコンソールにログインし、左側ナビゲーションバーで**メディア資産管理**を選択し、リスト上方の**プレフィックス検索**をクリックして**プレフィックス検索**を選択し、検索ボックスにキーワードを入力します。レコーディングファイルの命名ルールは次のとおりです。

- シングルストリーミングレコーディングMP4ファイル名のルール：
  `<sdkappid>_<roomid>_UserId_s_<userid>_UserId_e_<mediaid>_<index>.mp4`
- ミクスストリーミングレコーディングMP4ファイル名のルール： `<sdkappid>_<roomid>_<index>.mp4`

©2013-2022 Tencent Cloud. All rights reserved.

API参照（TUICallKit）
Android
API概要

最終更新日：：2022-11-21 16:34:35

TUICallKit (UIインターフェースあり)

TUICallKit APIはオーディオビデオ通話コンポーネントの**UIインターフェース付き**のものです。TUICallKit APIを使用することで、WeChatのようなオーディオビデオ通話シーンをシンプルなインターフェースでスピーディーに実現できます。

API	説明
createInstance	TUICallKitインスタンスの作成（シングルトンモード）
setSelfInfo	ユーザーのプロフィール画像、ニックネームの設定
call	1v1通話の開始
groupCall	グループ通話の開始
joinInGroupCall	現在のグループ通話に自主的に参加
setCallingBell	カスタム着信音の設定
enableMuteMode	ミュートモードのオン/オフ
enableFloatWindow	フローティングウィンドウ機能のオン/オフ

TUICallEngine (UIインターフェースなし)

TUICallEngine APIはオーディオビデオ通話コンポーネントの**UIインターフェースがない**ものです。TUICallKitのインタラクションではニーズを満たせない場合はこのAPIを使用し、業務ニーズに応じてパッケージをカスタマイズすることができます。

API	説明
createInstance	TUICallEngineインスタンスの作成（シングルトンモード）

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
destroyInstance	TUICallEngineインスタンスの破棄（シングルトンモード）
init	オーディオビデオ通話基本機能の認証完了
addObserver	イベントコールバックの追加
removeObserver	コールバックインターフェースの削除
call	1v1通話の開始
groupCall	グループ通話の開始
accept	通話応答
reject	通話拒否
hangup	通話終了
ignore	通話を無視
inviteUser	グループ通話中に他の人を招待
joinInGroupCall	現在のグループ通話に自主的に参加
switchCallMediaType	通話メディアタイプの切り替え。ビデオ通話からオーディオ通話への切り替えなど
startRemoteView	リモートユーザービデオストリームのサブスクリプション開始
stopRemoteView	リモートユーザービデオストリームのサブスクリプション停止
openCamera	カメラの起動
closeCamera	カメラの終了
switchCamera	フロント/リアカメラの切り替え
openMicrophone	マイクをオンにする
closeMicrophone	マイクをオフにする
selectAudioPlaybackDevice	オーディオ再生デバイスの選択（ヘッドホン/スピーカー）
setSelfInfo	ユーザーのニックネーム、プロフィール画像の設定
enableMultiDeviceAbility	TUICallEngineのマルチデバイスログインモードのオン/オフ（プレミアム版パッケージのみサポート）

©2013-2022 Tencent Cloud. All rights reserved.

TUICallObserver

TUICallObserverはTUICallEngineに対応するコールバックイベントクラスです。このコールバックによって、関心のあるコールバックイベントを監視することができます。

API	説明
onError	通話中のエラーコールバック
onCallReceived	通話リクエストのコールバック
onCallCancelled	通話キャンセルのコールバック
onCallBegin	通話接続のコールバック
onCallEnd	通話終了のコールバック
onCallMediaTypeChanged	通話メディアタイプ変更発生のコールバック
onUserReject	xxxxユーザーによる通話拒否のコールバック
onUserNoResponse	xxxxユーザーの応答なしのコールバック
onUserLineBusy	xxxxユーザーが通話中である場合のコールバック
onUserJoin	xxxxユーザーの通話参加のコールバック
onUserLeave	xxxxユーザーの通話からの退出のコールバック
onUserVideoAvailable	xxxユーザーのビデオストリームの有無のコールバック
onUserAudioAvailable	xxxユーザーのオーディオストリームの有無のコールバック
onUserVoiceVolumeChanged	全ユーザーの音量レベルフィードバックのコールバック
onUserNetworkQualityChanged	全ユーザーのネットワーク品質フィードバックのコールバック

主要なタイプの定義

API	説明
TUICallDefine.MediaType	通話のメディアタイプ。列挙タイプ：ビデオ通話、音声通話
TUICallDefine.Role	通話のロール。列挙タイプ：発呼側、着呼側

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
TUICallDefine.Status	通話の状態。列挙タイプ：アイドル状態、応答待ち、応答中
TUICommonDefine.RoomId	オーディオビデオルームId。数字、文字列の2種類をサポートしています
TUICommonDefine.Camera	カメラIdパラメータ。列挙タイプ：フロントカメラ、リアカメラ
TUICommonDefine.AudioPlaybackDevice	音声再生デバイス。列挙タイプ：スピーカー、ヘッドホン
TUICommonDefine.NetworkQualityInfo	現在のネットワーク品質情報

©2013-2022 Tencent Cloud. All rights reserved.

TUICallKit

最終更新日：：2022-11-21 16:34:35

TUICallKit APIの概要

TUICallKit APIはオーディオビデオ通話コンポーネントの**UIインターフェース付き**のものです。TUICallKit APIを使用することで、WeChatのようなオーディオビデオ通話シーンをシンプルなインターフェースでスピーディーに実現できます。より詳細なアクセス手順については、TUICallKitクイックアクセスをご参照ください。

APIの概要

API	説明
createInstance	TUICallKitインスタンスの作成（シングルトンモード）
setSelfInfo	ユーザーのニックネーム、プロフィール画像の設定
call	1v1通話の開始
groupCall	グループ通話の開始
joinInGroupCall	現在のグループ通話に自主的に参加
setCallingBell	カスタム着信音の設定
enableMuteMode	ミュートモードのオン/オフ
enableFloatWindow	フローティングウィンドウ機能のオン/オフ

APIの詳細

createInstance

TUICallKitのシングルトンを作成します。

```
TUICallKit createInstance(Context context)
```

setSelfInfo

©2013-2022 Tencent Cloud. All rights reserved.

ユーザーニックネーム、プロフィール画像を設定します。ユーザーニックネームは500バイト以内、ユーザープロフィール画像はURL形式でなければなりません。

```
void setSelfInfo(String nickname, String avatar, TUICommonDefine.Callback 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
nickname	String	ターゲットユーザーのニックネーム
avatar	String	ターゲットユーザーのプロフィール画像

call

電話をかけます（1v1通話）。

```
void call(String userId, TUICallDefine.MediaType callMediaType)
```

パラ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	ターゲットユーザーのuserId
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話など

groupCall

グループ通話を開始します。

> 注意：
> グループ通話を使用する前にIMグループを作成する必要があります。作成済みの場合は無視してください。

```
void groupCall(String groupId, List<String> userIdList, TUICallDefine.MediaType callMediaType) {
```

パラメータは下表に示すとおりです。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
groupId	String	今回のグループ通話のグループID
userIdList	List	ターゲットユーザーのuserIdリスト
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話など

joinInGroupCall

グループ通話を開始します。

> 注意：
> グループ通話を使用する前にIMグループを作成する必要があります。作成済みの場合は無視してください。

```
void joinInGroupCall(TUICommonDefine.RoomId roomId, String groupId, TUICallDefine.MediaType callMediaType, TUICommonDefine.Callback 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	TUICommonDefine.RoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
groupId	String	今回のグループ通話のグループID
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話など

setCallingBell

カスタム着信音を設定します。 ここではローカルファイルアドレスのみ渡すことができます。このファイルディレクトリにアプリケーションがアクセスできることを確認する必要があります。

着信音を設定後、デバイスにバインドします。ユーザーを変更しても着信音はそのまま有効です。

デフォルトの着信音に戻したい場合は、 `filePath` にnullを渡します。

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

©2013-2022 Tencent Cloud. All rights reserved.

enableMuteMode

ミュートモードをオン/オフします。

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

enableFloatWindow

フローティングウィンドウ機能をオン/オフします。

デフォルトでは `false` で、通話画面の左上隅のフローティングウィンドウボタンは非表示になっており、trueに設定すると表示されます。

```
void enableFloatWindow(boolean enable);
```

©2013-2022 Tencent Cloud. All rights reserved.

TUICallEngine

最終更新日：：2022-11-21 16:34:36

TUICallEngine APIの概要

TUICallEngine APIはオーディオビデオ通話コンポーネントの**UIインターフェースがない**ものです。TUICallKitのインタラクションではニーズを満たせない場合はこのインターフェースを使用し、パッケージのインタラクションをカスタマイズすることができます。

APIの概要

API	説明
createInstance	TUICallEngineインスタンスの作成（シングルトンモード）
destroyInstance	TUICallEngineインスタンスの破棄（シングルトンモード）
init	オーディオビデオ通話基本機能の認証完了
addObserver	イベントコールバックの追加
removeObserver	コールバックインターフェースの削除
call	1v1通話の開始
groupCall	グループ通話の開始
accept	通話応答
reject	通話拒否
hangup	通話終了
ignore	通話を無視
inviteUser	グループ通話中に他の人を招待
joinInGroupCall	現在のグループ通話に自主的に参加
switchCallMediaType	通話メディアタイプの切り替え。ビデオ通話からオーディオ通話への切り替えなど

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
startRemoteView	リモートユーザービデオストリームのサブスクリプション開始
stopRemoteView	リモートユーザービデオストリームのサブスクリプション停止
openCamera	カメラの起動
closeCamera	カメラの終了
switchCamera	フロント/リアカメラの切り替え
openMicrophone	マイクをオンにする
closeMicrophone	マイクをオフにする
selectAudioPlaybackDevice	オーディオ再生デバイスの選択（ヘッドホン/スピーカー）
setSelfInfo	ユーザーのニックネーム、プロフィール画像の設定
enableMultiDeviceAbility	TUICallEngineのマルチデバイスログインモードのオン/オフ （プレミアム版パッケージのみサポート）

APIの詳細

createInstance

TUICallEngineのシングルトンを作成します。

```
TUICallEngine createInstance(Context context)
```

destroyInstance

TUICallEngineのシングルトンを破棄します。

```
void destroyInstance();
```

init

関数を初期化します。すべての機能を使用する前にこの関数を呼び出し、通話サービス認証を含む初期化操作を完了させてください。

©2013-2022 Tencent Cloud. All rights reserved.

```
void init(int sdkAppId, String userId, String userSig, TUICommonDefine.Callback c
allback)
```

パラ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、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
callback	TUICommonDefine.Callback	初期化コールバック。 `onSuccess` は初期化に成功したことを表します

addObserver

このコールバックインターフェースを追加すると、この応答を通じて `TUICallObserver` 関連のイベントコールバックを監視できます。

```
void addObserver(TUICallObserver observer);
```

removeObserver

コールバックインターフェースを削除します。

```
void removeObserver(TUICallObserver observer);
```

call

電話をかけます（1v1通話）。

```
void call(TUICommonDefine.RoomId roomId, String userId, TUICallDefine.MediaType c
allMediaType,
TUICallDefine.CallParams params, TUICommonDefine.Callback callback);
```

パラメータは下表に示すとおりです。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomId	TUICommonDefine.RoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
userId	String	ターゲットユーザーのuserId
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。例：ビデオ通話、音声通話など
params	TUICallDefine.CallParams	通話パラメータ拡張フィールド。例：カスタムコンテンツのオフラインプッシュ

groupCall

グループ通話を開始します。注意：グループ通話を使用する前にIMグループを作成する必要があります。作成済みの場合は無視してください。

```
void groupCall(TUICommonDefine.RoomId roomId, String groupId, List<String> userIdList,
TUICallDefine.MediaType callMediaType, TUICallDefine.CallParams params,
TUICommonDefine.Callback 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	TUICommonDefine.RoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
groupId	String	今回のグループ通話のグループID
userIdList	List	ターゲットユーザーのuserIdリスト
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。例：ビデオ通話、音声通話など
params	TUICallDefine.CallParams	通話パラメータ拡張フィールド。例：カスタムコンテンツのオフラインプッシュ

accept

現在の通話を受信します。着呼側として `onCallReceived()` のコールバックを受信した場合は、この関数を呼び出して通話に応答することができます。

©2013-2022 Tencent Cloud. All rights reserved.

```
void accept(TUICommonDefine.Callback callback);
```

reject

現在の通話を拒否します。着呼側として `onCallReceived()` のコールバックを受信した場合は、この関数を呼び出して通話を拒否することができます。

```
void reject(TUICommonDefine.Callback callback);
```

ignore

現在の通話を無視します。着呼側としてonCallReceived()のコールバックを受信した場合は、この関数を呼び出して通話を無視することができ、このとき発呼側はonUserLineBusyのコールバックを受信します。

備考：業務内にライブストリーミング、ミーティングなどのシーンがある場合、ライブストリーミング/ミーティング中の場合もこの関数を呼び出して通話を無視することができます。

```
void ignore(TUICommonDefine.Callback callback);
```

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(TUICommonDefine.Callback callback);
```

inviteUser

今回のグループ通話にユーザーを招待します。

ユースケース：グループ通話中のユーザーが自主的に他の人を招待する場合に使用します。

```
void inviteUser(List<String> userIdList, TUICallDefine.CallParams params,
TUICommonDefine.ValueCallback 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	ターゲットユーザーのuserIdリスト
params	TUICallDefine.CallParams	通話パラメータ拡張フィールド。例：カスタムコンテンツのオフラインプッシュ

©2013-2022 Tencent Cloud. All rights reserved.

joinInGroupCall

今回のグループ通話に自主的に参加します。

ユースケース：グループ内のユーザーが今回のグループ通話に自主的に参加する場合に使用します。

```
void joinInGroupCall(TUICommonDefine.RoomId roomId, String groupId,
TUICallDefine.MediaType callMediaType, TUICommonDefine.Callback 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	TUICommonDefine.RoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
groupId	String	今回のグループ通話のグループID
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話など

switchCallMediaType

ビデオ通話を音声通話へ切り替えます。

```
void switchCallMediaType(TUICallDefine.MediaType callMediaType);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話など

startRemoteView

リモートユーザーのビデオデータのサブスクリプションを開始します。このインターフェースはsetRenderViewの後に呼び出します。

```
void startRemoteView(String userId, TUIVideoView videoView, TUICommonDefine.PlayC
allback 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	ターゲットユーザーのuserId

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
videoView	TUIVideoView	レンダリング対象のビュー

stopRemoteView

リモートユーザーのビデオデータのサブスクリプションを停止します。

```
void stopRemoteView(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	ターゲットユーザーのuserId

openCamera

カメラをオンにする

```
void openCamera(TUICommonDefine.Camera camera, TUIVideoView videoView, TUICommonDefine.Callback 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
camera	TUICommonDefine.Camera	フロントカメラ/リアカメラ
videoView	TUIVideoView	レンダリング対象のビュー

closeCamera

カメラをオフにします。

```
void closeCamera();
```

switchCamera

フロント/リアカメラを切り替えます。

```
void switchCamera(TUICommonDefine.Camera camera);
```

パラメータは下表に示すとおりです。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
camera	TUICommonDefine.Camera	フロントカメラ/リアカメラ

openMicrophone

マイクをオンにします

```
void openMicrophone(TUICommonDefine.Callback callback);
```

closeMicrophone

マイクをオフにします。

```
void closeMicrophone();
```

selectAudioPlaybackDevice

オーディオ再生デバイスを選択します。 現在はヘッドホン、スピーカーをサポートしています。通話のシーンでは、このインターフェースを使用してハンズフリーモードのオン/オフが行えます。

```
void selectAudioPlaybackDevice(TUICommonDefine.AudioPlaybackDevice device);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
device	TUICommonDefine.AudioPlaybackDevice	ヘッドホン/スピーカー

setSelfInfo

ユーザーニックネーム、プロフィール画像を設定します。ユーザーニックネームは500バイト以内、ユーザープロフィール画像はURL形式でなければなりません。

```
void setSelfInfo(String nickname, String avatar, TUICommonDefine.Callback 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
nickname	String	ユーザーニックネーム

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
avatar	String	ユーザープロフィール画像（形式はURL）

enableMultiDeviceAbility

TUICallEngineのマルチデバイスログインモードをオン/オフします （プレミアム版パッケージのみサポート）。

```
void enableMultiDeviceAbility(boolean enable, TUICommonDefine.Callback callback);
```

©2013-2022 Tencent Cloud. All rights reserved.

TUICallObserver

最終更新日：：2022-11-24 10:46:25

TUICallObserver APIの概要

TUICallObserverはTUICallEngineに対応するコールバックイベントクラスです。このコールバックによって、関心のあるコールバックイベントを監視することができます。

コールバックイベントの概要

API	説明
onError	通話中のエラーコールバック
onCallReceived	通話リクエストのコールバック
onCallCancelled	通話キャンセルのコールバック
onCallBegin	通話接続のコールバック
onCallEnd	通話終了のコールバック
onCallMediaTypeChanged	通話メディアタイプ変更発生のコールバック
onUserReject	xxxxユーザーによる通話拒否のコールバック
onUserNoResponse	xxxxユーザーの応答なしのコールバック
onUserLineBusy	xxxxユーザーが通話中である場合のコールバック
onUserJoin	xxxxユーザーの通話参加のコールバック
onUserLeave	xxxxユーザーの通話からの退出のコールバック
onUserVideoAvailable	xxxユーザーのビデオストリームの有無のコールバック
onUserAudioAvailable	xxxユーザーのオーディオストリームの有無のコールバック
onUserVoiceVolumeChanged	全ユーザーの音量レベルフィードバックのコールバック
onUserNetworkQualityChanged	全ユーザーのネットワーク品質フィードバックのコールバック

©2013-2022 Tencent Cloud. All rights reserved.

コールバックイベントの詳細

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 msg);
```

パラ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

onCallReceived

新しい通話リクエストコールバックを受信します。着呼側が受信します。このイベントを監視することで、通話応答画面を表示するかどうかを決定できます。

```
void onCallReceived(String callerId, List<String> calleeIdList, String groupId,
TUICallDefine.MediaType callMediaType);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callerId	String	発呼側ID（招待者）
calleeIdList	List	着呼側IDリスト（被招待者）
groupId	String	グループ通話ID
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話など

onCallCancelled

©2013-2022 Tencent Cloud. All rights reserved.

今回の通話が発呼側からキャンセルされたことを表します（キャンセルの原因は発呼側の自主的なキャンセル、または通話タイムアウトによるキャンセルの両方の可能性があります）。着呼側が受信します。このイベントを監視することで、未応答通話などに類似した表示ロジックを実現できます。

```
void onCallCancelled(String callerId);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callerId	String	キャンセルしたユーザーのID

onCallBegin

通話接続を表します。発呼側と着呼側がどちらも受信できます。このイベントを監視することで、クラウドレコーディング、コンテンツ審査などのフローを開始できます。

```
void onCallBegin(TUICommonDefine.RoomId roomId, TUICallDefine.MediaType callMedia
Type, TUICallDefine.Role callRole);
```

パラ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	TUICommonDefine.RoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話
callRole	TUICallDefine.Role	ロール。列挙タイプ：発呼側、着呼側

onCallEnd

通話の終了を表します。発呼側と着呼側がどちらも受信できます。このイベントを監視することで、通話時間、通話タイプなどの情報を表示したり、クラウドのレコーディングフローを停止したりすることができます。

```
void onCallEnd(TUICommonDefine.RoomId roomId, TUICallDefine.MediaType callMediaTy
pe, TUICallDefine.Role callRole, 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

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomId	TUICommonDefine.RoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
callMediaType	TUICallDefine.MediaType	通話のメディアタイプ。ビデオ通話、音声通話
callRole	TUICallDefine.Role	ロール。列挙タイプ：発呼側、着呼側
totalTime	long	今回の通話時間

注意：
クライアントのイベントは一般的にすべて、プロセスキルなどの異常イベントに伴って消失します。通話時間の監視によって料金計算などのロジックを完成させたい場合は、REST APIを使用してこの種のフローを完成させることをお勧めします。

onCallMediaTypeChanged

通話のメディアタイプに変更が発生したことを表します。

```
void onCallMediaTypeChanged(TUICallDefine.MediaType oldCallMediaType,TUICallDefine.MediaType newCallMediaType);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
oldCallMediaType	TUICallDefine.MediaType	変更前の通話タイプ
newCallMediaType	TUICallDefine.MediaType	変更後の通話タイプ

onUserReject

通話が拒否された場合のコールバックです。1v1通話では、発呼側のみが拒否のコールバックを受信します。グループ通話では、すべての被招待者がこのコールバックを受信することができます。

```
void onUserReject(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

©2013-2022 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

onUserNoResponse

相手方が応答しない場合のコールバック。

```
void onUserNoResponse(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	応答しないユーザーのID

onUserLineBusy

通話中である場合のコールバック。

```
void onUserLineBusy(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	通話中のユーザーのID

onUserJoin

今回の通話に参加したユーザーがいる場合のコールバック。

```
void onUserJoin(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	現在の通話に参加したユーザーのID

onUserLeave

今回の通話から退出したユーザーがいる場合のコールバック。

©2013-2022 Tencent Cloud. All rights reserved.

```
void onUserLeave(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	現在の通話から退出したユーザーのID

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。

```
void onUserVideoAvailable(String userId, boolean isVideoAvailable);
```

パラ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	通話ユーザーID
isVideoAvailable	boolean	ユーザーのビデオが使用可能かどうか

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。

```
void onUserAudioAvailable(String userId, boolean isAudioAvailable);
```

パラ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	ユーザーのオーディオが使用可能かどうか

onUserVoiceVolumeChanged

ユーザーの通話音量のコールバック。

```
void onUserVoiceVolumeChanged(Map<String, Integer> volumeMap);
```

パラメータは下表に示すとおりです。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
volumeMap	Map< String, Integer>	ボリュームメーター。各useridにつき、対応するユーザーの音量レベルを取得できます。音量の最小値は0、最大値は100です

onUserNetworkQualityChanged

ユーザーのネットワーク品質のコールバック。

```
void onUserNetworkQualityChanged(List<TUICallDefine.NetworkQualityInfo> networkQualityList);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
networkQualityList	List	ネットワーク状態。各userIdにつき、対応するユーザーの現在のネットワーク品質を取得できます

©2013-2022 Tencent Cloud. All rights reserved.

iOS
API概要

最終更新日：：2022-11-21 16:34:36

TUICallKit (UIインターフェースあり)

TUICallKit APIはオーディオビデオ通話コンポーネントの**UIインターフェース付き**のものです。TUICallKit APIを使用することで、WeChatのようなオーディオビデオ通話シーンをシンプルなインターフェースでスピーディーに実現できます。

API	説明
createInstance	TUICallKitインスタンスの作成（シングルトンモード）
setSelfInfo	ユーザーのプロフィール画像、ニックネームの設定
call	1v1通話の開始
groupCall	グループ通話の開始
joinInGroupCall	現在のグループ通話に自主的に参加
setCallingBell	カスタム着信音の設定
enableMuteMode	ミュートモードのオン/オフ
enableFloatWindow	フローティングウィンドウ機能のオン/オフ

TUICallEngine (UIインターフェースなし)

TUICallEngine APIはオーディオビデオ通話コンポーネントの**UIインターフェースがない**ものです。TUICallKitのインタラクションではニーズを満たせない場合はこのAPIを使用し、業務ニーズに応じてパッケージをカスタマイズすることができます。

API	説明
createInstance	TUICallEngineインスタンスの作成（シングルトンモード）
destroyInstance	TUICallEngineインスタンスの破棄（シングルトンモード）

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
init	オーディオビデオ通話基本機能の認証完了
addObserver	イベントコールバックの追加
removeObserver	コールバックインターフェースの削除
call	1v1通話の開始
groupCall	グループ通話の開始
accept	通話応答
reject	通話拒否
hangup	通話終了
ignore	通話を無視
inviteUser	グループ通話中に他の人を招待
joinInGroupCall	現在のグループ通話に自主的に参加
switchCallMediaType	通話メディアタイプの切り替え。ビデオ通話からオーディオ通話への切り替えなど
setRenderView	ビデオ画面に表示するViewオブジェクトの設定
startRemoteView	ビデオ画面に表示するViewオブジェクトの設定
stopRemoteView	ビデオ画面に表示するViewオブジェクトの設定
openCamera	カメラの起動
closeCamera	カメラの終了
switchCamera	フロント/リアカメラの切り替え
openMicrophone	マイクをオンにする
closeMicrophone	マイクをオフにする
selectAudioPlaybackDevice	オーディオ再生デバイスの選択（ヘッドホン/ハンズフリー）
setSelfInfo	ユーザーのプロフィール画像、ニックネームの設定
enableMultiDeviceAbility	TUICallEngineのマルチデバイスログインモードのオン/オフ （プレミアム版パッケージのみサポート）

©2013-2022 Tencent Cloud. All rights reserved.

TUICallObserver

TUICallObserverはTUICallEngineに対応するコールバックイベントクラスです。このコールバックによって、関心のあるコールバックイベントを監視することができます。

API	説明
onError	通話中のエラーコールバック
onCallReceived	通話リクエストのコールバック
onCallCancelled	通話キャンセルのコールバック
onCallBegin	通話接続のコールバック
onCallEnd	通話終了のコールバック
onCallMediaTypeChanged	通話メディアタイプ変更発生のコールバック
onUserReject	xxxxユーザーによる通話拒否のコールバック
onUserNoResponse	xxxxユーザーの応答なしのコールバック
onUserLineBusy	xxxxユーザーが通話中である場合のコールバック
onUserJoin	xxxxユーザーの通話参加のコールバック
onUserLeave	xxxxユーザーの通話からの退出のコールバック
onUserVideoAvailable	xxxユーザーのビデオストリームの有無のコールバック
onUserAudioAvailable	xxxユーザーのオーディオストリームの有無のコールバック
onUserVoiceVolumeChanged	全ユーザーの音量レベルフィードバックのコールバック
onUserNetworkQualityChanged	全ユーザーのネットワーク品質フィードバックのコールバック

主要なタイプの定義

API	説明
TUICallMediaType	通話のメディアタイプ。列挙タイプ：ビデオ通話、音声通話
TUICallRole	通話のロール。列挙タイプ：発呼側、着呼側

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
TUICallStatus	通話の状態。列挙タイプ：アイドル状態、応答待ち、応答中
TUIRoomId	オーディオビデオルームId。数字、文字列の2種類をサポートしています
TUICallCamera	カメラIdパラメータ。列挙タイプ：フロントカメラ、リアカメラ
TUIAudioPlaybackDevice	音声再生デバイス。列挙タイプ：スピーカー、ヘッドホン
TUINetworkQualityInfo	現在のネットワーク品質情報

©2013-2022 Tencent Cloud. All rights reserved.

TUICallKit

最終更新日：：2022-11-21 16:34:36

TUICallKit APIの概要

TUICallKit APIはオーディオビデオ通話コンポーネントの**UIインターフェース付き**のものです。TUICallKit APIを使用することで、WeChatのようなオーディオビデオ通話シーンをシンプルなインターフェースでスピーディーに実現できます。より詳細なアクセス手順については、TUICallKitクイックアクセスをご参照ください。

APIの概要

API	説明
createInstance	TUICallKitインスタンスの作成（シングルトンモード）
setSelfInfo	ユーザーのニックネーム、プロフィール画像の設定
call	1v1通話の開始
groupCall	グループ通話の開始
joinInGroupCall	現在のグループ通話に自主的に参加
setCallingBell	カスタム着信音の設定
enableMuteMode	ミュートモードのオン/オフ
enableFloatWindow	フローティングウィンドウ機能のオン/オフ

APIの詳細

createInstance

TUICallKitのシングルトンを作成します。

```
- (instancetype)createInstance;
```

setSelfInfo

©2013-2022 Tencent Cloud. All rights reserved.

ユーザーニックネーム、プロフィール画像を設定します。ユーザーニックネームは500バイト以内、ユーザープロフィール画像はURL形式でなければなりません。

```
- (void)setSelfInfo:(NSString * _Nullable)nickname avatar:(NSString * _Nullable)avatar succ:(TUICallSucc)succ fail:(TUICallFail)fail
```

パラメータ	タイプ	意味
nickName	NSString	ターゲットユーザーのニックネーム
avatar	NSString	ターゲットユーザーのプロフィール画像

call

電話をかけます（1v1通話）

```
- (void)call:(NSString *)userId callMediaType:(TUICallMediaType)callMediaType;
```

パラ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	ターゲットユーザーのuserId
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など

groupCall

グループ通話を開始します。注意：グループ通話を使用する前にIMグループを作成する必要があります。作成済みの場合は無視してください。

```
- (void)groupCall:(NSString *)groupId userIdList:(NSArray<NSString *> *)userIdList callMediaType:(TUICallMediaType)callMediaType;
```

パラメータ	タイプ	意味
groupId	NSString	今回のグループ通話のグループID
userIdList	NSArray	ターゲットユーザーのuserIdリスト
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など

joinInGroupCall

©2013-2022 Tencent Cloud. All rights reserved.

グループ通話を開始します。注意：グループ通話を使用する前にIMグループを作成する必要があります。作成済みの場合は無視してください。

```
- (void)joinInGroupCall:(TUIRoomId *)roomId groupId:(NSString *)groupId callMediaType:(TUICallMediaType)callMediaType;
```

パラメータ	タイプ	意味
roomId	TUIRoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
groupId	NSString	今回のグループ通話のグループID
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など

setCallingBell

カスタム着信音を設定します。 ここではローカルファイルアドレスのみ渡すことができます。このファイルディレクトリにアプリケーションがアクセスできることを確認する必要があります。

着信音を設定後、デバイスにバインドします。ユーザーを変更しても着信音はそのまま有効です。

デフォルトの着信音に戻したい場合は、 `filePath` にnullを渡します。

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

enableMuteMode

ミュートモードをオン/オフします。

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

enableFloatWindow

フローティングウィンドウ機能をオン/オフします。デフォルトでは `false` で、通話画面の左上隅のフローティングウィンドウボタンは非表示になっており、trueに設定すると表示されます。

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

©2013-2022 Tencent Cloud. All rights reserved.

TUICallEngine

最終更新日：：2022-11-21 16:34:36

TUICallEngine APIの概要

TUICallEngine APIはオーディオビデオ通話コンポーネントの**UIインターフェースがない**ものです。TUICallKitのインタラクションではニーズを満たせない場合はこのインターフェースを使用し、パッケージのインタラクションをカスタマイズすることができます。

APIの概要

API	説明
createInstance	TUICallEngineインスタンスの作成（シングルトンモード）
destroyInstance	TUICallEngineインスタンスの破棄（シングルトンモード）
init	オーディオビデオ通話基本機能の認証完了
addObserver	イベントコールバックの追加
removeObserver	コールバックインターフェースの削除
call	1v1通話の開始
groupCall	グループ通話の開始
accept	通話応答
reject	通話拒否
hangup	通話終了
ignore	通話を無視
inviteUser	グループ通話中に他の人を招待
joinInGroupCall	現在のグループ通話に自主的に参加
switchCallMediaType	通話メディアタイプの切り替え。ビデオ通話からオーディオ通話への切り替えなど

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
startRemoteView	リモートユーザービデオストリームのサブスクリプション開始
stopRemoteView	リモートユーザービデオストリームのサブスクリプション停止
openCamera	カメラの起動
closeCamera	カメラの終了
switchCamera	フロント/リアカメラの切り替え
openMicrophone	マイクをオンにする
closeMicrophone	マイクをオフにする
selectAudioPlaybackDevice	オーディオ再生デバイスの選択（ヘッドホン/スピーカー）
setSelfInfo	ユーザーのニックネーム、プロフィール画像の設定
enableMultiDeviceAbility	TUICallEngineのマルチデバイスログインモードのオン/オフ （プレミアム版パッケージのみサポート）

APIの詳細

createInstance

TUICallEngineのシングルトンを作成します。

```
- (TUICallEngine *)createInstance;
```

destroyInstance

TUICallEngineのシングルトンを破棄します。

```
- (void)destroyInstance;
```

init

関数を初期化します。すべての機能を使用する前にこの関数を呼び出し、通話サービス認証を含む初期化操作を完了させてください。

```
- (void)init:(NSString *)sdkAppID userId:(NSString *)userId userSig:(NSString *)u
serSig succ:(TUICallSucc)succ fail:(TUICallFail)fail;
```

©2013-2022 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
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、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
callback	TUIDefine.Callback	初期化コールバック。 `onSuccess` は初期化に成功したことを表します

addObserver

このコールバックインターフェースを追加すると、この応答を通じて `TUICallObserver` 関連のイベントコールバックを監視できます。

```
- (void)addObserver:(id<TUICallObserver>)observer;
```

removeObserver

コールバックインターフェースを削除します。

```
- (void)removeObserver:(id<TUICallObserver>)observer;
```

call

電話をかけます（1v1通話）

```
- (void)call:(TUIRoomId *)roomId userId:(NSString *)userId callMediaType:(TUICall
MediaType)callMediaType params:(TUICallParams *)params succ:(TUICallSucc)succ fai
l:(TUICallFail)fail
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomId	TUIRoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
userId	NSString	ターゲットユーザーのuserId
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など
params	TUICallParams	通話パラメータ拡張フィールド。例：カスタムコンテンツのオフラインプッシュ

groupCall

グループ通話を開始します。注意：グループ通話を使用する前にIMグループを作成する必要があります。作成済みの場合は無視してください。

```
- (void)groupCall:(TUIRoomId *)roomId groupId:(NSString *)groupId userIdList:(NSArray <NSString *> *)userIdList callMediaType:(TUICallMediaType)callMediaType params:(TUICallParams *)params succ:(TUICallSucc)succ fail:(TUICallFail)fail
```

パラメータ	タイプ	意味
roomId	TUIRoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
groupId	NSString	今回のグループ通話のグループID
userIdList	NSArray	ターゲットユーザーのuserIdリスト
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など
params	TUICallParams	通話パラメータ拡張フィールド。例：カスタムコンテンツのオフラインプッシュ

accept

現在の通話を受信します。着呼側として `onCallReceived()` のコールバックを受信した場合は、この関数を呼び出して通話に応答することができます。

```
- (void)accept:(TUICallSucc)succ fail:(TUICallFail)fail;
```

©2013-2022 Tencent Cloud. All rights reserved.

reject

現在の通話を拒否します。着呼側として `onCallReceived()` のコールバックを受信した場合は、この関数を呼び出して通話を拒否することができます。

```
- (void)reject:(TUICallSucc)succ fail:(TUICallFail)fail;
```

ignore

現在の通話を無視します。着呼側として `onCallReceived()` のコールバックを受信した場合は、この関数を呼び出して通話を無視することができ、このとき発呼側は `onUserLineBusy` のコールバックを受信します。備考：業務内にライブストリーミング、ミーティングなどのシーンがある場合、ライブストリーミング/ミーティング中の場合もこの関数を呼び出して通話を無視することができます。

```
- (void)ignore:(TUICallSucc)succ fail:(TUICallFail)fail;
```

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:(TUICallSucc)succ fail:(TUICallFail)fail;
```

inviteUser

今回のグループ通話にユーザーを招待します。ユースケース：グループ通話中のユーザーが自主的に他の人を招待する場合に使用します。

```
- (void)inviteUser:(NSArray<NSString *> *)userIdList params:(TUICallParams *)para
ms succ:(void(^)(NSArray <NSString *> *userIdList))succ fail:(TUICallFail)fail
```

パラメータ	タイプ	意味
userIdList	NSArray	ターゲットユーザーのuserIdリスト
params	TUICallParams	通話パラメータ拡張フィールド。例：カスタムコンテンツのオフラインプッシュ

joinInGroupCall

今回のグループ通話に自主的に参加します。ユースケース：グループ内のユーザーが今回のグループ通話に自主的に参加する場合に使用します。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (void)joinInGroupCall:(TUIRoomId *)roomId groupId:(NSString *)groupId callMedia
Type:(TUICallMediaType)callMediaType succ:(TUICallSucc)succ fail:(TUICallFail)fai
l;
```

パラメータ	タイプ	意味
roomId	TUIRoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
groupId	NSString	今回のグループ通話のグループID
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など

switchCallMediaType

ビデオ通話を音声通話へ切り替えます。

```
- (void)switchCallMediaType:(TUICallMediaType)newType;
```

パラメータ	タイプ	意味
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など

startRemoteView

ビデオ画面に表示するViewオブジェクトを設定します

```
- (void)startRemoteView:(NSString *)userId videoView:(TUIVideoView *)videoView on
Playing:(void(^)(NSString *userId))onPlaying onLoading:(void(^)(NSString *userI
d))onLoading onError:(void(^)(NSString *userId, int code, NSString *errMsg))onErr
or;
```

パラメータ	タイプ	意味
userId	NSString	ターゲットユーザーのuserId
videoView	TUIVideoView	レンダリング対象のビュー

stopRemoteview

リモートユーザーのビデオデータのサブスクリプションを停止します。

```
- (void)stopRemoteView:(NSString *)userId;
```

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	NSString	ターゲットユーザーのuserId

openCamera

カメラをオンにする

```
- (void)openCamera:(TUICallCamera)camera videoView:(TUIVideoView *)videoView succ:(TUICallSucc)succ fail:(TUICallFail)fail;
```

パラメータ	タイプ	意味
camera	TUICallCamera	フロントカメラ/リアカメラ
videoView	TUIVideoView	レンダリング対象のビュー

closeCamera

カメラをオフにします。

```
- (void)closeCamera;
```

switchCamera

フロント/リアカメラを切り替えます。

```
- (void)switchCamera:(TUICallCamera)camera;
```

パラメータ	タイプ	意味
camera	TUICallCamera	フロントカメラ/リアカメラ

openMicrophone

マイクをオンにします

```
- (void)openMicrophone:(TUICallSucc)succ fail:(TUICallFail)fail;
```

closeMicrophone

マイクをオフにします。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (void)closeMicrophone;
```

selectAudioPlaybackDevice

オーディオ再生デバイスを選択します。 現在はヘッドホン、スピーカーをサポートしています。通話のシーンでは、このインターフェースを使用してハンズフリーモードのオン/オフが行えます。

```
- (void)selectAudioPlaybackDevice:(TUIAudioPlaybackDevice)device;
```

パラメータ	タイプ	意味
device	TUIAudioPlaybackDevice	ヘッドホン/スピーカー

setSelfInfo

ユーザーニックネーム、プロフィール画像を設定します。ユーザーニックネームは500バイト以内、ユーザープロフィール画像はURL形式でなければなりません。

```
- (void)setSelfInfo:(NSString * _Nullable)nickName avatar:(NSString * _Nullable)a
vatar succ:(TUICallSucc)succ fail:(TUICallFail)fail;
```

enableMultiDeviceAbility

TUICallEngineのマルチデバイスログインモードをオン/オフします （プレミアム版パッケージのみサポート）

```
- (void)enableMultiDeviceAbility:(BOOL)enable succ:(TUICallSucc)succ fail:(TUICal
lFail)fail;
```

©2013-2022 Tencent Cloud. All rights reserved.

TUICallObserver

最終更新日：：2022-11-21 16:34:36

TUICallObserver APIの概要

TUICallObserverはTUICallEngineに対応するコールバックイベントクラスです。このコールバックによって、関心のあるコールバックイベントを監視することができます。

コールバックイベントの概要

API	説明
onError	通話中のエラーコールバック
onCallReceived	通話リクエストのコールバック
onCallCancelled	通話キャンセルのコールバック
onCallBegin	通話接続のコールバック
onCallEnd	通話終了のコールバック
onCallMediaTypeChanged	通話メディアタイプ変更発生のコールバック
onUserReject	xxxxユーザーによる通話拒否のコールバック
onUserNoResponse	xxxxユーザーの応答なしのコールバック
onUserLineBusy	xxxxユーザーが通話中である場合のコールバック
onUserJoin	xxxxユーザーの通話参加のコールバック
onUserLeave	xxxxユーザーの通話からの退出のコールバック
onUserVideoAvailable	xxxユーザーのビデオストリームの有無のコールバック
onUserAudioAvailable	xxxユーザーのオーディオストリームの有無のコールバック
onUserVoiceVolumeChanged	全ユーザーの音量レベルフィードバックのコールバック
onUserNetworkQualityChanged	全ユーザーのネットワーク品質フィードバックのコールバック

©2013-2022 Tencent Cloud. All rights reserved.

コールバックイベントの詳細

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 * _Nullable)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

onCallReceived

新しい通話リクエストコールバックを受信します。着呼側が受信します。このイベントを監視することで、通話応答画面を表示するかどうかを決定できます。

```
- (void)onCallReceived:(NSString *)callerId calleeIdList:(NSArray<NSString *> *)calleeIdList groupId:(NSString * _Nullable)groupId callMediaType:(TUICallMediaType)callMediaType
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callerId	NSString	発呼側ID（招待者）
calleeIdList	NSArray	着呼側IDリスト（被招待者）
groupId	NSString	グループ通話ID
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話など

onCallCancelled

©2013-2022 Tencent Cloud. All rights reserved.

今回の通話が発呼側からキャンセルされたことを表します（キャンセルの原因は発呼側の自主的なキャンセル、または通話タイムアウトによるキャンセルの両方の可能性があります）。着呼側が受信します。このイベントを監視することで、未応答通話などに類似した表示ロジックを実現できます。

```
- (void)onCallCancelled:(NSString *)callerId;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
callerId	NSString	キャンセルしたユーザーのID

onCallBegin

通話接続を表します。発呼側と着呼側がどちらも受信できます。このイベントを監視することで、クラウドレコーディング、コンテンツ審査などのフローを開始できます。

```
- (void)onCallBegin:(TUIRoomId *)roomId callMediaType:(TUICallMediaType)callMediaType callRole:(TUICallRole)callRole;
```

パラ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	TUIRoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話
callRole	TUICallRole	ロール。列挙タイプ：発呼側、着呼側

onCallEnd

通話の終了を表します。発呼側と着呼側がどちらも受信できます。このイベントを監視することで、通話時間、通話タイプなどの情報を表示したり、クラウドのレコーディングフローを停止したりすることができます。

```
- (void)onCallEnd:(TUIRoomId *)roomId callMediaType:(TUICallMediaType)callMediaType callRole:(TUICallRole)callRole totalTime:(float)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

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomId	TUIRoomId	今回の通話のオーディオビデオルームID。現在は数字のルームナンバーのみサポートしています。文字列のルームナンバーは今後のバージョンでサポート予定です
callMediaType	TUICallMediaType	通話のメディアタイプ。ビデオ通話、音声通話
callRole	TUICallRole	ロール。列挙タイプ：発呼側、着呼側
totalTime	float	今回の通話時間

注意：
クライアントのイベントは一般的にすべて、プロセスキルなどの異常イベントに伴って消失します。通話時間の監視によって料金計算などのロジックを完成させたい場合は、REST APIを使用してこの種のフローを完成させることをお勧めします。

onCallMediaTypeChanged

通話のメディアタイプに変更が発生したことを表します。

```
- (void)onCallMediaTypeChanged:(TUICallMediaType)oldCallMediaType newCallMediaTyp
e:(TUICallMediaType)newCallMediaType;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
oldCallMediaType	TUICallMediaType	変更前の通話タイプ
newCallMediaType	TUICallMediaType	変更後の通話タイプ

onUserReject

通話が拒否された場合のコールバックです。1v1通話では、発呼側のみが拒否のコールバックを受信します。グループ通話では、すべての被招待者がこのコールバックを受信することができます。

```
- (void)onUserReject:(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

©2013-2022 Tencent Cloud. All rights reserved.

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

onUserNoResponse

相手方が応答しない場合のコールバック。

```
- (void)onUserNoResponse:(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	応答しないユーザーのID

onUserLineBusy

通話中である場合のコールバック。

```
- (void)onUserLineBusy:(NSString *)userId;
```

onUserJoin

今回の通話に参加したユーザーがいる場合のコールバック。

```
- (void)onUserJoin:(NSString *)userId;
```

onUserLeave

今回の通話から退出したユーザーがいる場合のコールバック。

```
- (void)onUserLeave:(NSString *)userId;
```

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。

```
- (void)onUserAudioAvailable:(NSString *)userId isAudioAvailable:(BOOL)isAudioAva
ilable;
```

パラメータは下表に示すとおりです。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	NSString	ユーザーID
isAudioAvailable	BOOL	ユーザーのオーディオ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。

```
- (void)onUserVideoAvailable:(NSString *)userId isVideoAvailable:(BOOL)isVideoAva
ilable;
```

パラ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	通話ユーザーID
isVideoAvailable	BOOL	ユーザーのビデオが使用可能かどうか

onUserVoiceVolumeChanged

ユーザーの通話音量のコールバック。

```
- (void)onUserVoiceVolumeChanged:(NSDictionary <NSString *, NSNumber *> *)volumeM
ap;
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
volumeMap	NSDictionary <nsstring *,="" nsnumber="" *="">	ボリュームメーター。各useridに応じて、対応する音量レベルを取得できます。音量の最小値は0、最大値は100です
volumeMap	NSDictionary	ボリュームメーター。各userIdに応じて、対応する音量レベルを取得できます。音量の最小値は0、最大値は100です

onUserNetworkQualityChanged

ユーザーのネットワーク品質のコールバック。

```
- (void)onUserNetworkQualityChanged:(NSArray<TUINetworkQualityInfo *> *)networkQu
alityList;
```

©2013-2022 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
networkQualityList	NSArray	ネットワーク状態。各userIdにつき、対応するユーザーの現在のネットワーク品質を取得できます

©2013-2022 Tencent Cloud. All rights reserved.

Web
API概要

最終更新日：：2022-11-21 16:34:37

TUICallEngine (UIインターフェースなし)

TUICallEngine APIはオーディオビデオ通話コンポーネントの**UIインターフェースがない**ものです。このAPIを使用し、業務ニーズに応じてパッケージをカスタマイズすることができます。

API	説明
createInstance	TUICallEngineインスタンスの作成（シングルトンモード）
destroyInstance	TUICallEngineインスタンスの破棄（シングルトンモード）
on	イベントの監視
off	イベント監視のキャンセル
login	ログインインターフェース
logout	ログアウトインターフェース
setSelfInfo	ユーザーニックネームおよびプロフィール画像の設定
call	C2C通話への招待
groupCall	グループチャット通話への招待
accept	通話応答
reject	通話拒否
hangup	通話終了
switchCallingType	オーディオビデオ通話の切り替え
startRemoteView	リモート画面レンダリングの起動
stopRemoteView	リモート画面レンダリングの停止
startLocalView	ローカル画面レンダリングの起動
stopLocalView	ローカル画面レンダリングの停止

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
openCamera	カメラの起動
closeCamara	カメラの終了
openMicrophone	マイクをオンにする
closeMicrophone	マイクをオフにする
setMicMute	デバイスマイクのミュートの有無
setVideoQuality	ビデオ画質の設定
getDeviceList	デバイスリストの取得
switchDevice	カメラまたはマイクデバイスの切り替え

イベントタイプの定義

TUICallEventはTUICallEngineに対応するコールバックイベントクラスです。このコールバックによって、関心のあるコールバックイベントを監視することができます。

EVENT	説明
TUICallEvent.ERROR	SDKの内部でエラーが発生しました
TUICallEvent.SDK_READY	SDKがready状態に入ったときにこのコールバックを受信します
TUICallEvent.KICKED_OUT	重複ログインです。このコールバックを受信した場合は、ルームからの強制退出を意味します
TUICallEvent.USER_ACCEPT	応答したユーザーがいる場合に、このコールバックを受信します
TUICallEvent.USER_ENTER	通話への参加に同意したユーザーがいる場合に、このコールバックを受信します
TUICallEvent.USER_LEAVE	通話からの退出に同意したユーザーがいる場合に、このコールバックを受信します
TUICallEvent.REJECT	ユーザーが通話を拒否
TUICallEvent.NO_RESP	招待したユーザーからの応答なし

©2013-2022 Tencent Cloud. All rights reserved.

EVENT	説明
TUICallEvent.LINE_BUSY	招待者が通話中
TUICallEvent.CALLING_TIMEOUT	被招待者が受信します。このコールバックを受信した場合は、今回の通話に応答せずタイムアウトしたことを意味します
TUICallEvent.USER_VIDEO_AVAILABLE	リモートユーザーによるカメラのオン/オフがあった場合に、このコールバックを受信します
TUICallEvent.USER_AUDIO_AVAILABLE	リモートユーザーによるマイクのオン/オフがあった場合に、このコールバックを受信します
TUICallEvent.USER_VOICE_VOLUME	リモートユーザーがスピーカーの音量調整を行った場合に、このコールバックを受信します
TUICallEvent.GROUP_CALL_INVITEE_LIST_UPDATE	グループチャットの招待リストが更新された場合にこのコールバックを受信します
TUICallEvent.INVITED	通話に招待されました
TUICallEvent.CALLING_CANCEL	被招待者が受信します。このコールバックを受信した場合は、今回の通話がキャンセルされたことを意味します
TUICallEvent.CALLING_END	このコールバックを受信した場合は、今回の通話が終了したことを意味します
TUICallEvent.DEVICED_UPDATED	デバイスリストが更新された場合にこのコールバックを受信します
TUICallEvent.CALL_TYPE_CHANGED	通話タイプが切り替わった場合にこのコールバックを受信します

©2013-2022 Tencent Cloud. All rights reserved.

TUICallKit

最終更新日：：2023-02-10 18:26:06

APIの概要

TUICallKit APIはオーディオビデオ通話コンポーネントの**UIインターフェース付き**のものです。TUICallKit APIを使用することで、WeChatのようなオーディオビデオ通話シーンをシンプルなインターフェースでスピーディーに実現できます。より詳細なアクセス手順については、TUICallKitクイックアクセスをご参照ください。

APIの概要

`<TUICallKit/>` ：UI通話コンポーネントを主体としています
`<TUICallKitMini/>` ：UI通話フローティングウィンドウ
で、 `<TUICallKit/>allowedMinimized` が `true` に設定されている場合、 `<TUICallKitMini/>` はページ内に配置する必要があります
`TUICallKitServer` : 通話インスタンス、メンバーの関数です。
init TUICallKitの初期化
call 1v1通話の開始
groupCall グループ通話の開始
destroyed TUICallKitの破棄

`<TUICallKit/>` APIの詳細

属性

パラメータ	説明	タイプ	入力必須かどうか	デフォルト値
allowedMinimized	最小化を許可するかどうか。最小化ボタンは非表示	boolean	いいえ	false
allowedFullScreen	フルスクリーンを許可するかどうか。フルスクリーンボタンは非表示	boolean	いいえ	true

方法

©2013-2022 Tencent Cloud. All rights reserved.

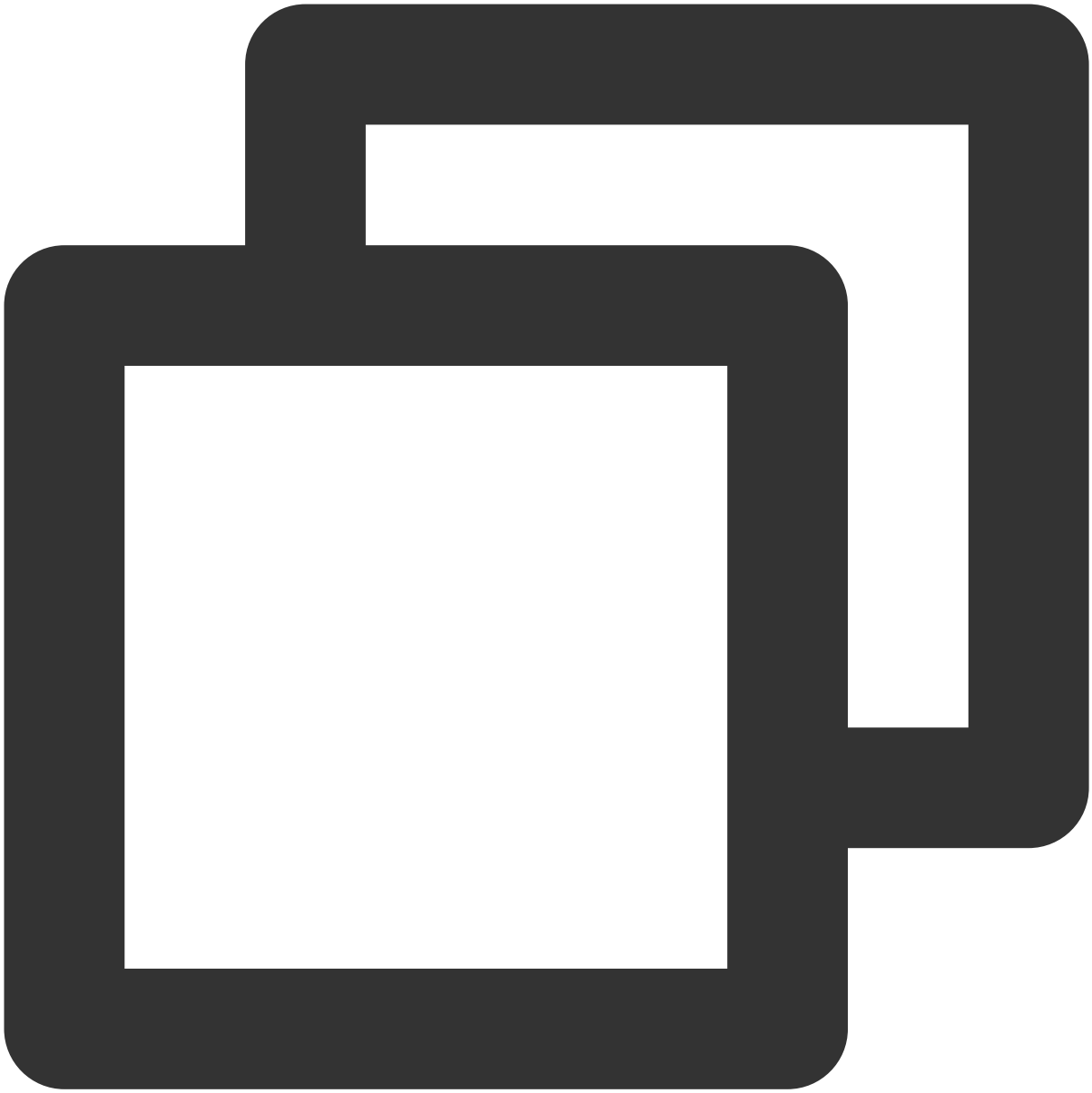
パラメータ	説明	タイプ	入力必須かどうか	デフォルト値
beforeCalling	電話をかける前と通話の招待を受信する前にこの関数を実行	function(type、error)	いいえ	-
afterCalling	通話終了後にこの関数を実行	function()	いいえ	-
onMinimized	コンポーネントを最小化状態に切り替えた時にこの関数を実行	function(oldStatus、newStatus)	いいえ	-

`<TUICallKitMini/>` APIの詳細

なし

サンプルコード

©2013-2022 Tencent Cloud. All rights reserved.

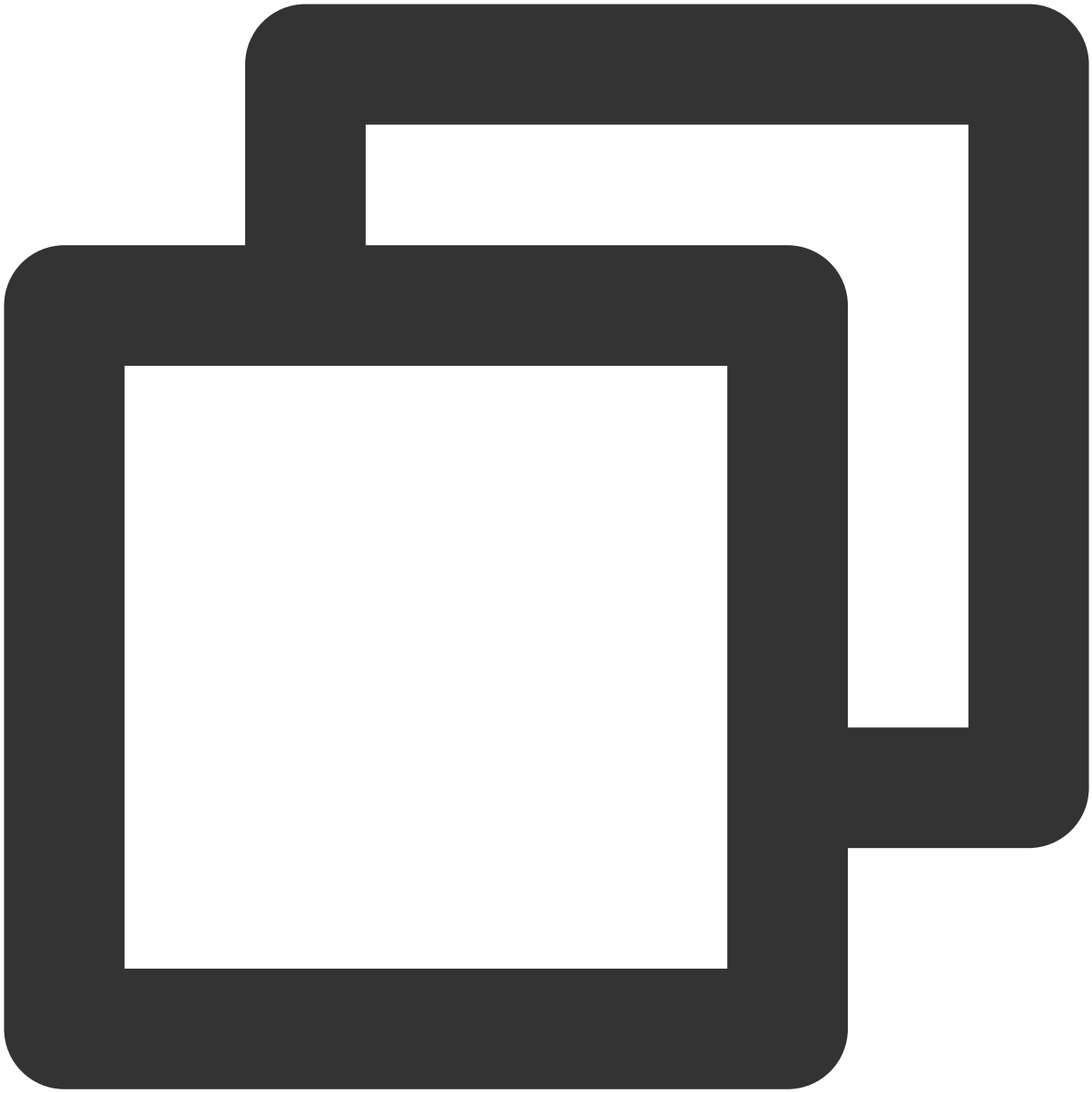


```
/**
* beforeCalling
 * @param { string } type値は"invited"です | "call" | "groupCall"は、 着信と発信の区別
 * @param { number } error.code エラーコード
 * @param { string } error.type エラータイプ
 * @param { string } error.code エラー情報
 */
function beforeCalling(type, error) {
  console.log("通話前にこの関数を実行、タイプ: "、type、error);
}
function afterCalling() {
```

©2013-2022 Tencent Cloud. All rights reserved.

```
  console.log("通話後にこの関数を実行");
}
/**
 * onMinimized
 * @param { boolean } oldStatus
 * @param { boolean } newStatus
 */
function onMinimized(oldStatus, newStatus) {
  if (newStatus === true) {
    console.log("TUICallKit 最小化状態に進む");
  }else{
    console.log("TUICallKit 最小化状態を終了");
  }
}
```

©2013-2022 Tencent Cloud. All rights reserved.



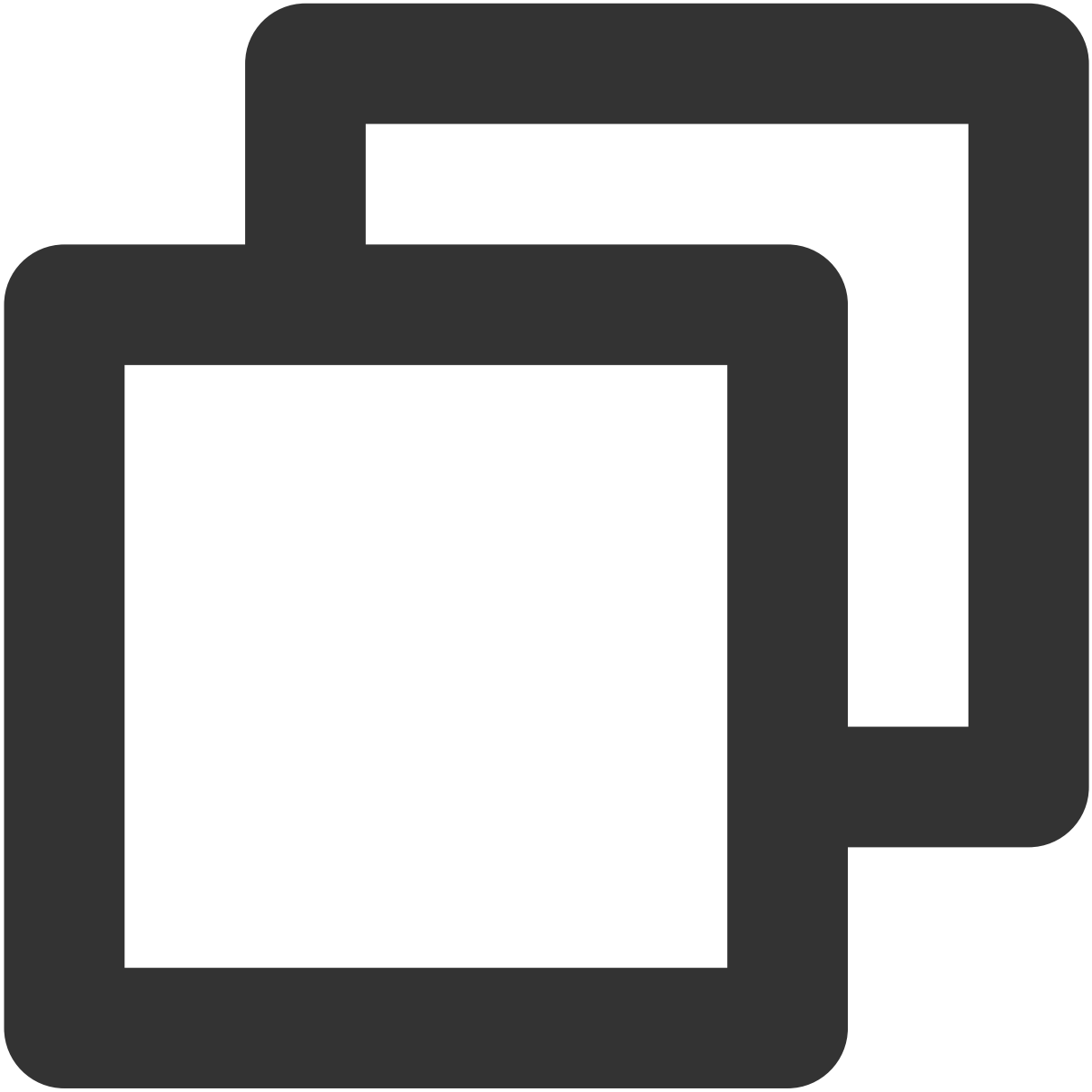
```
<TUICallKit
  :allowedMinimized="true"
  :allowedFullScreen="true"
  :beforeCalling="beforeCalling"
  :afterCalling="afterCalling"
  :onMinimized="onMinimized"
/>
<TUICallKitMini />
```

©2013-2022 Tencent Cloud. All rights reserved.

TUICallKitServer APIの詳細

init

TUICallKitの初期化はcall、groupCallの前に行う必要があります。



```
import { TUICallKitServer } from "./components/TUICallKit/Web";
TUICallKitServer.init({
  SDKAppID,
  userID,
  userSig,
```

©2013-2022 Tencent Cloud. All rights reserved.

```
    tim,
});
```

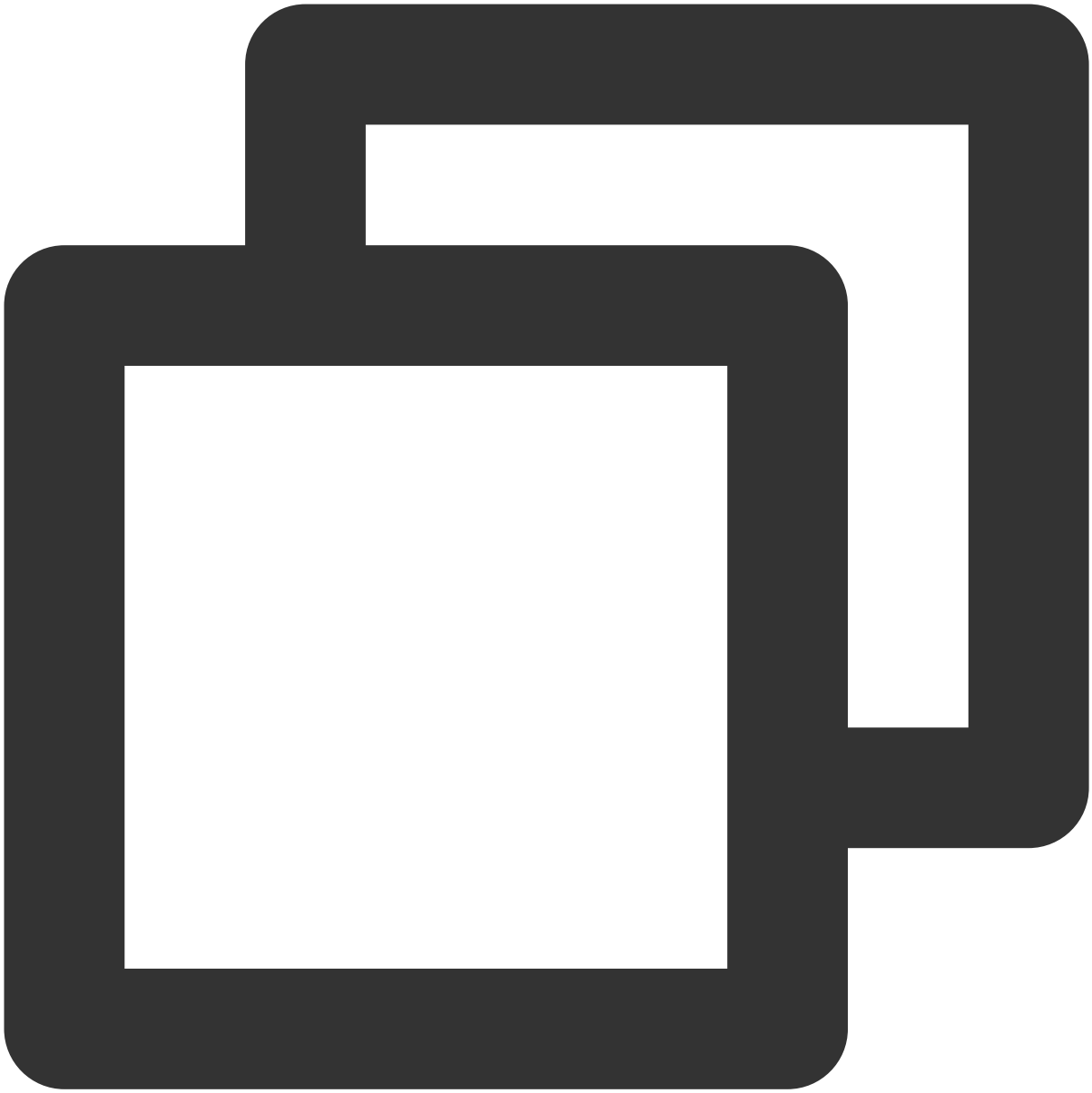
パラメータは下表に示すとおりです。

パラメータ	タイプ	入力必須かどうか	意味
SDKAppID	Number	はい	IMアプリケーションのSDKAppID
userID	String	はい	現在のユーザーID、文字列タイプでは、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	はい	Tencent Cloudによって設計されたセキュリティ保護署名です。取得方法については、UserSigをご参照ください
TIMインスタンス	Any	いいえ	timパラメータはサービス内にすでに存在するTIMインスタンスに適用され、TIMインスタンスの一意性を保証します

call

電話をかけます（1v1通話）。

©2013-2022 Tencent Cloud. All rights reserved.



```
import { TUICallKitServer } from "./components/TUICallKit/Web";
TUICallKitServer.call({
  userID: 'jack',
  type: 1,
});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	入力必須かどうか	意味

©2013-2022 Tencent Cloud. All rights reserved.

userID	String	はい	ターゲットユーザーのuserId
type	Number	はい	通話のメディアタイプ。音声通話(type = 1)、ビデオ通話(type = 2)
timeout	Number	いいえ	通話のタイムアウト時間。0はタイムアウトしていないことを意味し、単位はs（秒）（オプション） - デフォルト 30s
offlinePushInfo	Object	いいえ	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）-- tsignaling バージョンは >= 0.8.0が必要です

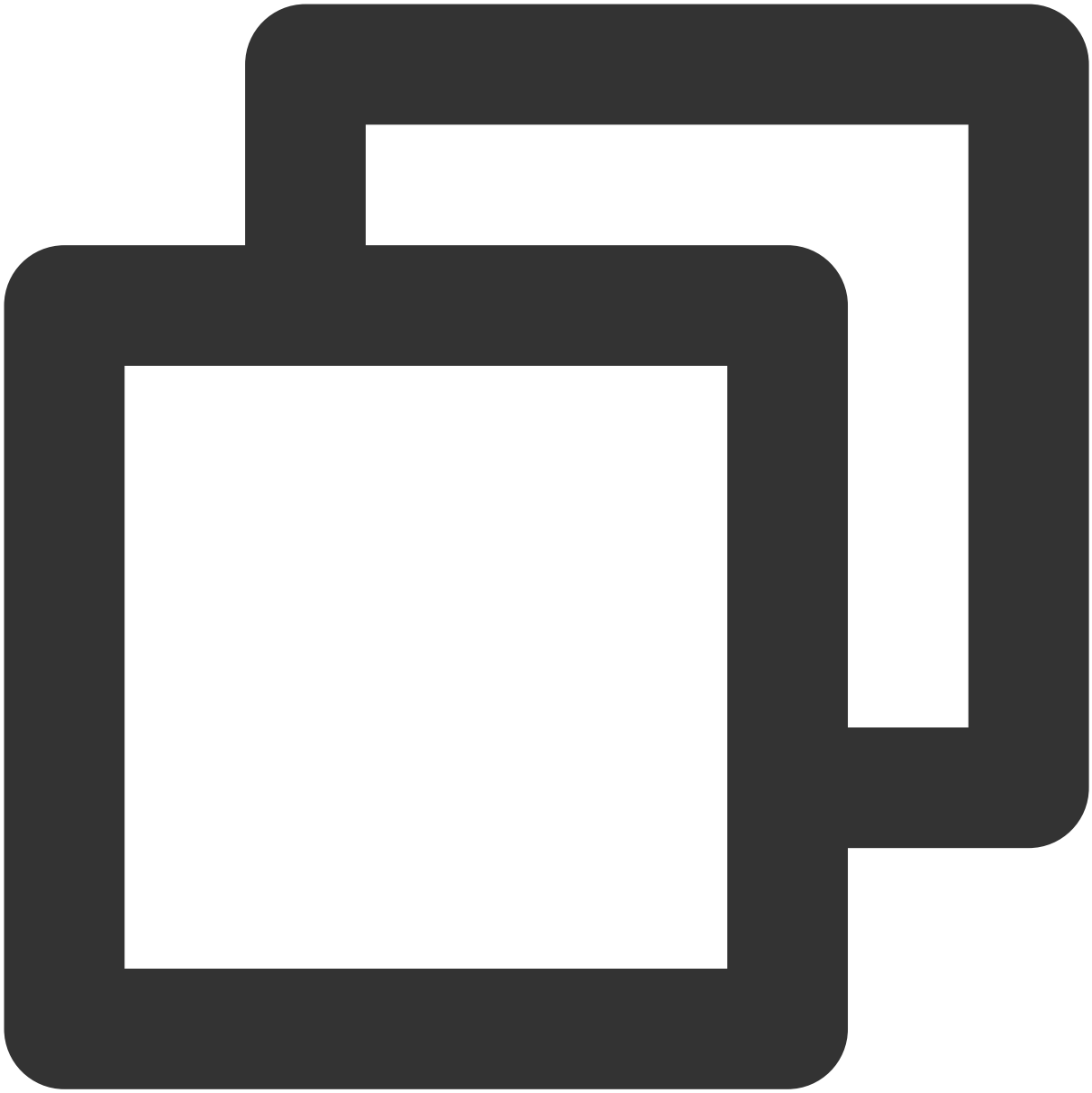
そのうち `offlinePushInfo` について

パラメータ	タイプ	入力必須かどうか	意味
offlinePushInfo.title	String	いいえ	タイトルのオフラインプッシュ（オプション）
offlinePushInfo.description	String	いいえ	コンテンツのオフラインプッシュ（オプション）
offlinePushInfo.androidOPPOChannelID	String	いいえ	オフラインプッシュでは、OPPO携帯（8.0およびそれ以降のシステム）のチャンネルIDを設定します（オプション）
offlinePushInfo.extension	String	いいえ	オフラインプッシュによるコンテンツのパススルー（オプション）（tsignalingバージョン >= 0.9.0）

groupCall

グループ通話を開始します。

©2013-2022 Tencent Cloud. All rights reserved.



```
import { TUICallKitServer } from "./components/TUICallKit/Web";
TUICallKitServer.groupCall({
  userIDList: ['jack', 'tom'],
  groupID: 'xxx',
  type: 1,
});
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	入力必須かど	意味

©2013-2022 Tencent Cloud. All rights reserved.

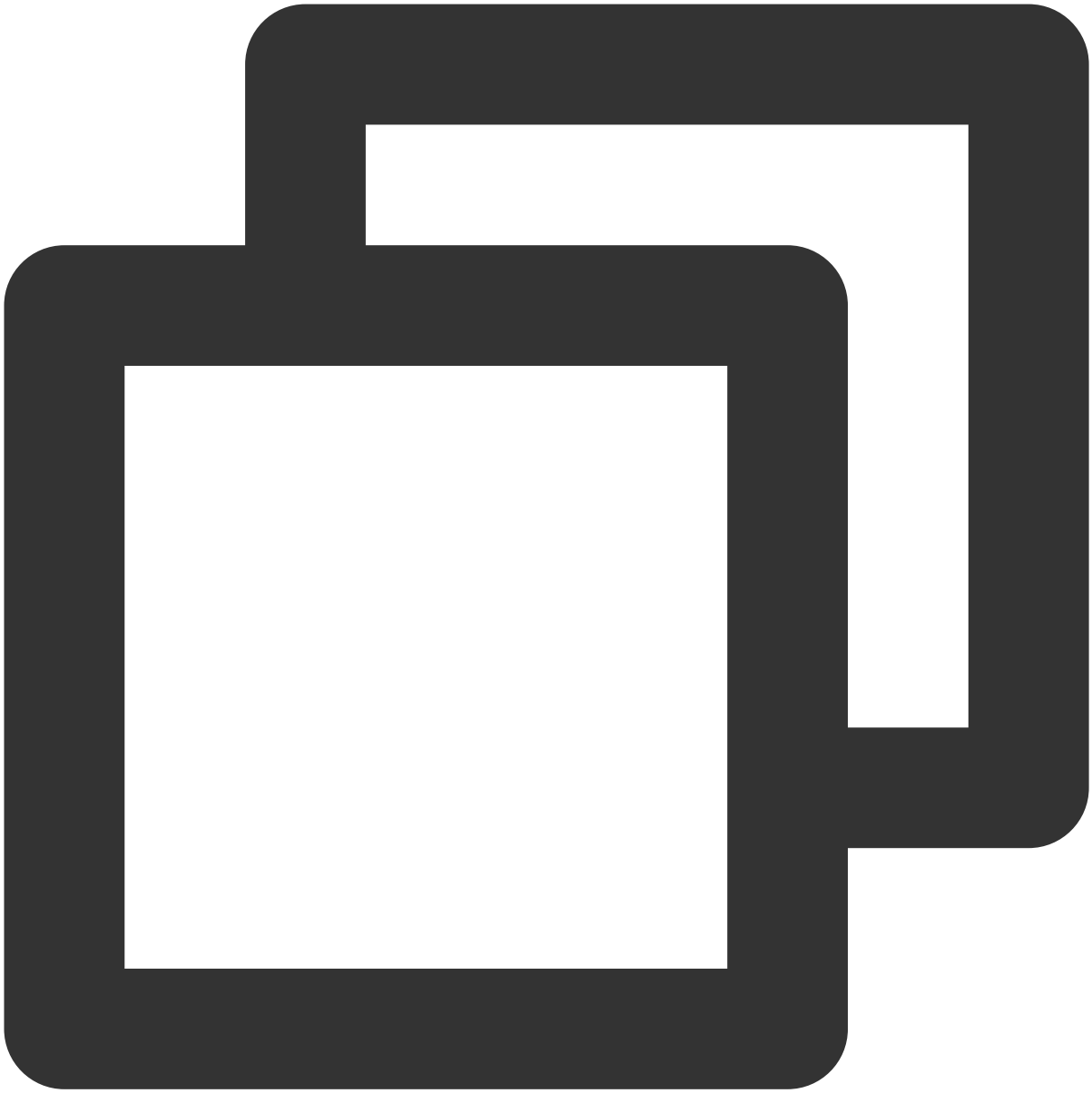
		うか	
userIDList	Array	はい	招待リストメンバーリスト
type	Number	はい	通話のメディアタイプ。音声通話(type = 1)、ビデオ通話(type = 2)
groupID	String	はい	グループIDの呼び出し
timeout	Number	いいえ	通話のタイムアウト時間。0はタイムアウトしていないことを意味し、単位はs（秒）（オプション） - デフォルト 30s
offlinePushInfo	Object	いいえ	メッセージのオフラインプッシュをカスタマイズします（オプション）-- tsignaling バージョンは >= 0.8.0が必要です

そのうち `offlinePushInfo` は、 `call` インターフェースと一致します。

destroyed

TUICallKitを破棄します。

©2013-2022 Tencent Cloud. All rights reserved.



```
import { TUICallKitServer } from "./components/TUICallKit/Web";
TUICallKitServer.destroyed();
```

©2013-2022 Tencent Cloud. All rights reserved.

TUICallEngine

最終更新日：：2023-03-06 09:48:59

TUICallEngine APIの概要

TUICallEngine APIはオーディオビデオ通話コンポーネントの**UIインターフェースがない**ものです。

API概要

API	説明
createInstance	TUICallEngineのインスタンス（シングルトンモード）
destroyInstance	TUICallEngineのインスタンス（シングルトンモード）
on	イベントの監視
off	イベントの監視をキャンセルします
login	インターフェースにログインします
logout	インターフェースからログアウトします
setSelfInfo	ユーザーニックネームおよびプロフィール画像を設定し
call	C2C通話に招待します
groupCall	グループチャット通話に招待します
accept	通話に応答します
reject	通話を拒否します
hangup	通話を終了します
inviteUser	グループ通話で、他の人を招待します
joinInGroupCall	現在のグループ通話に自主的に参加します
switchCallMediaType	現在の通話タイプを切り替えます
startRemoteView	リモート画面のレンダリングを起動します

©2013-2022 Tencent Cloud. All rights reserved.

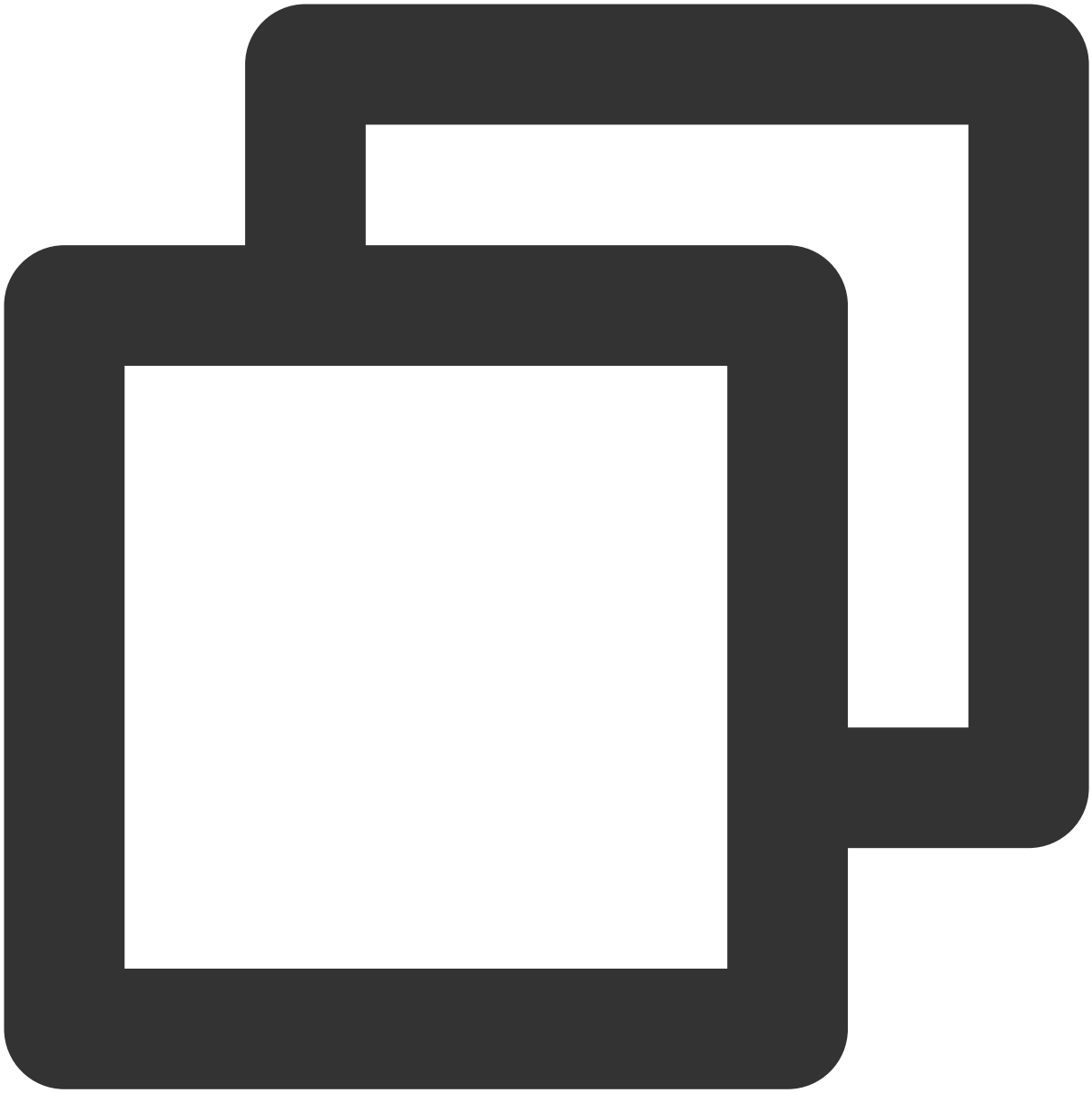
stopRemoteView	リモート画面のレンダリングを停止します
startLocalView	ローカル画面のレンダリングを起動します
stopLocalView	ローカル画面のレンダリングを停止します
openCamera	カメラをオンにします
closeCamara	カメラをオフにします
openMicrophone	マイクをオンにします
closeMicrophone	マイクをオフにします
setVideoQuality	ビデオ画質を設定します
getDeviceList	デバイスリストを取得します
switchDevice	カメラまたはマイクデバイスを切り替えます
enableAIVoice	AIノイズリダクションのオン/オフ

APIの詳細

createInstance

TUICallEngineのシングルトンを作成します。

©2013-2022 Tencent Cloud. All rights reserved.



```
const tuiCallEngine = TUICallEngine.createInstance({
    SDKAppID: 0, // アクセス時に、0をIMアプリケーションのSDKAppIDに置き換える必要があります
    tim: tim          // 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、T
});
```

パラメータは下表に示すとおりです

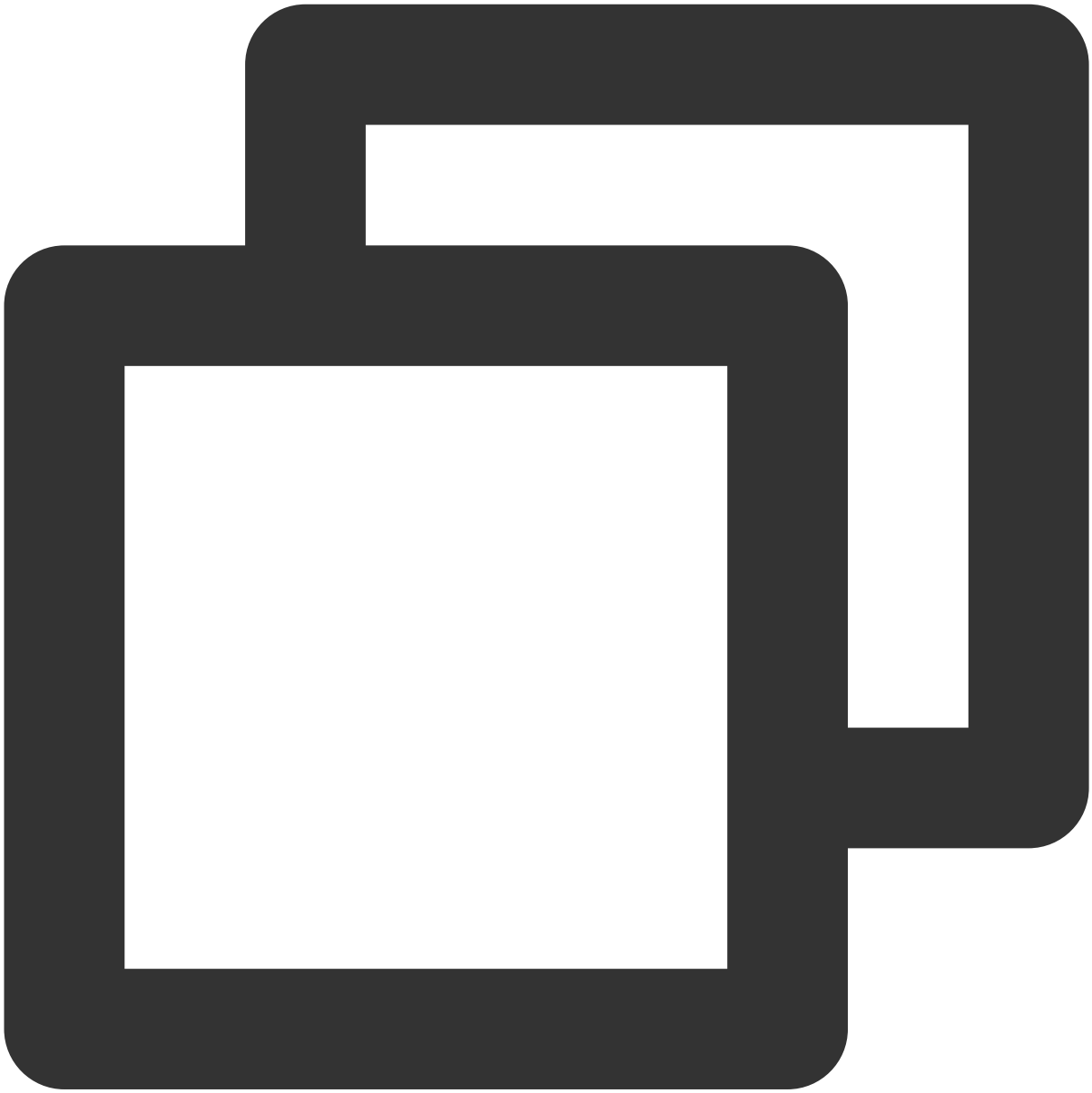
パラメータ	タイプ	意味
SDKAppID	Number	Instant Messagingアプリケーション

©2013-2022 Tencent Cloud. All rights reserved.

tim	Any	TIMインスタンス（オプション）

destroyInstance

TUICallEngineのシングルトンを破棄します。

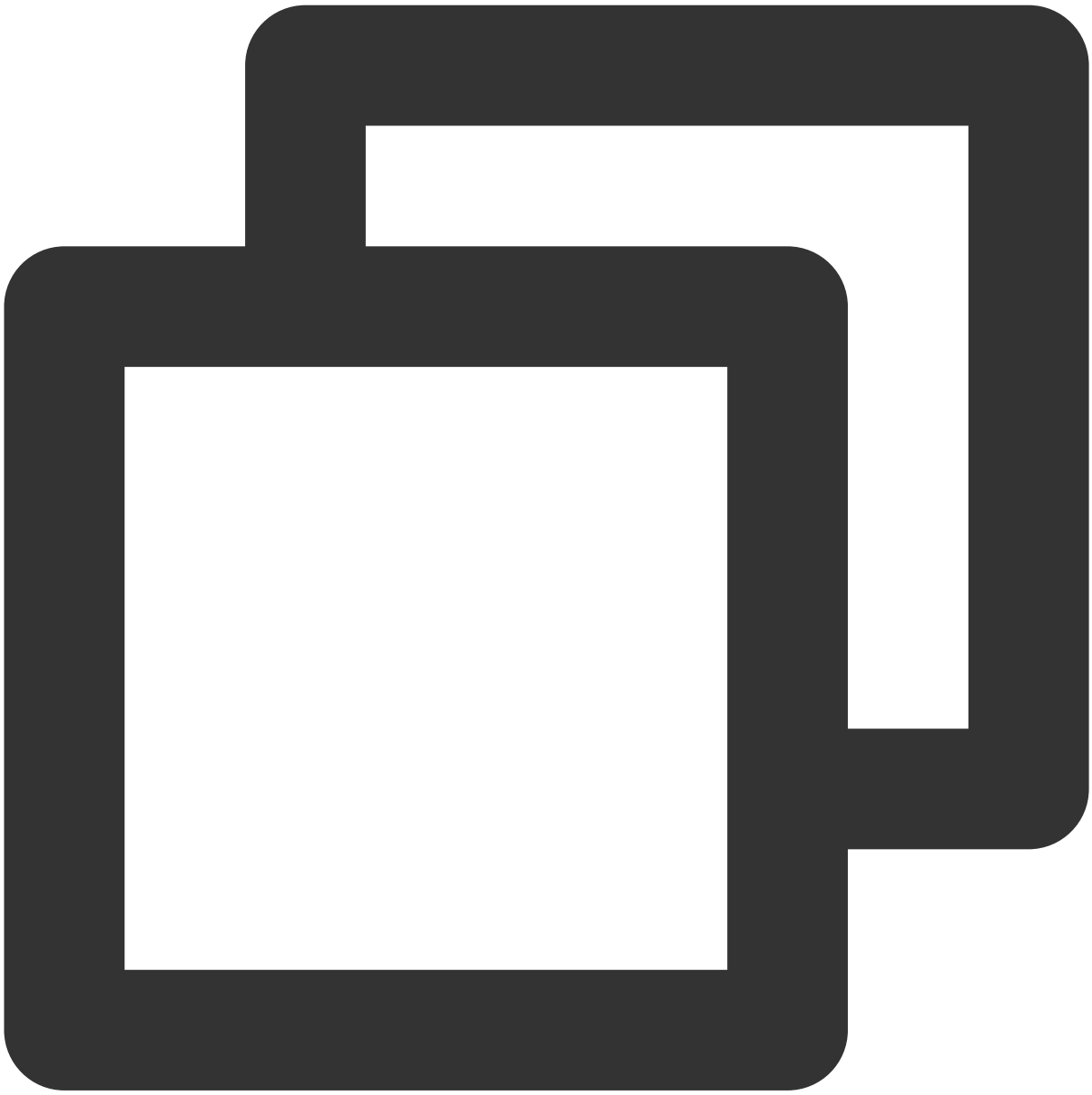


```
tuiCallEngine.destroyInstance().then(() => {
    //success
}).catch(error => {
    console.warn('destroyInstance error:', error);
});
```

©2013-2022 Tencent Cloud. All rights reserved.

on

イベントを監視します。



```
let onError = function(error) {
    console.log(error);
};
tuiCallEngine.on(TUICallEvent.ERROR, onError, this);
```

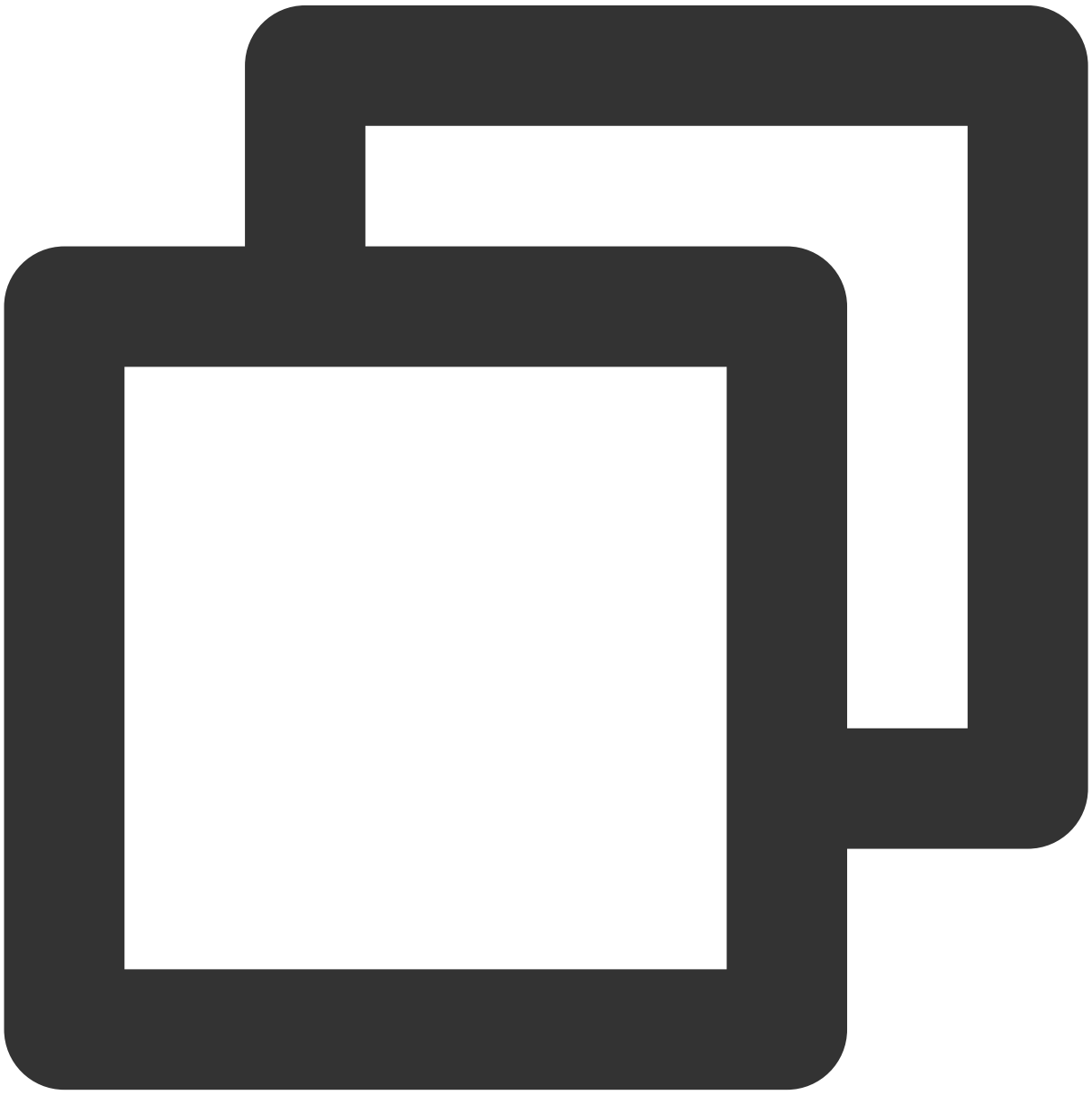
パラメータは下表に示すとおりです

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
eventName	String	イベント名
callback	function	イベントレスポンスコールバック
context	Any	callback実行希望時のコンテキスト

off

イベントの監視をキャンセルします。



©2013-2022 Tencent Cloud. All rights reserved.

```
let onError = function(error) {
    console.log(error);
};
tuiCallEngine.off(TUICallEvent.ERROR, onError, this);
```

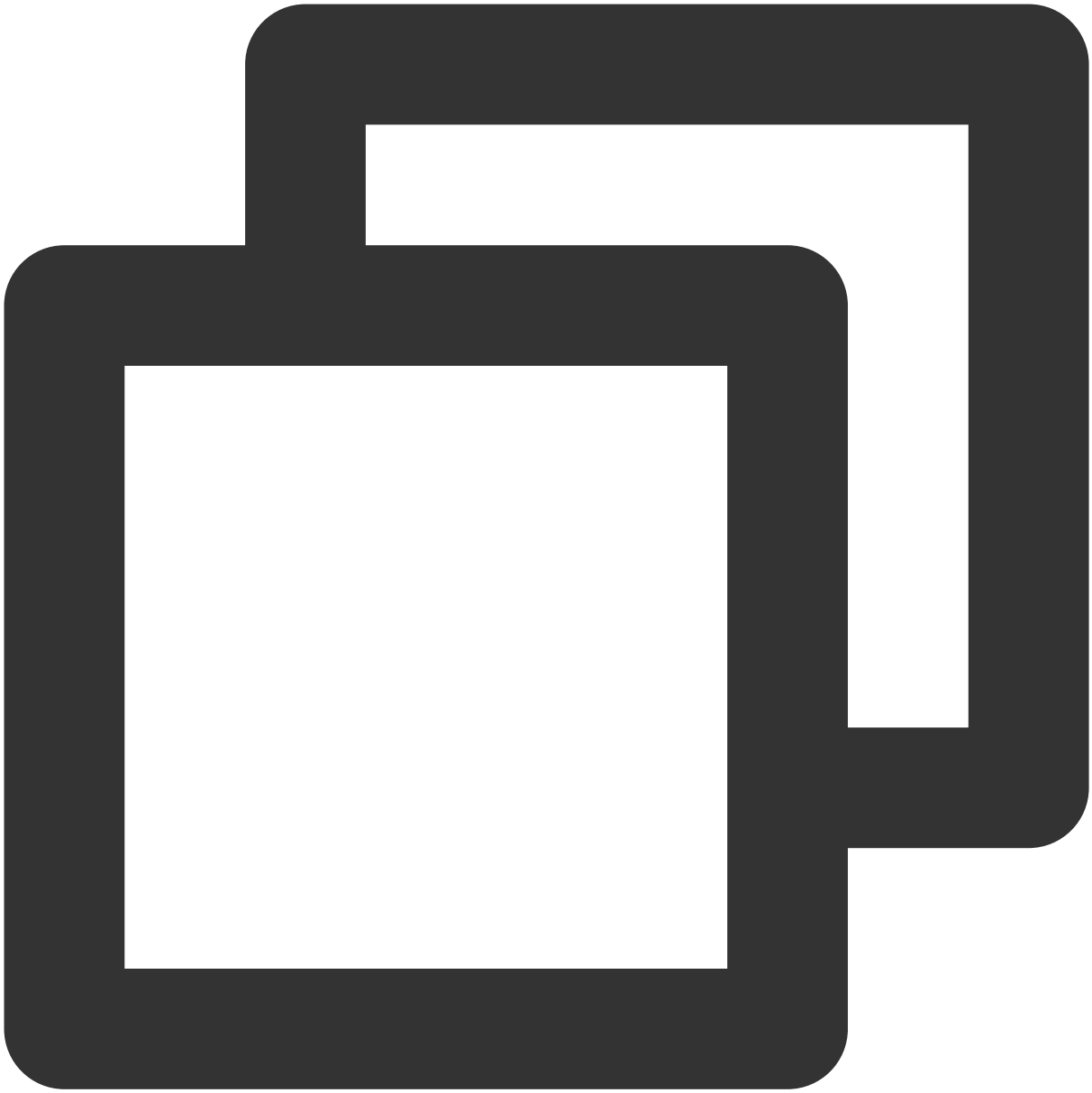
パラメータは下表に示すとおりです

パラメータ	タイプ	意味
eventName	String	イベント名
callback	function	イベントレスポンスコールバック
context	Any	callback実行希望時のコンテキスト

login

インターフェースにログインします。

©2013-2022 Tencent Cloud. All rights reserved.



```
const params = {
    userID: 'john', // your userID
    userSig: 'xxxx', // 'your userSig'
    assetsPath: 'https://xx/'
};
let promise = tuiCallEngine.login(params);
promise.then(() => {
    //success
}).catch(error => {
    console.warn('login error:', error);
});
```

©2013-2022 Tencent Cloud. All rights reserved.

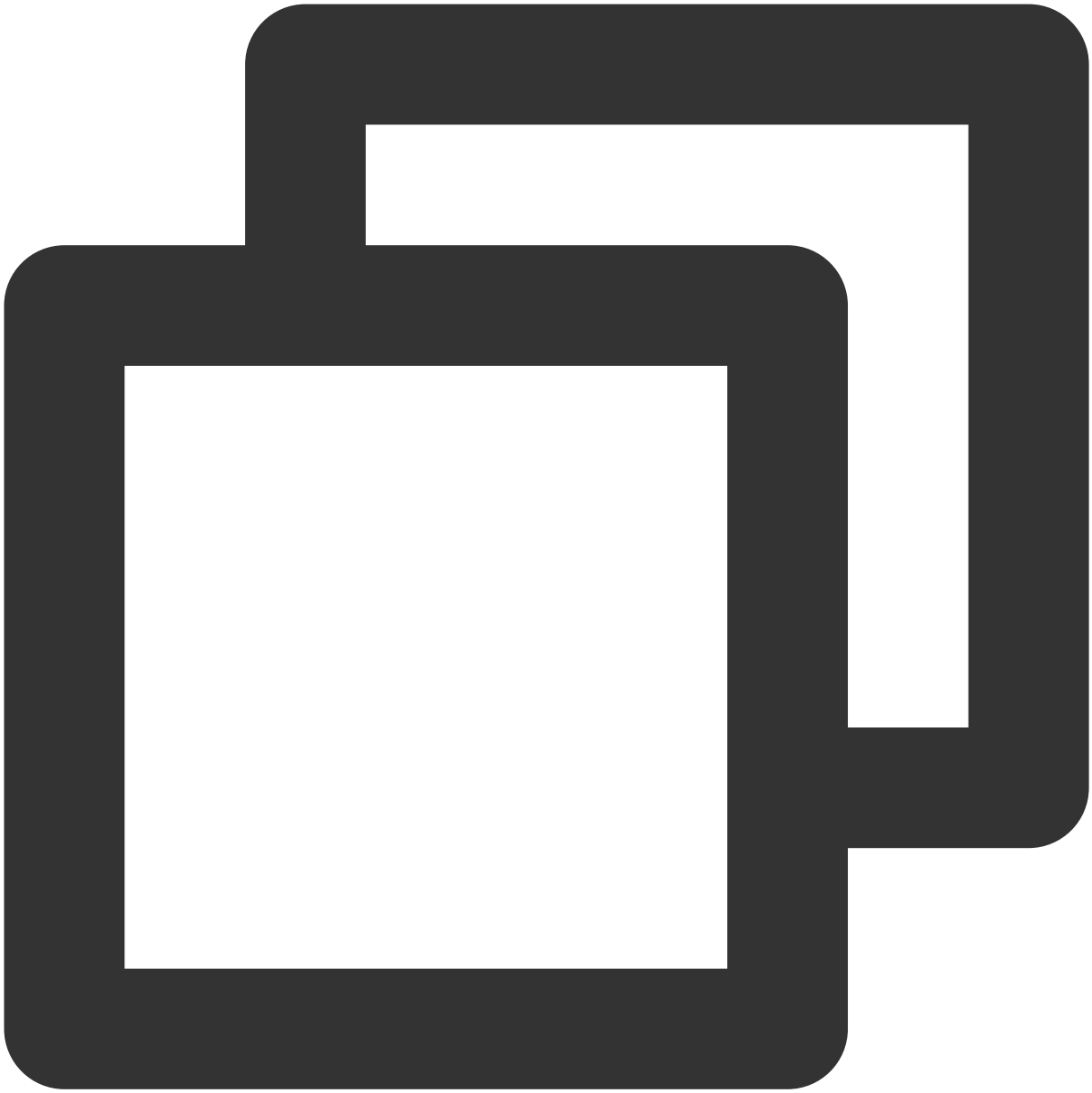
パラメータは下表に示すとおりです

パラメータ	タイプ	意味
userID	String	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）、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
assetsPath	String	AIノイズリダクションの依存関係denoiser-wasm.js ファイルがデプロイされるCDNまたは静的リソースサーバーへのパス。詳細は参照ください。

logout

インターフェースからログアウトします。

©2013-2022 Tencent Cloud. All rights reserved.

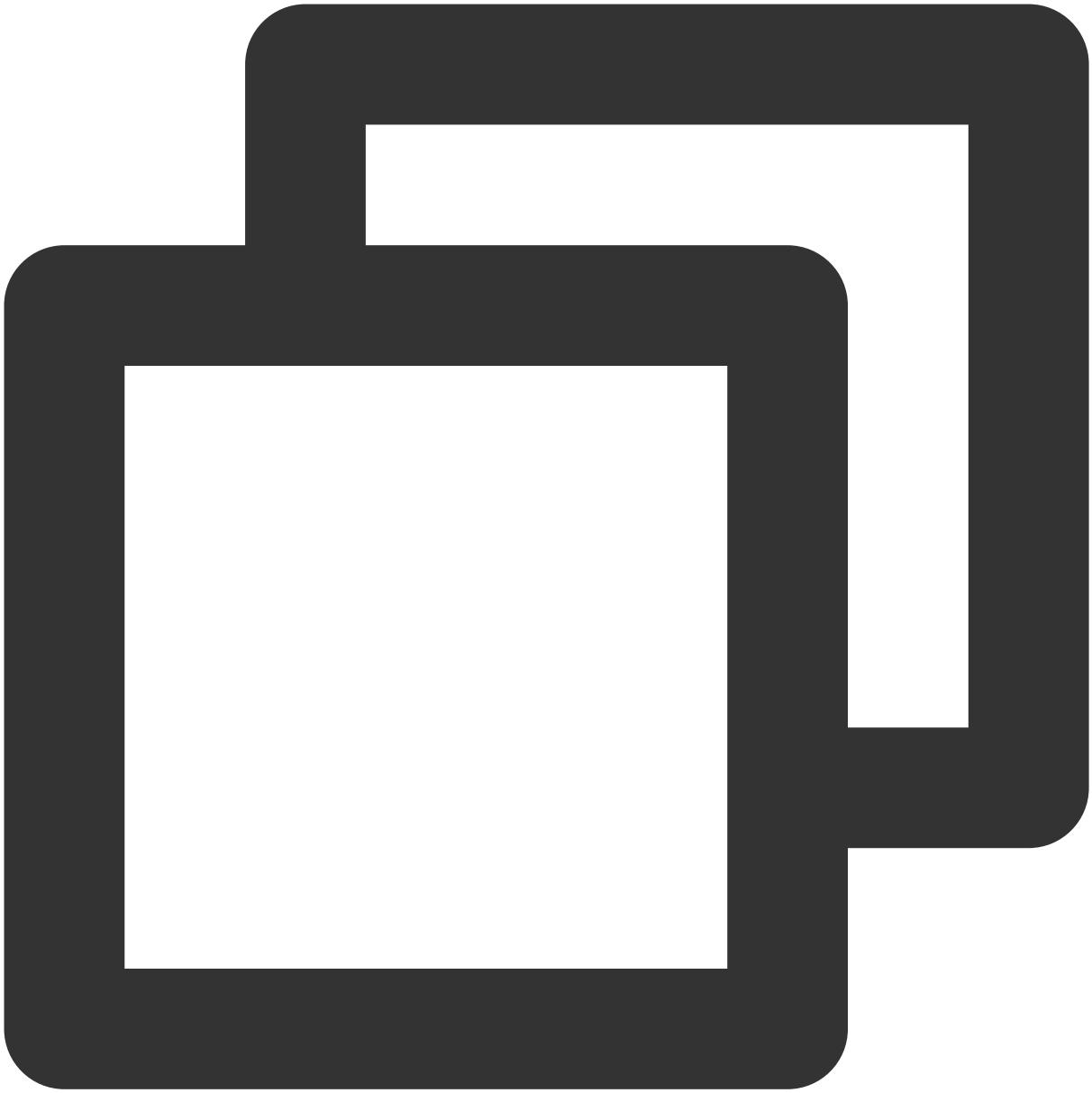


```
let promise = tuiCallEngine.logout();
promise.then(() => {
    //success
}).catch(error => {
    console.warn('logout error:', error);
});
```

setSelfInfo

ユーザーニックネームおよびプロフィール画像を設定します。

©2013-2022 Tencent Cloud. All rights reserved.



```
let promise = tuiCallEngine.setSelfInfo({
    nickName: 'video',
    avatar:'http(s)://url/to/image.jpg'
});
promise.then(() => {
    //success
}).catch(error => {
    console.warn('setSelfInfo error:', error);
});
```

パラメータは下表に示すとおりです

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
nickName	String	ニックネー
avatar	String	プロフィー

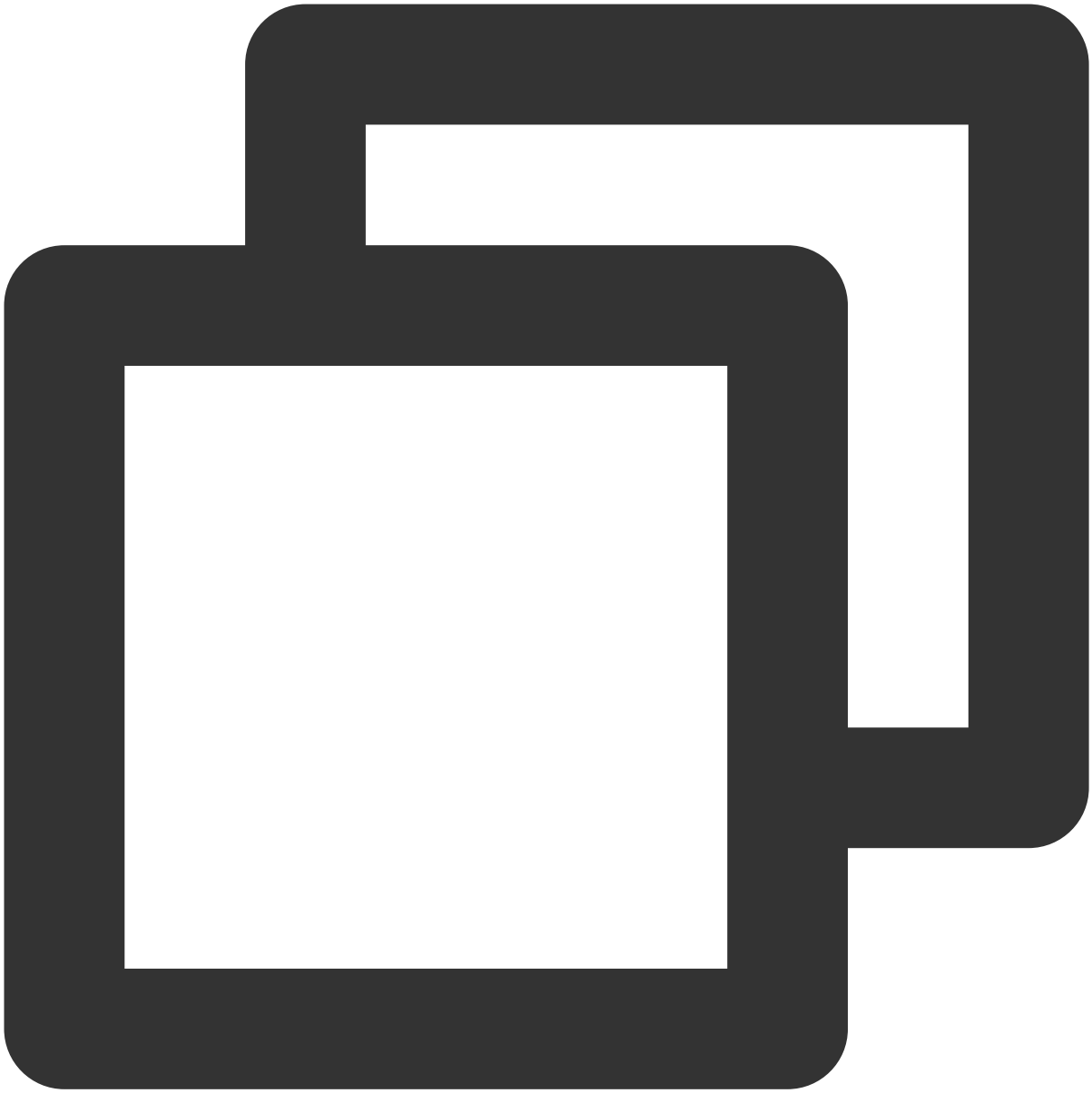
call

C2Cの通話に招待します。被招待者はTUICallEvent.INVITEDイベントを受信します。

注意

オフラインプッシュは端末（AndroidまたはiOS)にのみ適用され、WebおよびWeChat Mini Programではサポートされません。

©2013-2022 Tencent Cloud. All rights reserved.



```
let promise = tuiCallEngine.call({
    userID: 'user1',
    type: 1,
});
promise.then(() => {
    //success
}).catch(error => {
    console.warn('call error:', error);
});
```

パラメータは下表に示すとおりです

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userID	String	招待された側のuserID
type	Number	0-不明、1-音声通話、2-ビデオ通話

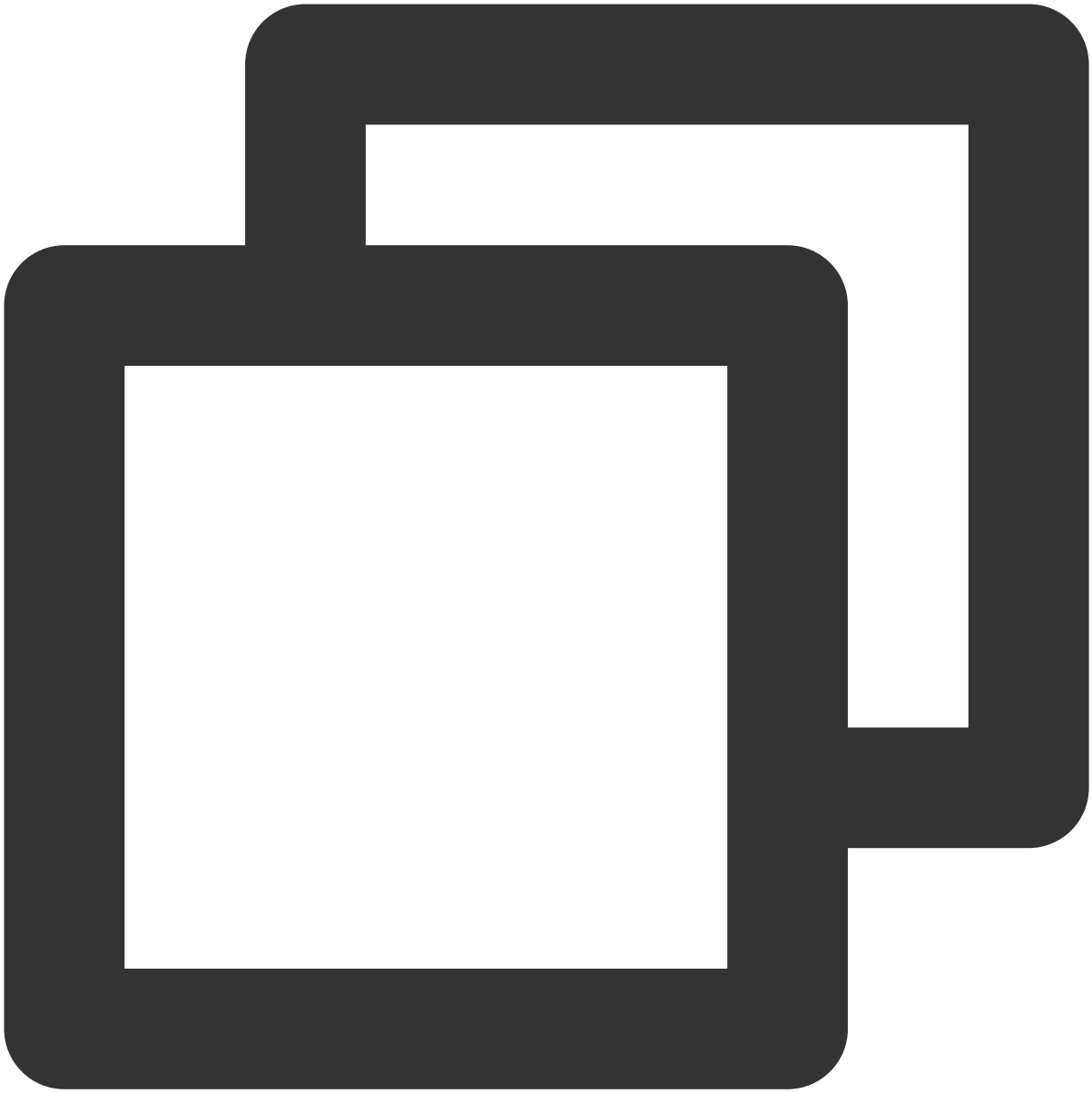
groupCall

IMグループの通話に招待します。被招待者は `EVENT.INVITED` イベントを受信します。

注意

オフラインプッシュは端末（AndroidまたはiOS)にのみ適用され、WebおよびWeChat Mini Programではサポートされません。

©2013-2022 Tencent Cloud. All rights reserved.



```
let promise = tuiCallEngine.groupCall({
    userIDList: ['user1', 'user2'],
    type: 1,
    groupID: 'IMグループID',
});
promise.then(() => {
    //success
}).catch(error => {
    console.warn('groupCall error:', error);
});
```

©2013-2022 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	Array	招待リスト
type	Number	0-不明、1-音声通話、2-ビデオ通話
groupID	String	IMグループ
timeout	String	タイムアウト時間（オプション）
roomID	String	ルームID（オプション）
offlinePushInfo	Object	カスタムオフラインメッセージプッシュ（オプション、tsignalingバージョ

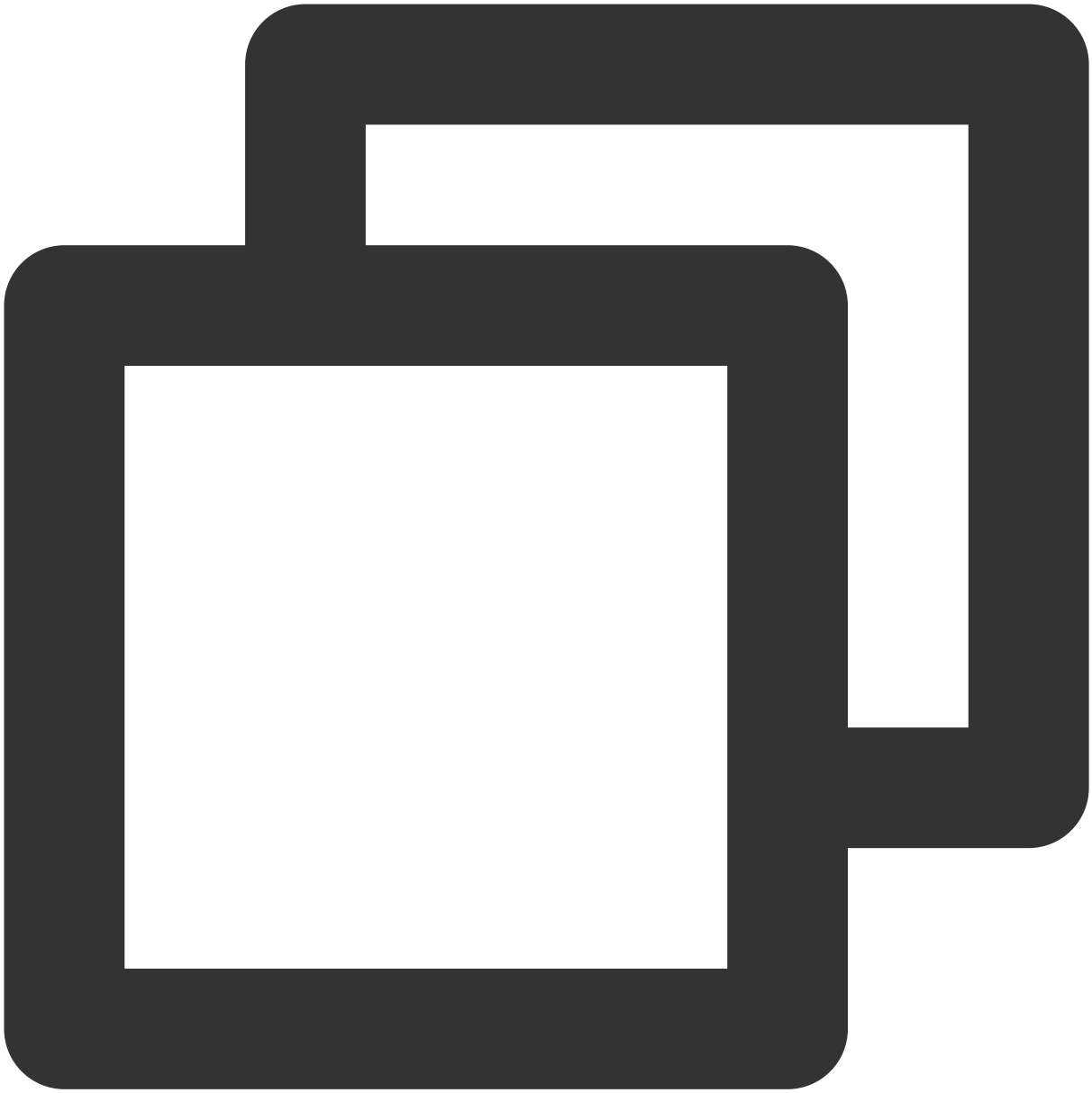
offlinePushInfo

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内容（オプション）
androidOPPOChannelID	string	オフラインプッシュでOPPO携帯電話8.0システム以上のチャンネ
extension	string	オフラインプッシュパススルー内容（オプション）（tsignalingバ

accept

被招待者として `TUICallEvent.INVITED` イベントのコールバックを受信した場合は、このインターフェースを呼び出して通話に応答することができます。

©2013-2022 Tencent Cloud. All rights reserved.

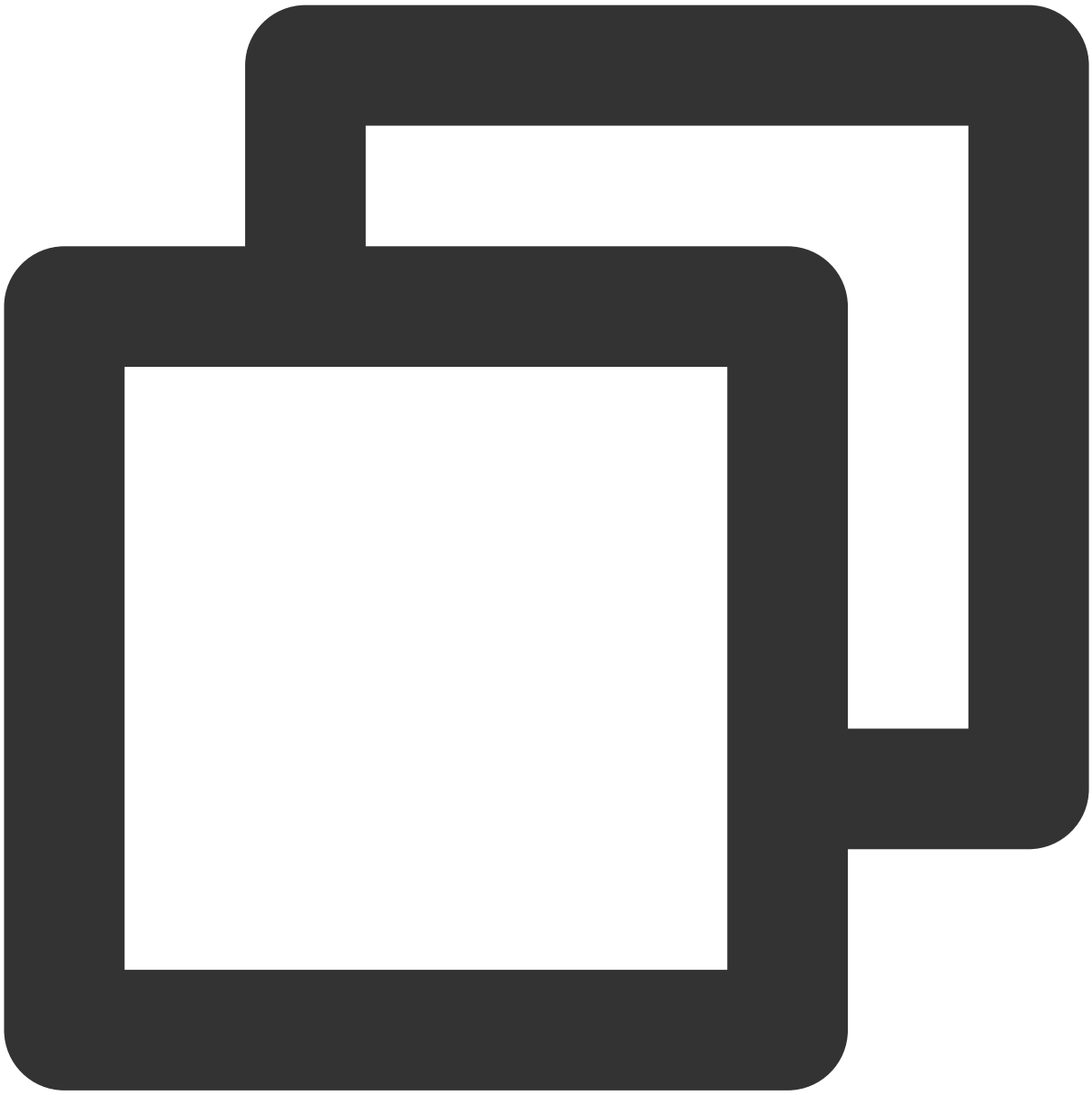


```
tuiCallEngine.on(TUICallEvent.INVITED, () => {
    tuiCallEngine.accept().promise.then(() => {
        //success
    }).catch(error => {
        console.warn('accept error:', error);
    });
});
```

reject

©2013-2022 Tencent Cloud. All rights reserved.

現在の通話を拒否します。着呼側として `TUICallEvent.INVITED` のコールバックを受信した場合は、この関数を呼び出して通話を拒否することができます。

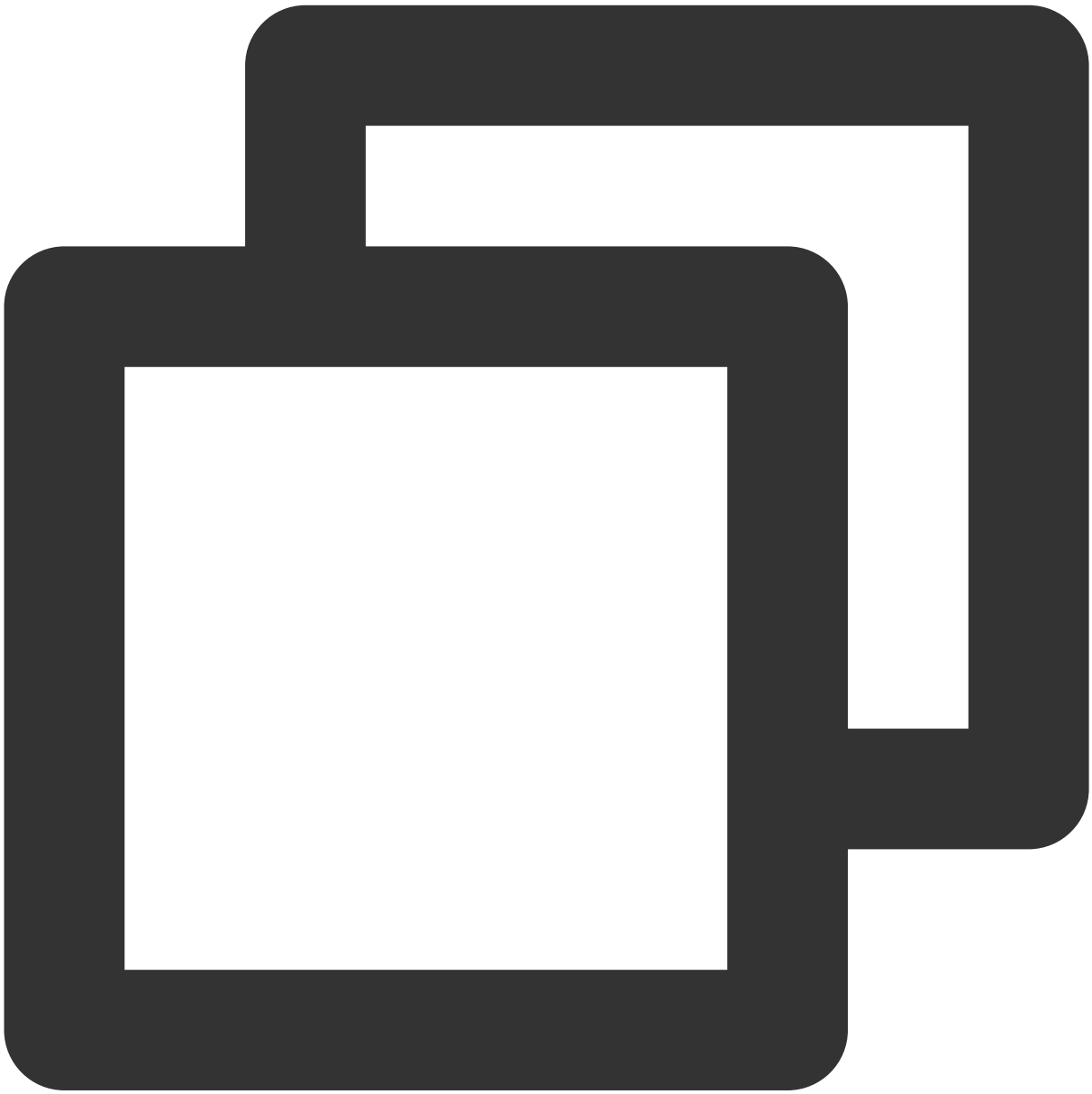


```
tuiCallEngine.on(TUICallEvent.INVITED, () => {
    tuiCallEngine.reject().then(() => {
        //success
    }).catch(error => {
        console.warn('reject error:', error);
    });
});
```

©2013-2022 Tencent Cloud. All rights reserved.

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。
通話中にこのインターフェースを呼び出せば、通話を終了することができます
ダイヤルしていない状態では、通話をキャンセルするために用いることができます

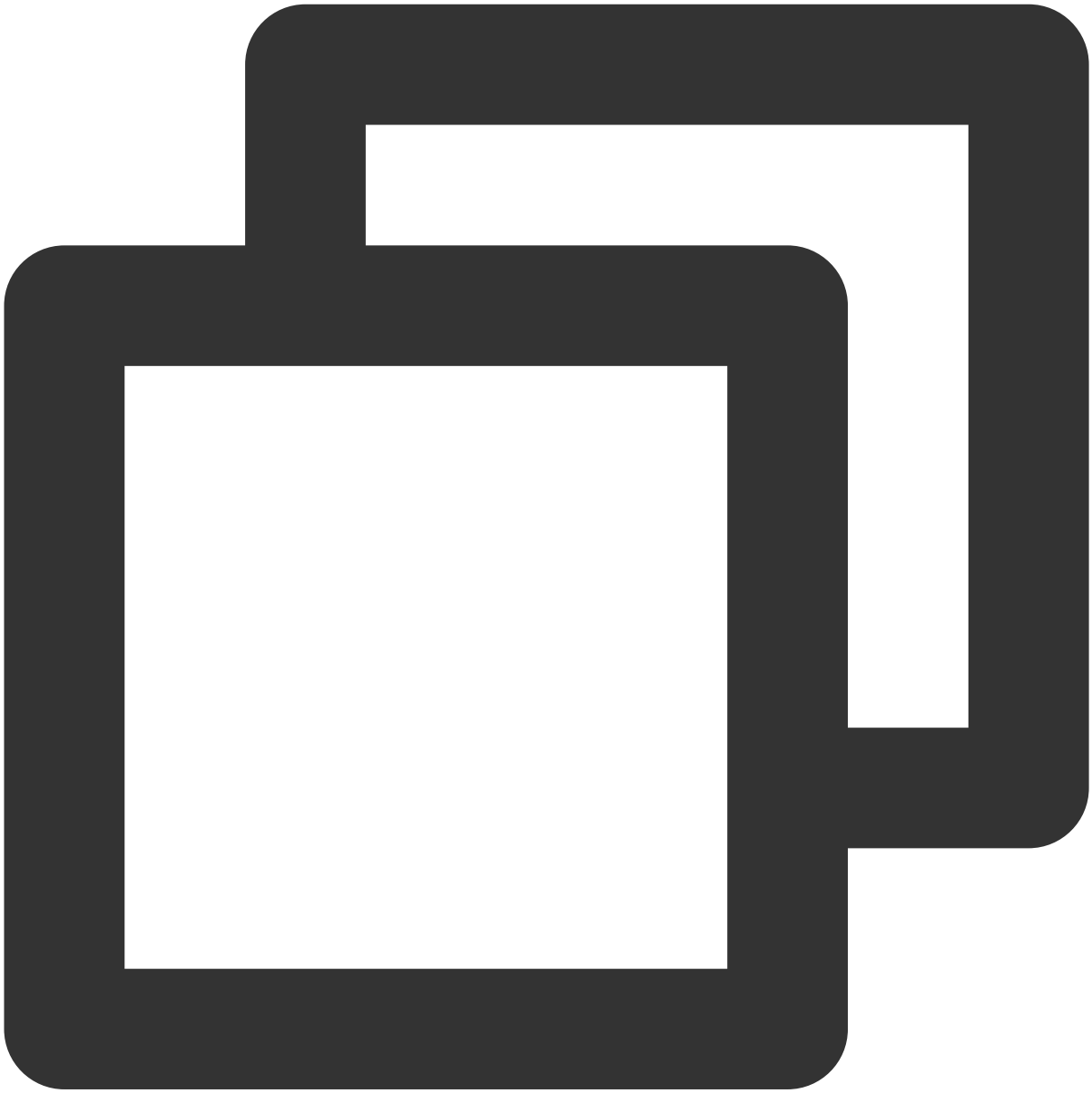


```
tuiCallEngine.hangup().then(() => {
    //success
 }).catch(error => {
    console.warn('hangup error:', error);
 });
```

©2013-2022 Tencent Cloud. All rights reserved.

inviteUser

今回のグループ通話にユーザーを招待します。
ユースケース：グループ通話中のユーザーが自主的に他の人を招待する場合に使用します。



```
const userIDList = ['jack', 'john'];
const params = {
    userIDList
};
tuiCallEngine.inviteUser(params).then(() => {
    // success
```

©2013-2022 Tencent Cloud. All rights reserved.

```
}).catch(error => {
    console.error('inviteUser error:', error);
});
```

パラ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	ターゲットユーザーのuserIDリスト
offlinePushInfo	Object	カスタムオフラインメッセージプッシュ（オプション、tsignalingバージョ

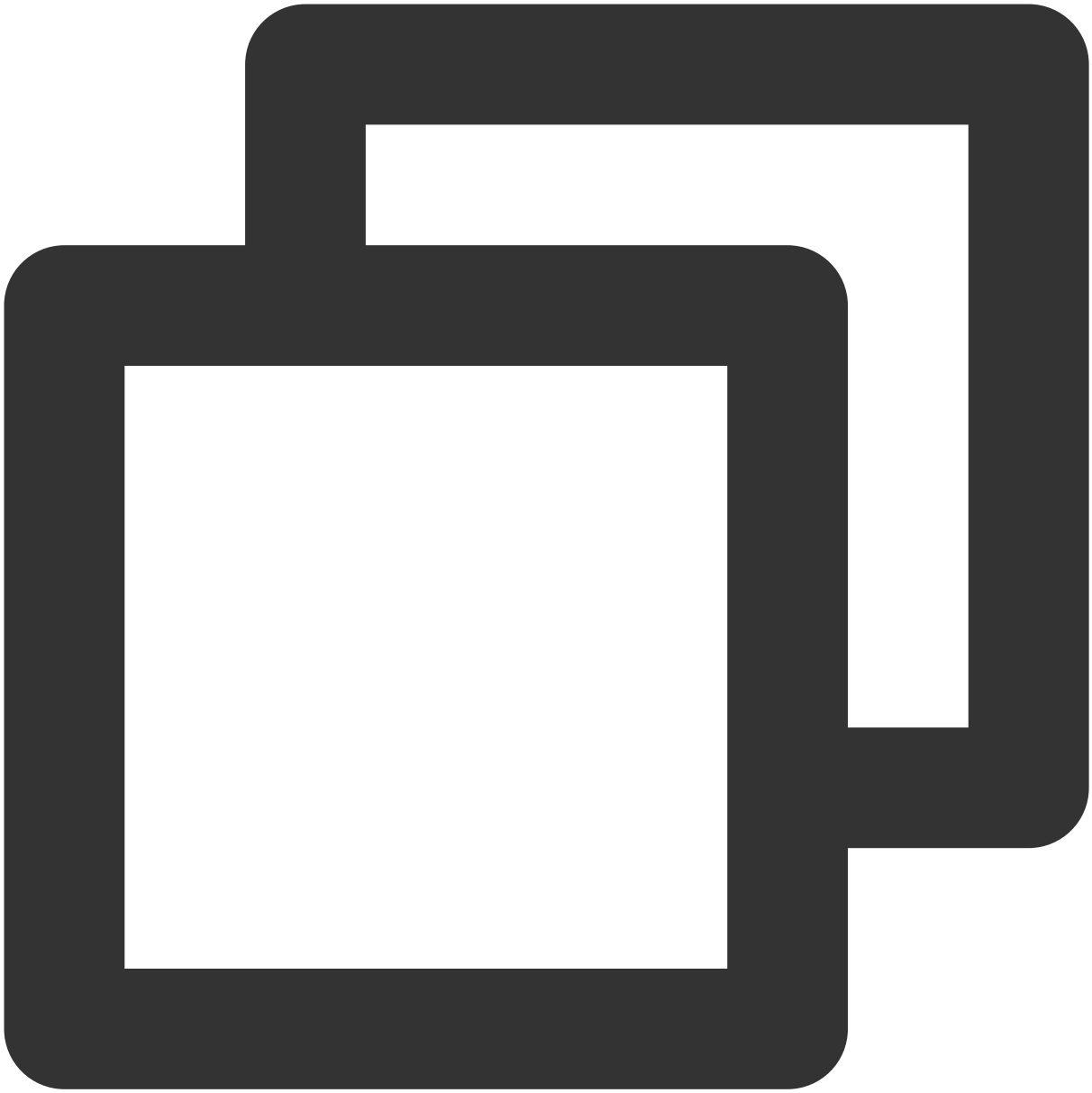
offlinePushInfo

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内容（オプション）
androidOPPOChannelID	string	オフラインプッシュでOPPO携帯電話8.0システム以上のチャンネ
extension	string	オフラインプッシュパススルー内容（オプション）（tsignalingバ

joinInGroupCall

今回のグループ通話に自主的に参加します。
ユースケース：グループ内のユーザーが今回のグループ通話に自主的に参加する場合に使用します。

©2013-2022 Tencent Cloud. All rights reserved.



```
const params = {
    roomID: 123,
    type: 1,
    groupID: 111
};
tuiCallEngine.joinInGroupCall(params).then(() => {
    // success
}).catch(error => {
    console.error('joinInGroupCall error:', error);
});
```

©2013-2022 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	Number	今回の通話のオーディオビデオルームID。現在は数字 ます。文字列のルームナンバーは今後のバージョンて
groupID	String	このグループ通話のグループID
type	Number	通話のメディアタイプ。例：1-音声通話、2-ビデオ通

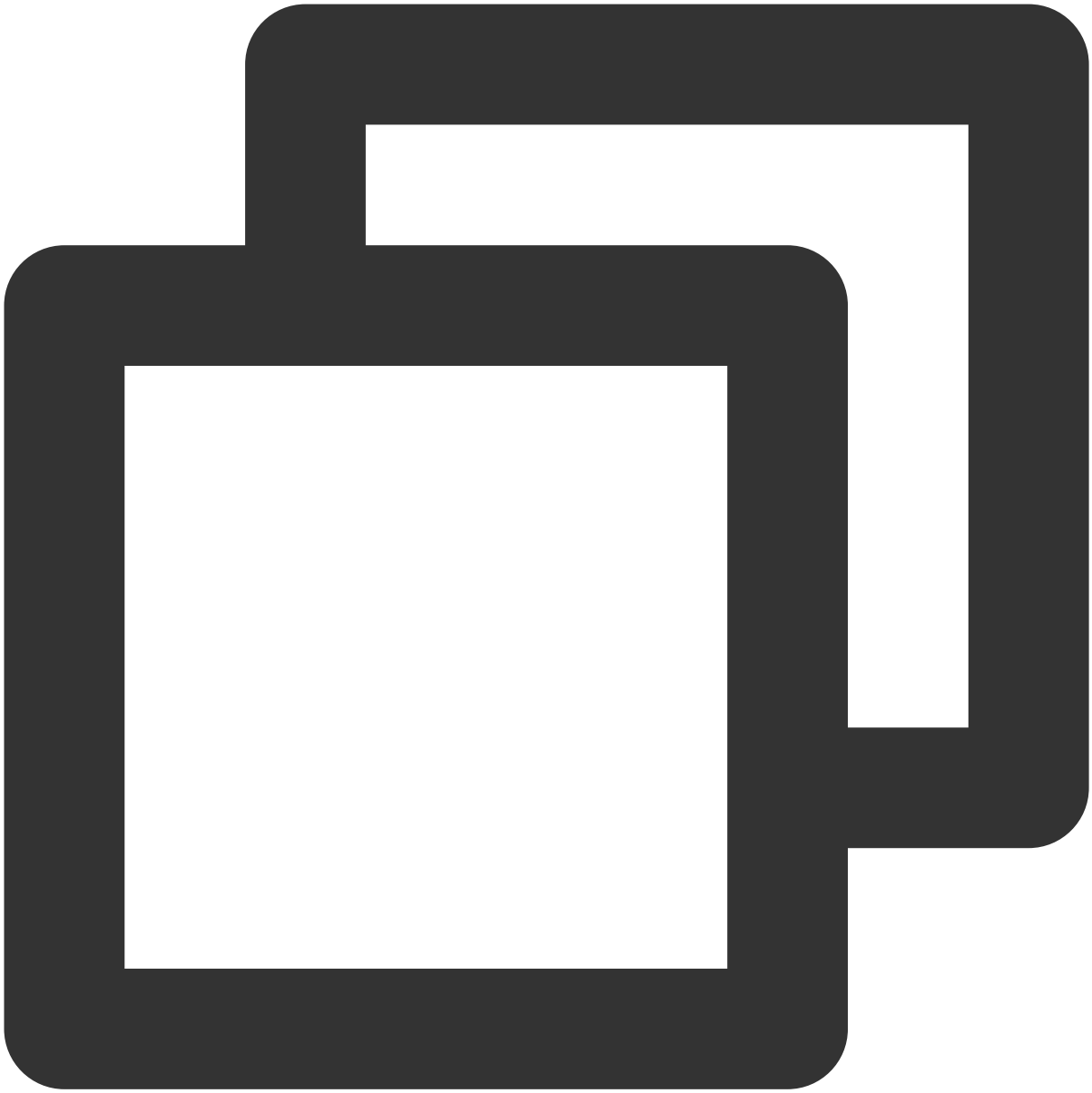
switchCallMediaType

現在の通話タイプを切り替えます。

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

©2013-2022 Tencent Cloud. All rights reserved.



```
// 1は音声通話、2はビデオ通話を意味します
tuiCallEngine.switchCallMediaType(2).then(() => {
  //success
}).catch(error => {
  console.warn('switchCallMediaType error:', error);
});
```

パラメータは下表に示すとおりです

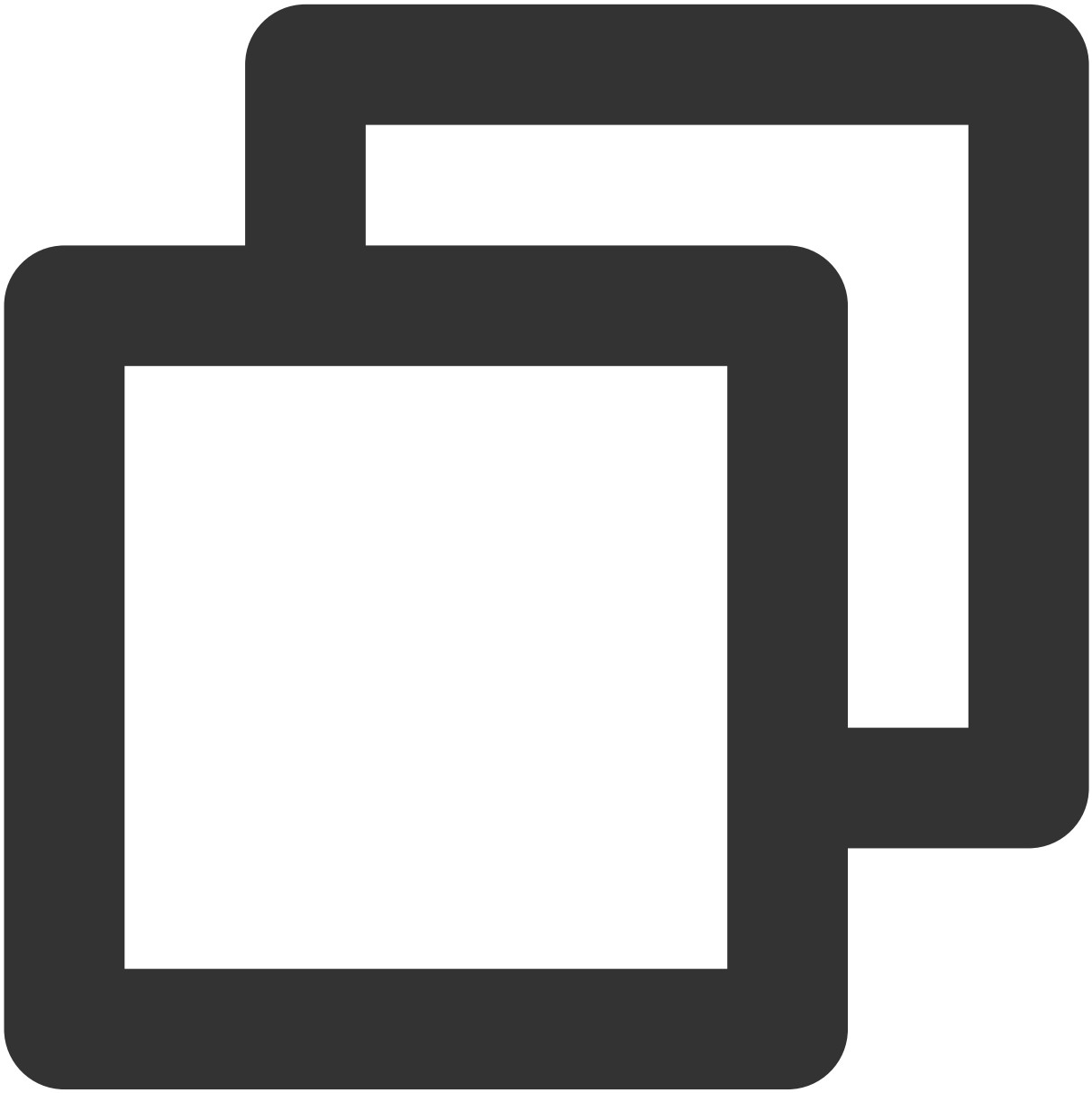
パラメータ	タイプ	意味

©2013-2022 Tencent Cloud. All rights reserved.

newMediaType	Number	1-音声通話、2-ビデオ通話

startRemoteView

リモート画面レンダリングを起動します。USER_VIDEO_AVAILABLEイベントを受信してから、対応するリモート画面を再レンダリングしてください。



```
let promise = tuiCallEngine.startRemoteView({
    userID: 'user1',
    videoViewDomID: 'video_1',
});
```

©2013-2022 Tencent Cloud. All rights reserved.

```
promise.then(() => {
    //success
}).catch(error => {
    console.warn('startRemoteView error:', error);
});
```

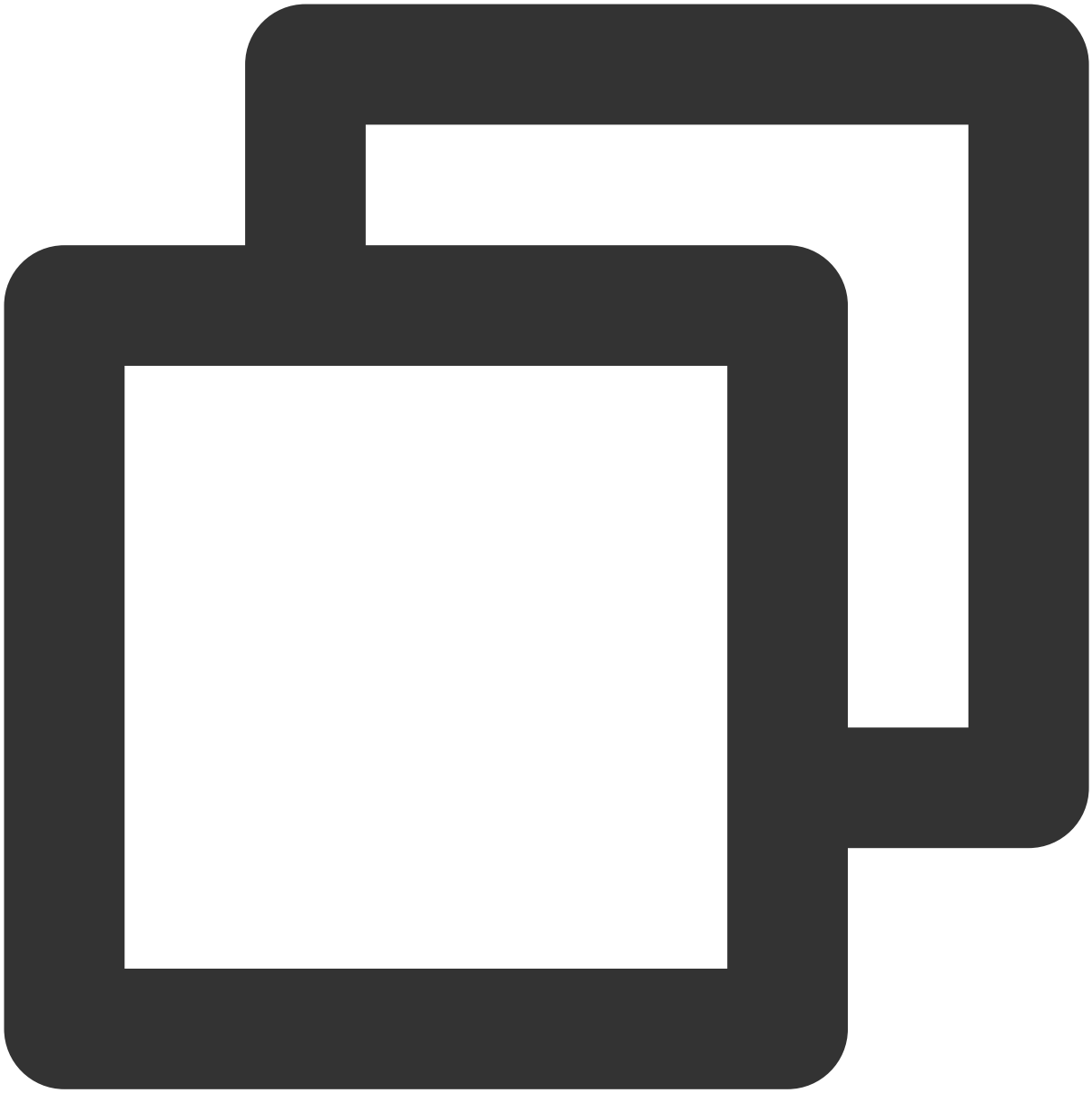
パラメータは下表に示すとおりです

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dom idノードにレンダ

stopRemoteView

リモート画面のレンダリングを停止します

©2013-2022 Tencent Cloud. All rights reserved.



```
tuiCallEngine.stopRemoteView({userID: 'user1'});
```

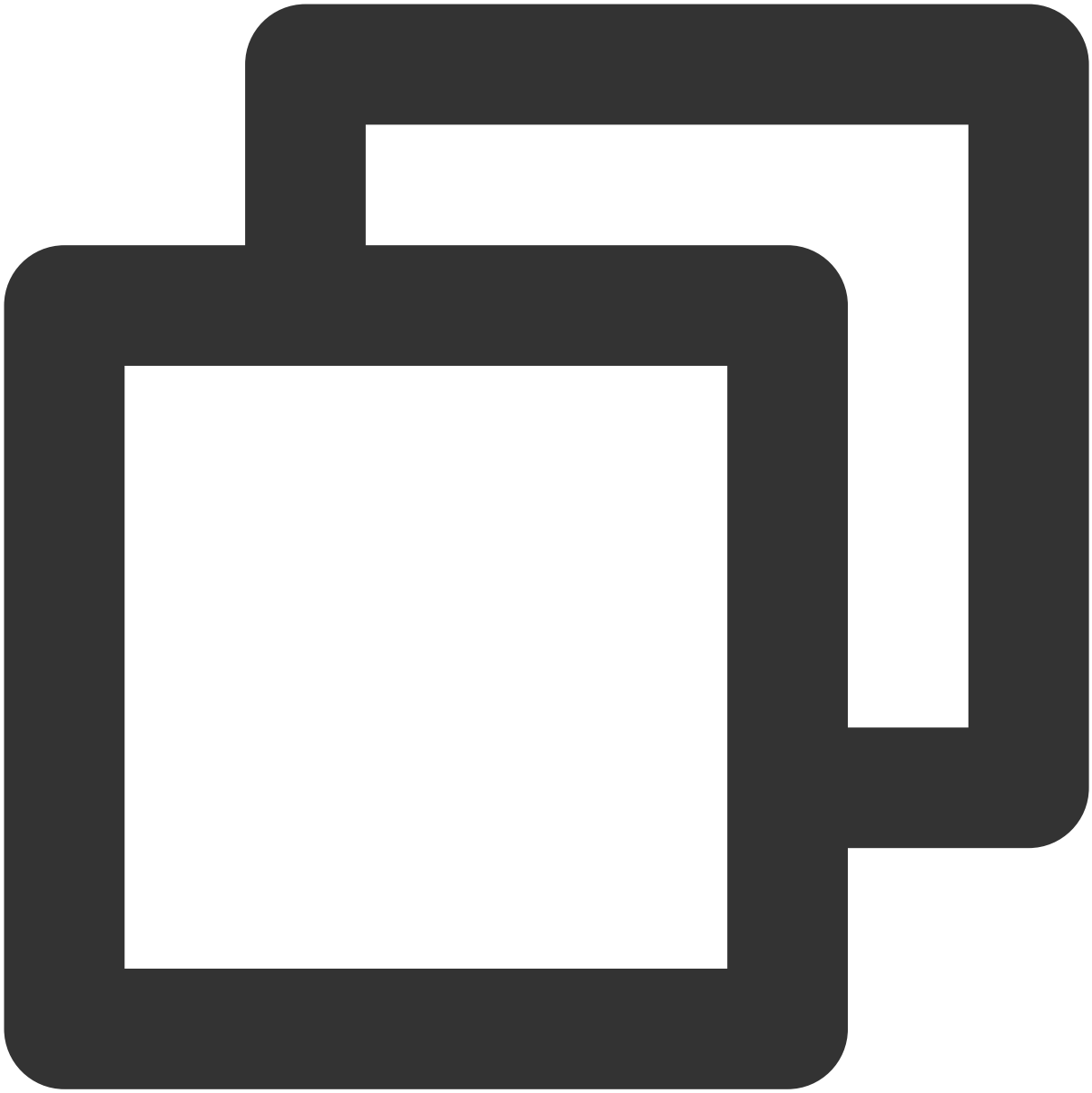
パラメータは下表に示すとおりです

パラメータ	タイプ	意味
userID	String	ユーザーi

startLocalView

ローカル画面のレンダリングを起動します

©2013-2022 Tencent Cloud. All rights reserved.



```
let promise = tuiCallEngine.startLocalView({
    userID: 'user1',
    videoViewDomID: 'video_1'
});
promise.then(() => {
    //success
}).catch(error => {
    console.warn('startLocalView error:', error);
});
```

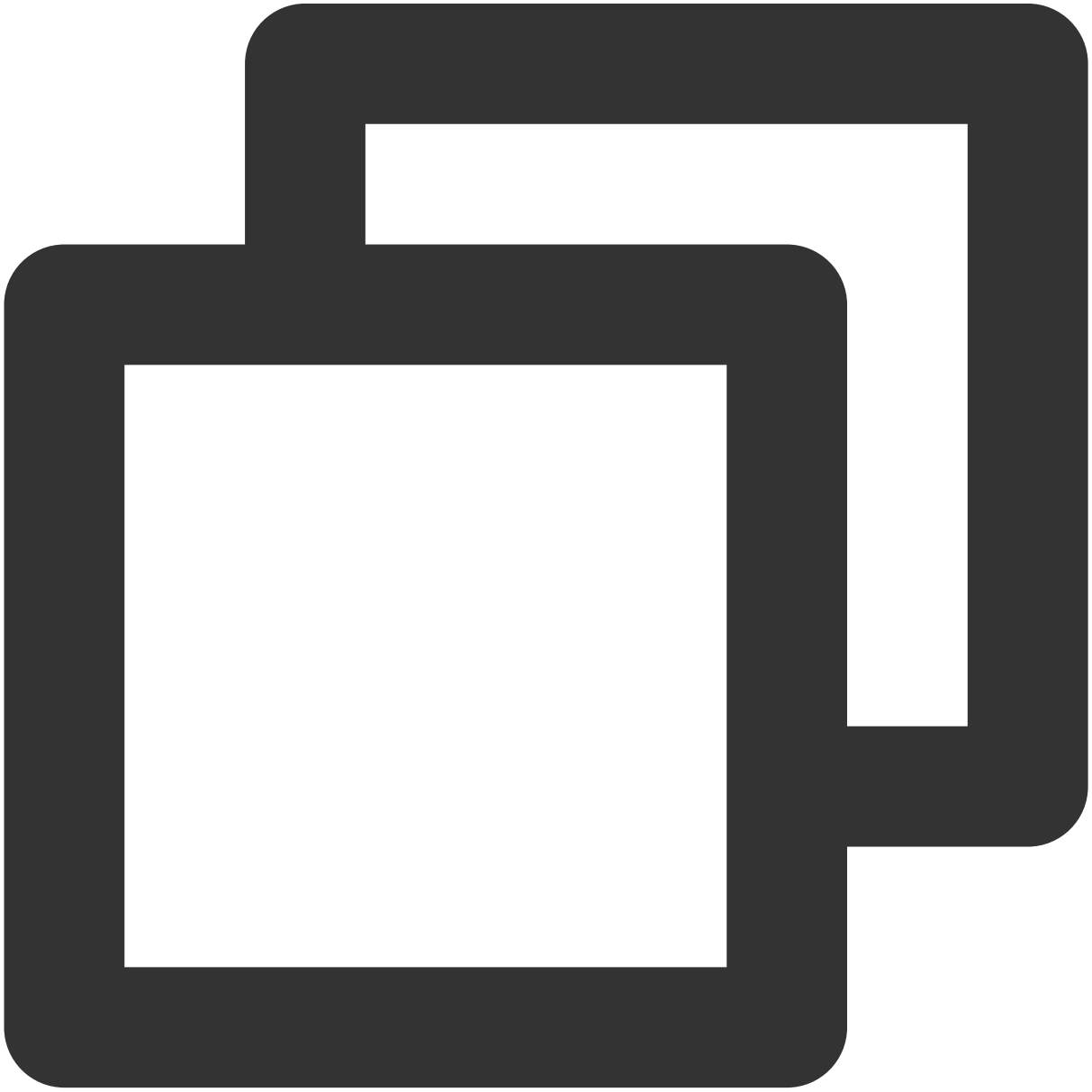
パラメータは下表に示すとおりです

©2013-2022 Tencent Cloud. All rights reserved.

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dom idノードにレン

stopLocalView

ローカル画面のレンダリングを停止します



```
let promise = tuiCallEngine.stopLocalView({userID: 'user1'});
```

©2013-2022 Tencent Cloud. All rights reserved.

```
promise.then(() => {
    //success
}).catch(error => {
    console.warn('stopLocalView error:', error)
});
```

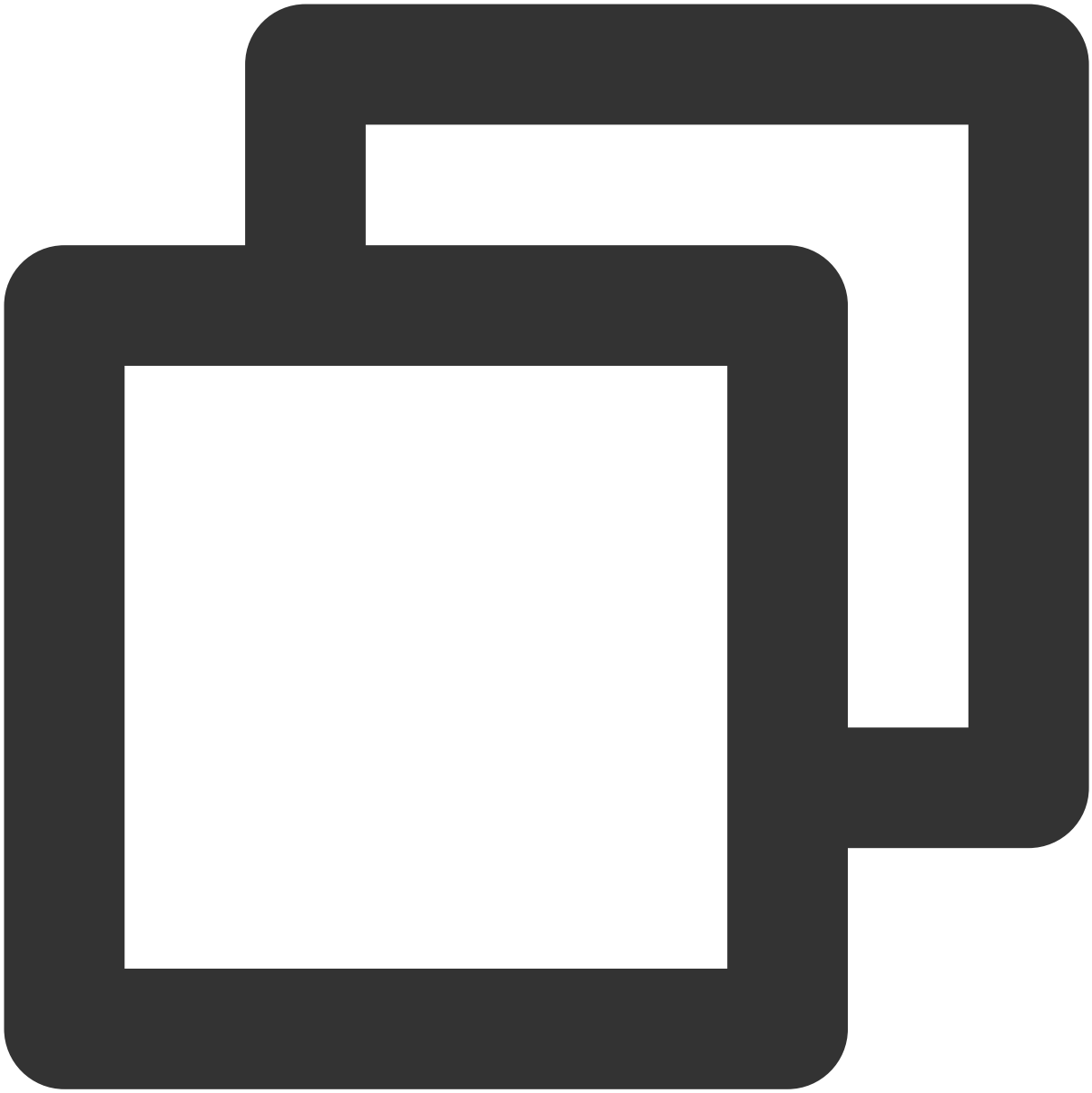
パラメータは下表に示すとおりです

パラメータ	タイプ	意味
userID	String	ユーザーID

openCamera

カメラをオンにします。

©2013-2022 Tencent Cloud. All rights reserved.

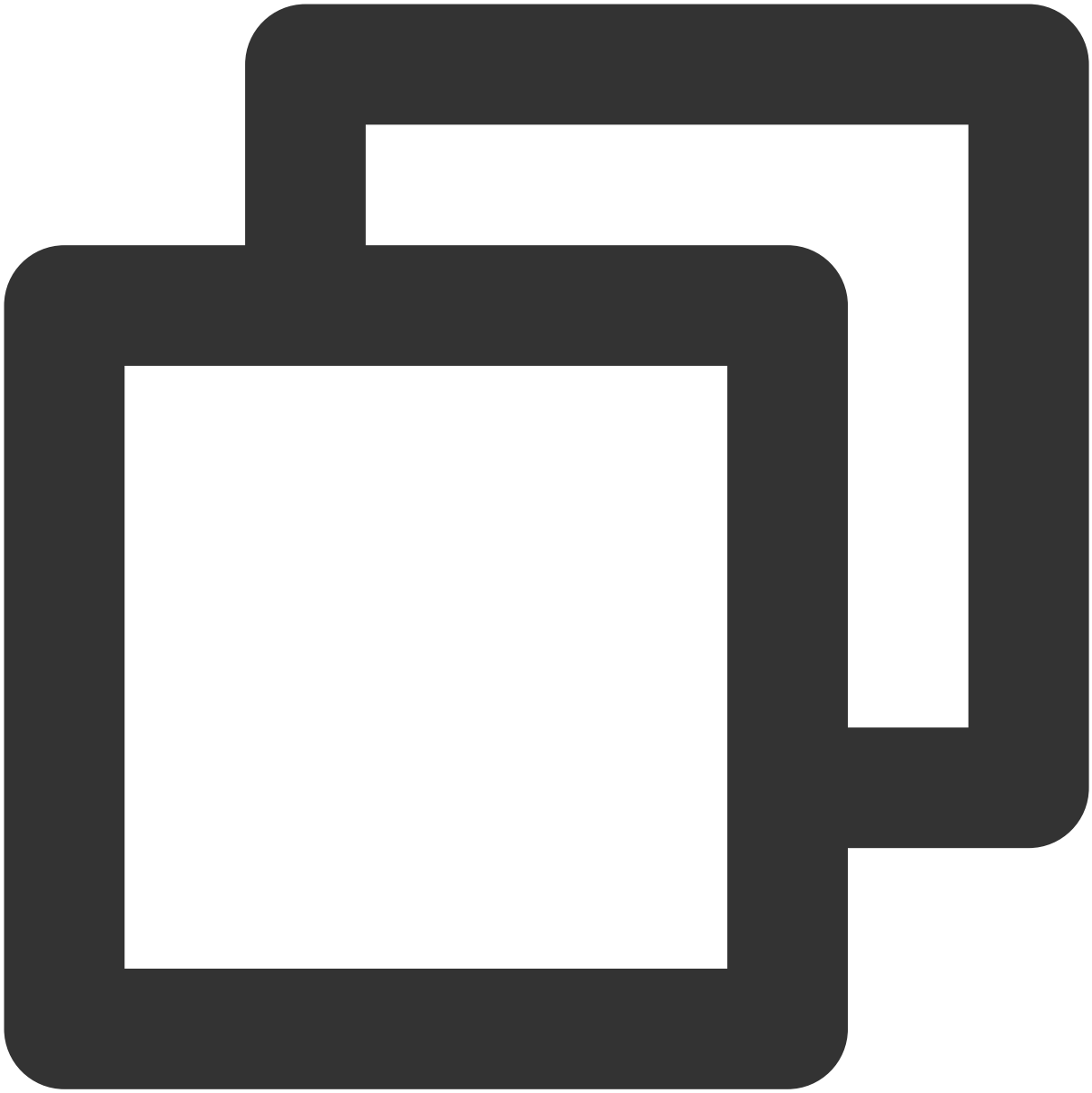


```
tuiCallEngine.openCamera().then(() => {
    //success
}).catch(error => {
    console.warn('openCamera error:', error);
});
```

closeCamara

カメラをオフにします

©2013-2022 Tencent Cloud. All rights reserved.

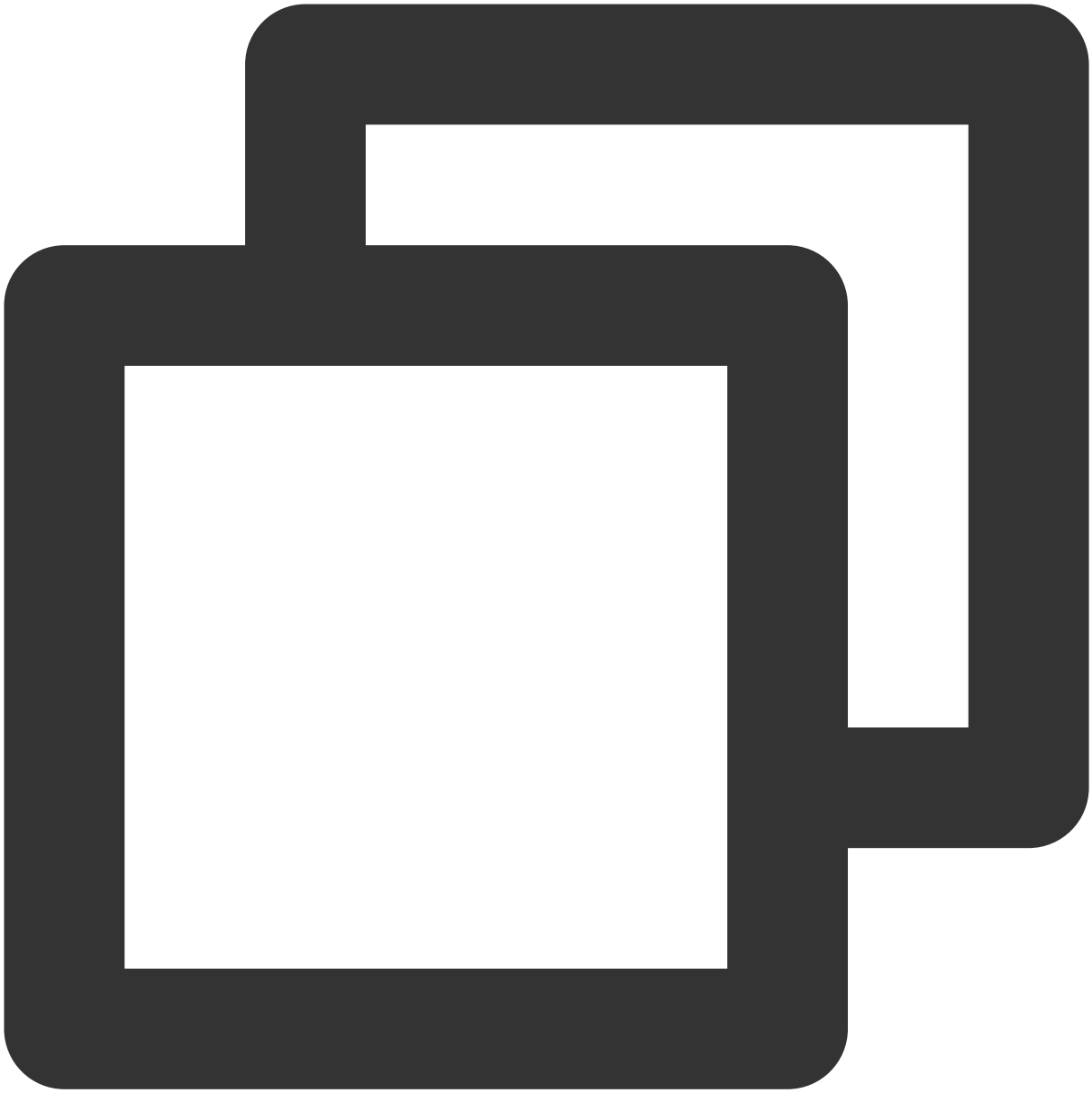


```
tuiCallEngine.closeCamera().then(() => {
    //success
}).catch(error => {
    console.warn('closeCamara error:', error);
});
```

openMicrophone

マイクをオンにします。

©2013-2022 Tencent Cloud. All rights reserved.

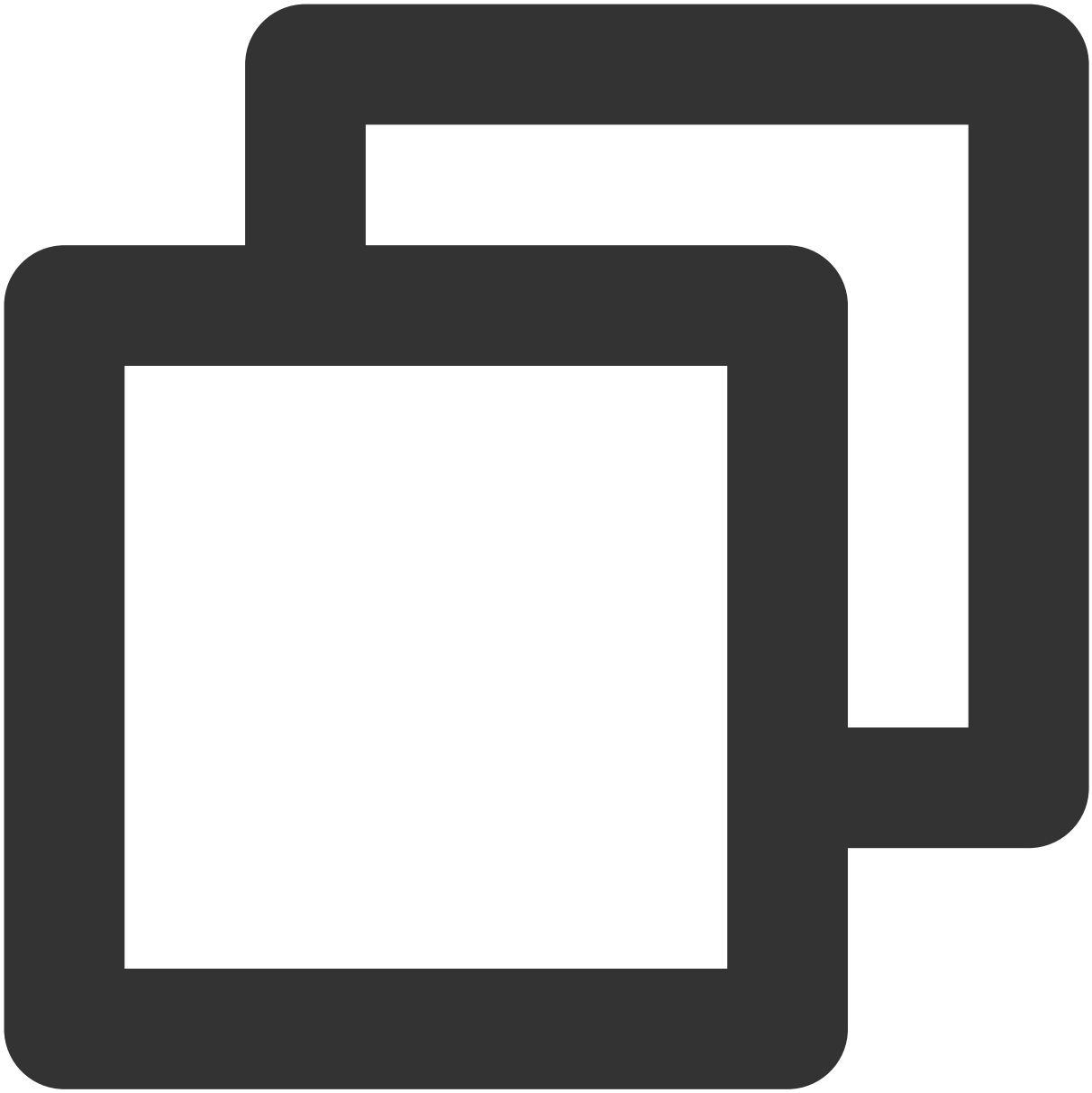


```
tuiCallEngine.openMicrophone().then(() => {
    //success
}).catch(error => {
    console.warn('openMicrophone error:', error);
});
```

closeMicrophone

マイクをオフにします。

©2013-2022 Tencent Cloud. All rights reserved.

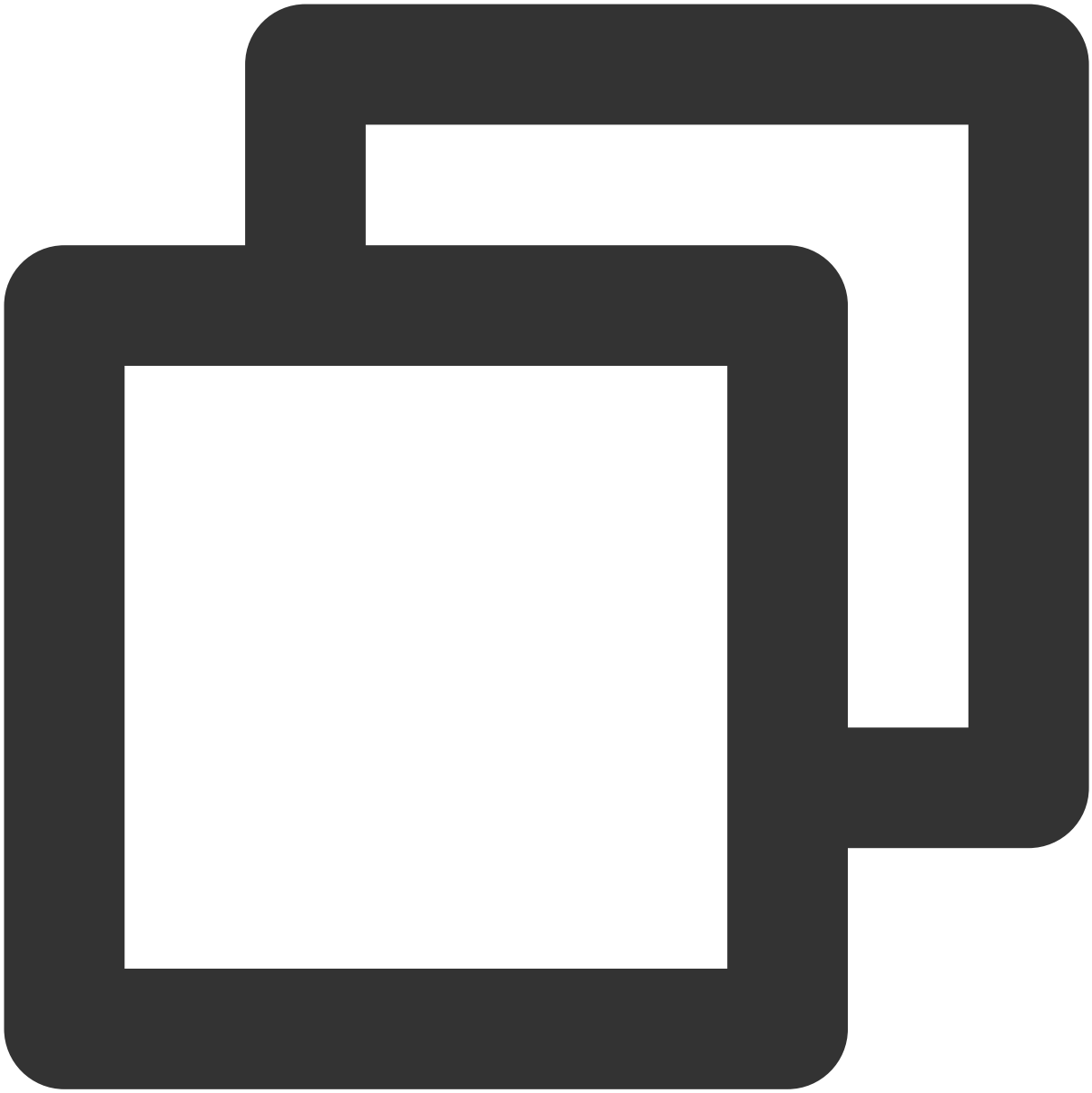


```
tuiCallEngine.closeMicrophone().then(() => {
    //success
}).catch(error => {
    console.warn('closeMicrophone error:', error);
});
```

setVideoQuality

ビデオ画質を設定します。

©2013-2022 Tencent Cloud. All rights reserved.



```
const profile = '720p';
tuiCallEngine.setVideoQuality(profile).then(() => {
    //success
}).catch(error => {
    console.warn('setVideoQuality error:', error)
});    // ビデオ画質を720pに設定します
```

パラメータは下表に示すとおりです

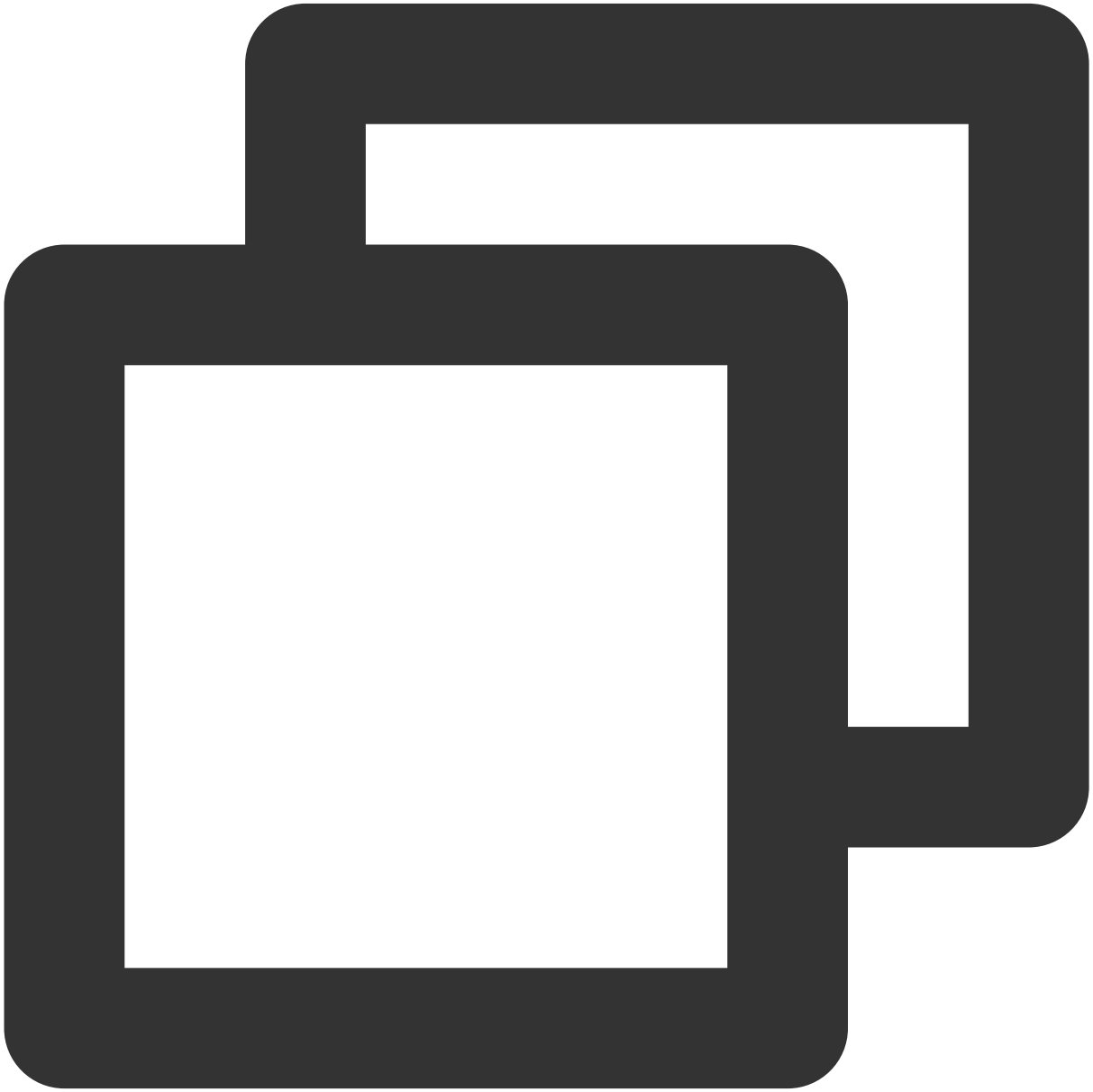
ビデオProfile	解像度（幅 x 高さ）

©2013-2022 Tencent Cloud. All rights reserved.

480p	640 × 480
720p	1280 × 720
1080p	1920 × 1080

getDeviceList

デバイスリストを取得します。



```
tuiCallEngine.getDeviceList("camera").then((devices) => {
```

©2013-2022 Tencent Cloud. All rights reserved.

```
    console.log(devices);
}).catch(error => {
    console.warn('getDeviceList error:', error);
});
```

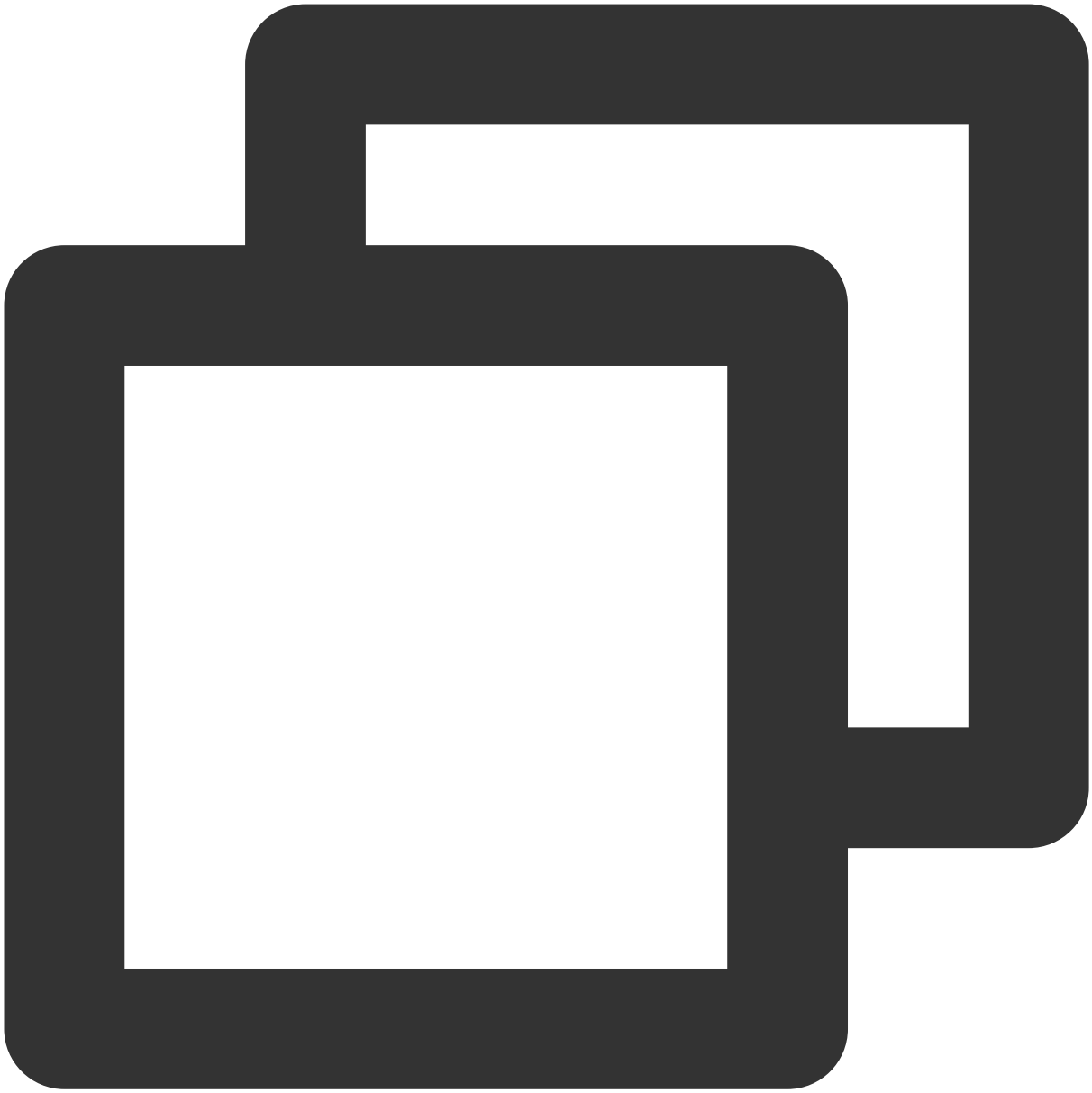
パラメータは下表に示すとおりです

パラメータ	タイプ	意味
deviceType	String	'camera'-カメラ、'microphones'-マイク

switchDevice

カメラまたはマイクデバイスを切り替えます。

©2013-2022 Tencent Cloud. All rights reserved.



```
let promsie = tuiCallEngine.switchDevice({
    deviceType: 'video',
    deviceId: cameras[0].deviceId
});
promise.then(() => {
    //success
}).catch(error => {
    console.warn('switchDevice error:', error)
});
```

パラメータは下表に示すとおりです

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
deviceType	String	切り替えが必要なデバイスタイプ 'video'カメラ 'audio'マイク
deviceId	String	切り替えが必要なデバイスID カメラデバイスの識別子は、getCameras()で取得します マイクデバイスの識別子は、getMicrophones()で取得します

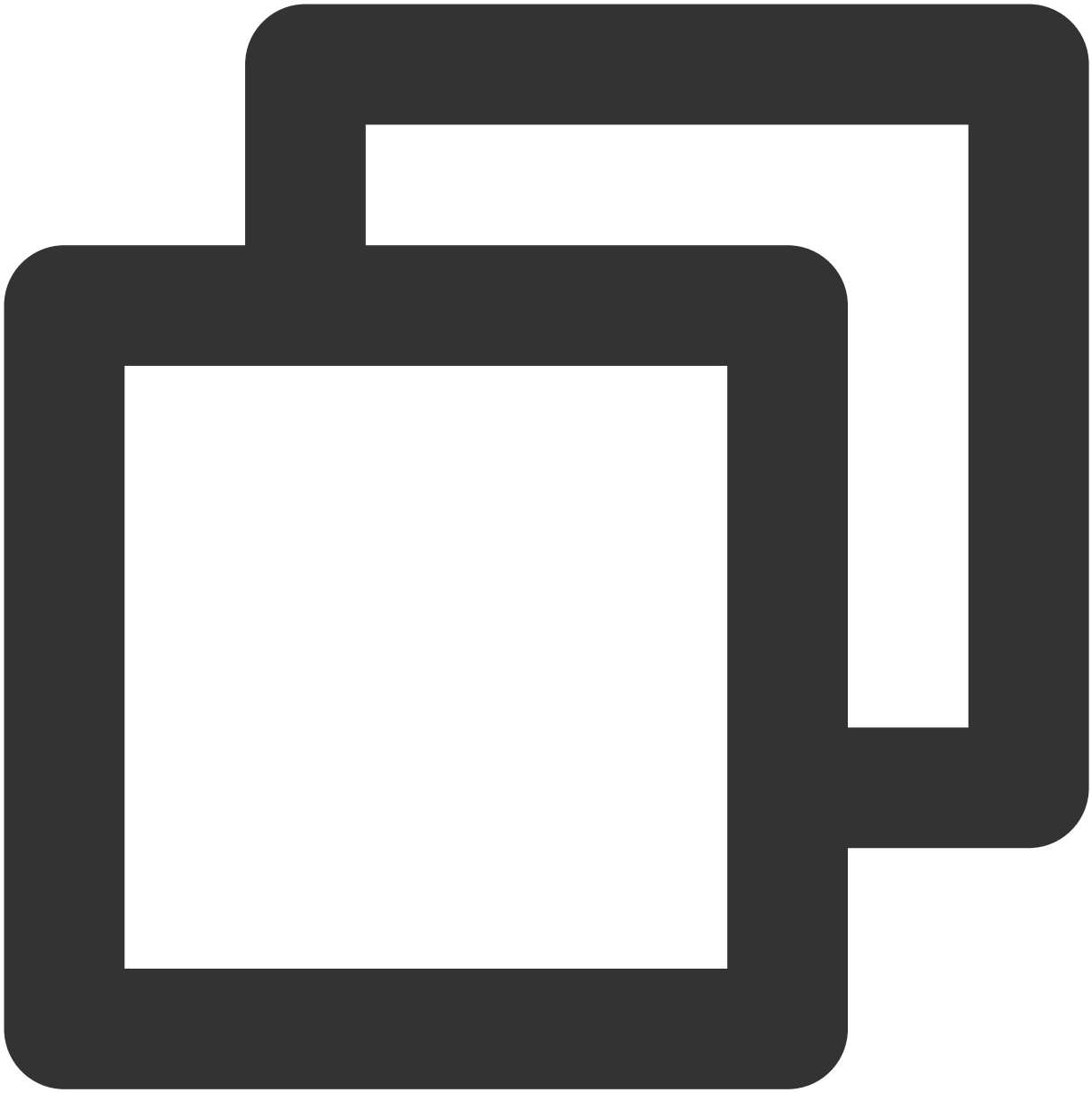
enableAIVoice

AIノイズリダクションのオン/オフ。

注意

バージョン4.12.1以降でサポートしています。使用方法の詳細については、AIノイズリダクションの使用をご参照ください。

©2013-2022 Tencent Cloud. All rights reserved.



```
let promsie = tuiCallEngine.enableAIVoice(true);
promise.then(() => {
    // success
}).catch(error => {
    console.warn('enableAIVoice error:', error)
});
```

パラメータは下表に示すとおりです

パラメータ	タイプ	意味

©2013-2022 Tencent Cloud. All rights reserved.

enable	Boolean	AIノイズリダクションのオン/オフ。enable=trueはオンを示し、デフォルションはオンになっています

©2013-2022 Tencent Cloud. All rights reserved.

TUICallEvent

最終更新日：：2022-11-21 16:34:37

TUICallEvent APIの概要

TUICallEvent APIはオーディオビデオ通話コンポーネントの**イベントインターフェース**です。

イベントリスト

EVENT	説明
TUICallEvent.ERROR	SDKの内部でエラーが発生しました
TUICallEvent.SDK_READY	SDKがready状態に入ったときにこのコールバックを受信します
TUICallEvent.KICKED_OUT	重複ログインです。このコールバックを受信した場合は、ルームからの強制退出を意味します
TUICallEvent.USER_ACCEPT	応答したユーザーがいる場合に、このコールバックを受信します
TUICallEvent.USER_ENTER	通話への参加に同意したユーザーがいる場合に、このコールバックを受信します
TUICallEvent.USER_LEAVE	通話からの退出に同意したユーザーがいる場合に、このコールバックを受信します
TUICallEvent.REJECT	ユーザーが通話を拒否
TUICallEvent.NO_RESP	招待したユーザーからの応答なし
TUICallEvent.LINE_BUSY	招待者が通話中
TUICallEvent.CALLING_TIMEOUT	被招待者が受信します。このコールバックを受信した場合は、今回の通話に応答せずタイムアウトしたことを意味します
TUICallEvent.USER_VIDEO_AVAILABLE	リモートユーザーによるカメラのオン/オフがあった場合に、このコールバックを受信します

©2013-2022 Tencent Cloud. All rights reserved.

EVENT	説明
TUICallEvent.USER_AUDIO_AVAILABLE	リモートユーザーによるマイクのオン/オフがあった場合に、このコールバックを受信します
TUICallEvent.USER_VOICE_VOLUME	リモートユーザーがスピーカーの音量調整を行った場合に、このコールバックを受信します
TUICallEvent.GROUP_CALL_INVITEE_LIST_UPDATE	グループチャットの招待リストが更新された場合にこのコールバックを受信します
TUICallEvent.INVITED	通話に招待されました
TUICallEvent.CALLING_CANCEL	被招待者が受信します。このコールバックを受信した場合は、今回の通話がキャンセルされたことを意味します
TUICallEvent.CALLING_END	このコールバックを受信した場合は、今回の通話が終了したことを意味します
TUICallEvent.DEVICED_UPDATED	デバイスリストが更新された場合にこのコールバックを受信します
TUICallEvent.CALL_TYPE_CHANGED	通話タイプが切り替わった場合にこのコールバックを受信します

ERROR

SDK内部にエラーが発生しました。

```
let onError = function(error) {
console.log(error)
};
tuiCallEngine.on(TUICallEvent.ERROR, onError);
```

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

```
let onSDKReady = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.SDK_READY, onSDKReady);
```

KICKED_OUT

©2013-2022 Tencent Cloud. All rights reserved.

重複ログインです。このコールバックを受信した場合は、ルームからの強制退出を意味します。

```
let handleOnKickedOut = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.KICKED_OUT, handleOnKickedOut);
```

USER_ACCEPT

応答したユーザーがいる場合に、このコールバックを受信します。

```
let handleUserAccept = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.USER_ACCEPT, handleUserAccept);
```

USER_ENTER

通話への参加に同意したユーザーがいる場合に、このコールバックを受信します。

```
let handleUserEnter = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.USER_ENTER, handleUserEnter);
```

USER_LEAVE

通話からの退出に同意したユーザーがいる場合に、このコールバックを受信します。

```
let handleUserLeave = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.USER_LEAVE, handleUserLeave);
```

REJECT

ユーザーが通話を拒否しました。

```
let handleInviteeReject = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.REJECT, handleInviteeReject);
```

©2013-2022 Tencent Cloud. All rights reserved.

NO_RESP

招待されたユーザーは応答しませんでした。

C2C通話では、発信者のみが応答なしのコールバックを受信します。例えばA がB、C を通話に招待し、Bが応答しなかった場合、Aはこのコールバックを受信できますが、Cは受信できません。

IMグループ通話では、すべての被招待者がこのコールバックを受信できます。例えばA がB、C を通話に招待し、Bが応答しなかった場合、A、Cのどちらもこのコールバックを受信できます。

```
let handleNoResponse = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.NO_RESP, handleNoResponse);
```

LINE_BUSY

招待者が通話中です。

```
let handleLineBusy = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.LINE_BUSY, handleLineBusy);
```

CALLING_TIMEOUT

被招待者が受信します。このコールバックを受信した場合は、今回の通話に応答せずタイムアウトしたことを意味します。

```
let handleCallingTimeout = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.CALLING_TIMEOUT, handleCallingTimeout);
```

USER_VIDEO_AVAILABLE

リモートユーザーによるカメラのオン/オフがあった場合に、このコールバックを受信します。

```
let handleUserVideoChange = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.USER_VIDEO_AVAILABLE, handleUserVideoChange);
```

USER_AUDIO_AVAILABLE

©2013-2022 Tencent Cloud. All rights reserved.

リモートユーザーによるマイクのオン/オフがあった場合に、このコールバックを受信します。

```
let handleUserAudioChange = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.USER_AUDIO_AVAILABLE, handleUserAudioChange);
```

USER_VOICE_VOLUME

リモートユーザーがスピーカーの音量調整を行った場合に、このコールバックを受信します。

```
let handleUserVoiceVolumeChange = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.USER_VOICE_VOLUME, handleUserVoiceVolumeChange);
```

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

```
let handleGroupInviteeListUpdate = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.GROUP_CALL_INVITEE_LIST_UPDATE, handleGroupInviteeL
istUpdate);
```

INVITED

通話に招待されました。

```
let handleNewInvitationReceived = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.INVITED, handleNewInvitationReceived);
```

CALLING_CANCEL

被招待者が受信します。このコールバックを受信した場合は、今回の通話がキャンセルされたことを意味します。

```
let handleCallingCancel = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.CALLING_CANCEL, handleCallingCancel);
```

©2013-2022 Tencent Cloud. All rights reserved.

CALLING_END

このコールバックを受信した場合は、今回の通話が終了したことを意味します。

```
let handleCallingEnd = function(event) {
console.log(event)
};
tuiCallEngine.on(TUICallEvent.CALLING_END, handleCallingEnd);
```

DEVICED_UPDATED

デバイスリストが更新された場合にこのコールバックを受信します。

```
let handleDeviceUpdated = function({ microphoneList, cameraList, currentMicrophon
eID, currentCameraID }) {
console.log(microphoneList, cameraList, currentMicrophoneID, currentCameraID)
};
tuiCallEngine.on(TUICallEvent.DEVICED_UPDATED, handleDeviceUpdated);
```

CALL_TYPE_CHANGED

通話タイプが切り替わった場合にこのコールバックを受信します。

```
let handleCallTypeChanged = function({ oldCallType, newCallType }) {
console.log(oldCallType, newCallType)
};
tuiCallEngine.on(TUICallEvent.CALL_TYPE_CHANGED, handleDeviceUpdated);
```

©2013-2022 Tencent Cloud. All rights reserved.

アップグレード方法

Web

最終更新日：：2023-02-17 16:36:15

Webアップグレード方法

TUICallKitはTencent Cloudがリリースした新しいオーディオビデオ通話のUI コンポーネントで、TRTCCallingのアップグレードバージョンです。グループ通話、AIノイズリダクションなどのより多くの機能特性がサポートされています。また、全プラットフォーム間で相互呼び出しがサポートされているため、機能がより安定しています。新しいTUICallKitコンポーネントの使用を歓迎します。アップグレード前には、以下をご理解ください。
TRTCCallingおよびTUICallKitは相互呼び出しをサポートしています。相互通信に影響するため、アップグレードの前後でSDKAppIDを変更しないでください。
TUICallKitはIMオーディオビデオ通話機能パッケージとマッチングする必要があります。まず、IMコンソールをクリックし、対応するSDKAppIDアプリケーションの基本設定ページに進みます。次に、ページ右下隅でTRTCサービスをアクティブ化する機能エリアを見つけ、無料体験をクリックすると、TUICallKitの60日間無料トライアルサービスをアクティブ化することができます。



アップグレード概要

©2013-2022 Tencent Cloud. All rights reserved.

TUICallKitは、設計の開始時にTRTCCallingクライアントのアップグレードニーズが考慮されているため、20分程度のいくつかの簡単な手順だけでアップグレードが完了します。
2種類のアップグレード方式から選択することができます。
1. TUICallKit Vue3 UI オープンソースコンポーネントにアップグレードすると、数行のコードを記述するだけでAppにオーディオビデオ通話機能を追加することができ、オフラインの呼び出し機能がサポートされます。（推奨）
2. TUICallEngine SDKにアップグレードすると、直接旧バージョンのTRTCCallingから移行でき、UI層は開発済のバージョンを応用できます。
続いて2種類の方法のアップグレードステップを、それぞれご紹介します。

TUICallKit Vue3 UIコンポーネントのアップグレードステップ

TUICallKit Vue3 UIは、UI付きのオープンソースコンポーネントを提供します。開発時間の90%を節約できるため、オーディオビデオ通話アプリケーションをスピーディーにリリースできます。
通話効果を体験してデバッグしたい場合は、 Demoクイックスタートをご参照ください。
機能をプロジェクト内に直接埋め込みたい場合は、TUICallKitへのクイックアクセスをご参照ください。
UIインターフェースを修正したい場合は、TUICallKitインターフェースのカスタマイズガイドをご参照ください。

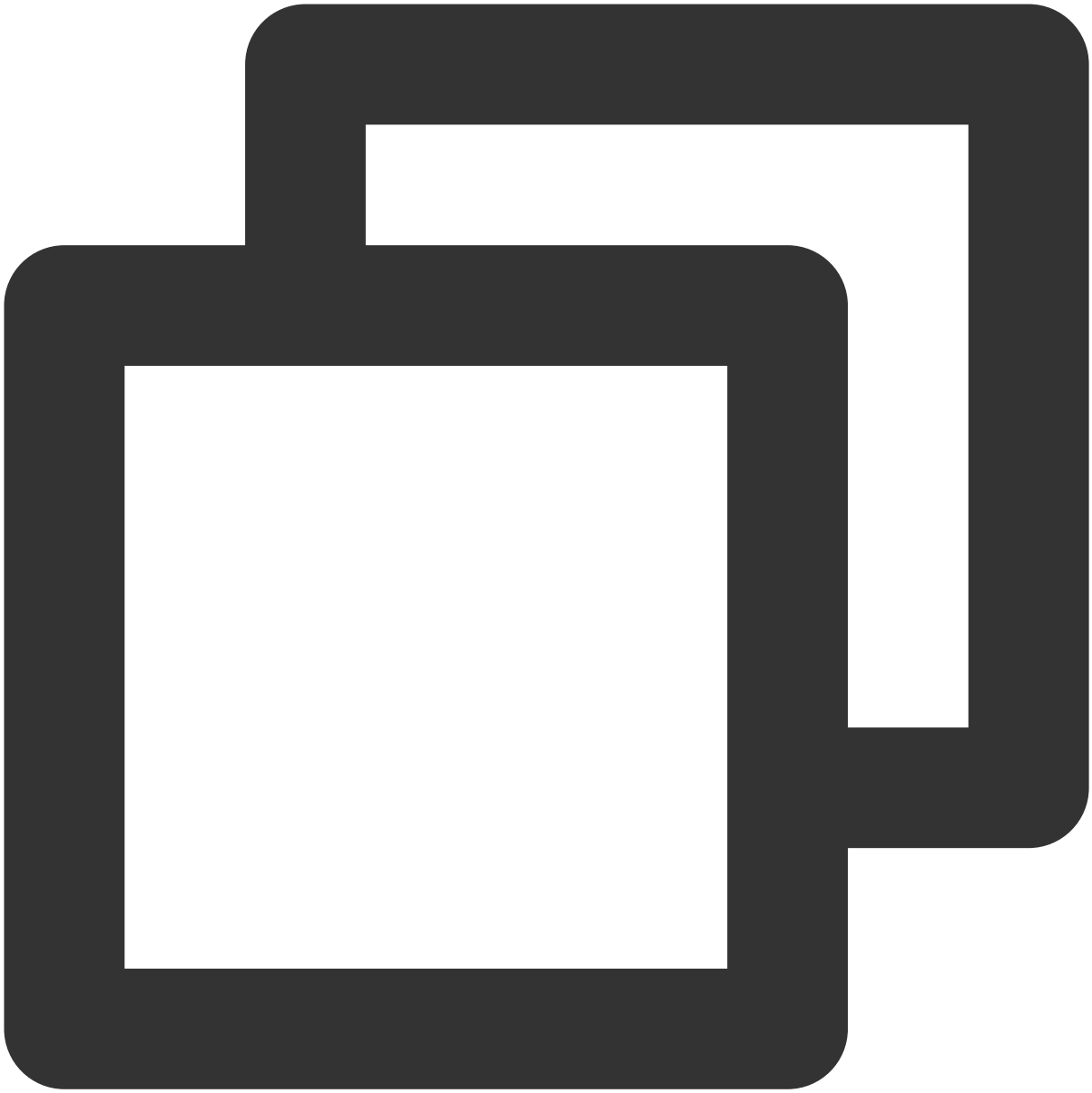
TUICallEngine SDK 1.x UIアップグレードステップなし

この部分はTRTCCalling 1.xからTUICallEngine 1.xへのアップグレードを支援します。

1. SDKパッケージの依存関係をアップグレードする

TUICallEngine SDKパッケージをインストールし、その他の依存関係を最新バージョンに更新します。

©2013-2022 Tencent Cloud. All rights reserved.



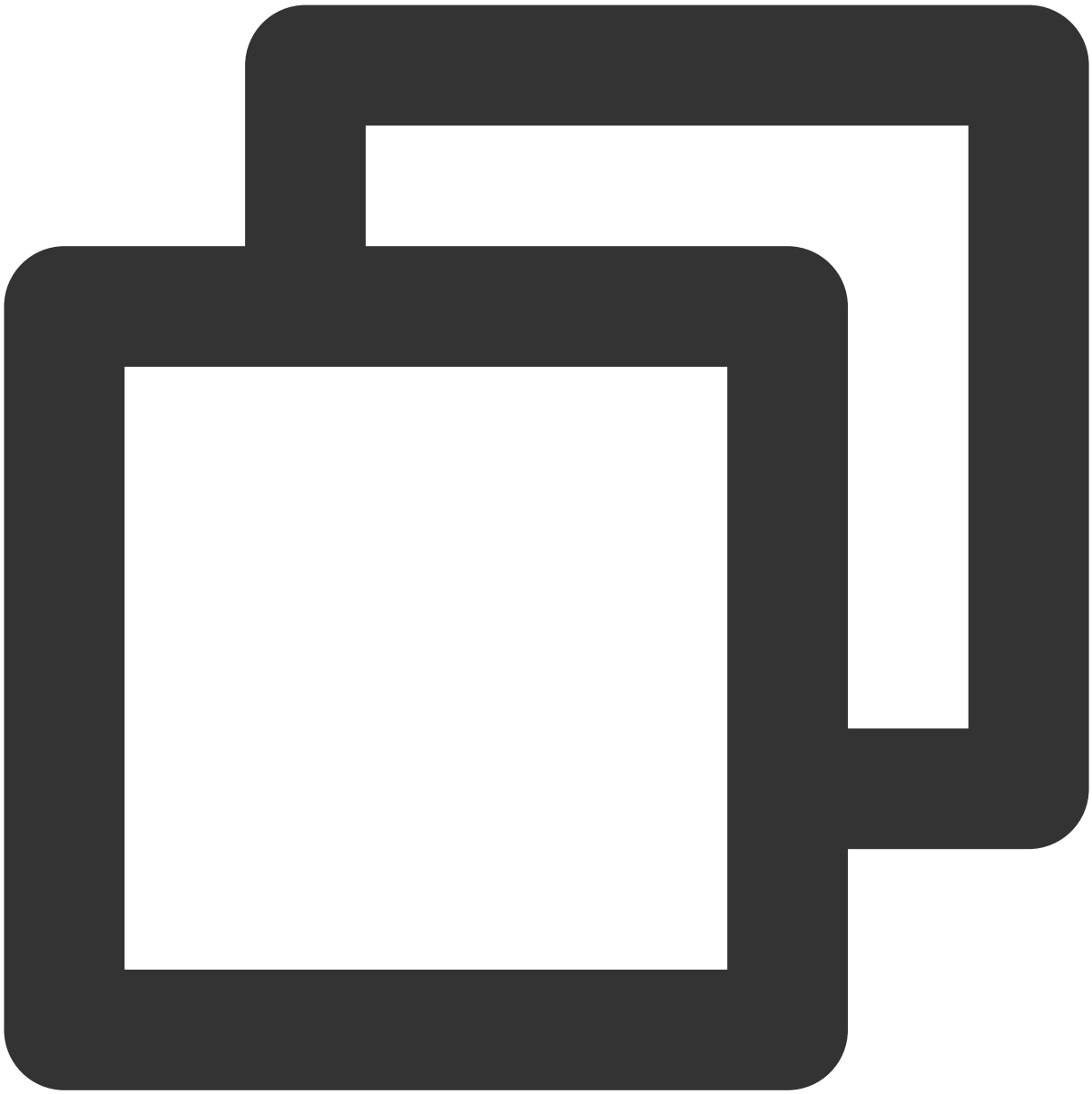
```
yarn add tuicall-engine-webrtc # 新バージョンのSDKをインストールします
yarn add tim-js-sdk@latest trtc-js-sdk@latest tsignaling@latest # 依存関係を最新バージ
yarn remove trtc-calling-js # 旧バージョンの SDKを削除します
# yarnをインストールしていない場合、先に以下を実行できます。npm install -g yarn
# 現時点でnpmを使用して依存関係の管理を行っている場合、「yarn add」を「npm install」に切り替え、
```

2. SDKの導入方法とインスタンスの作成方法を切り替える

上記のステップを完了した後に、プロジェクトが正常にコンパイルできない場合、次のステップはその中の競合している場所を修正する方法について詳しくご説明します。

©2013-2022 Tencent Cloud. All rights reserved.

1. まず、導入方法を切り替えます。

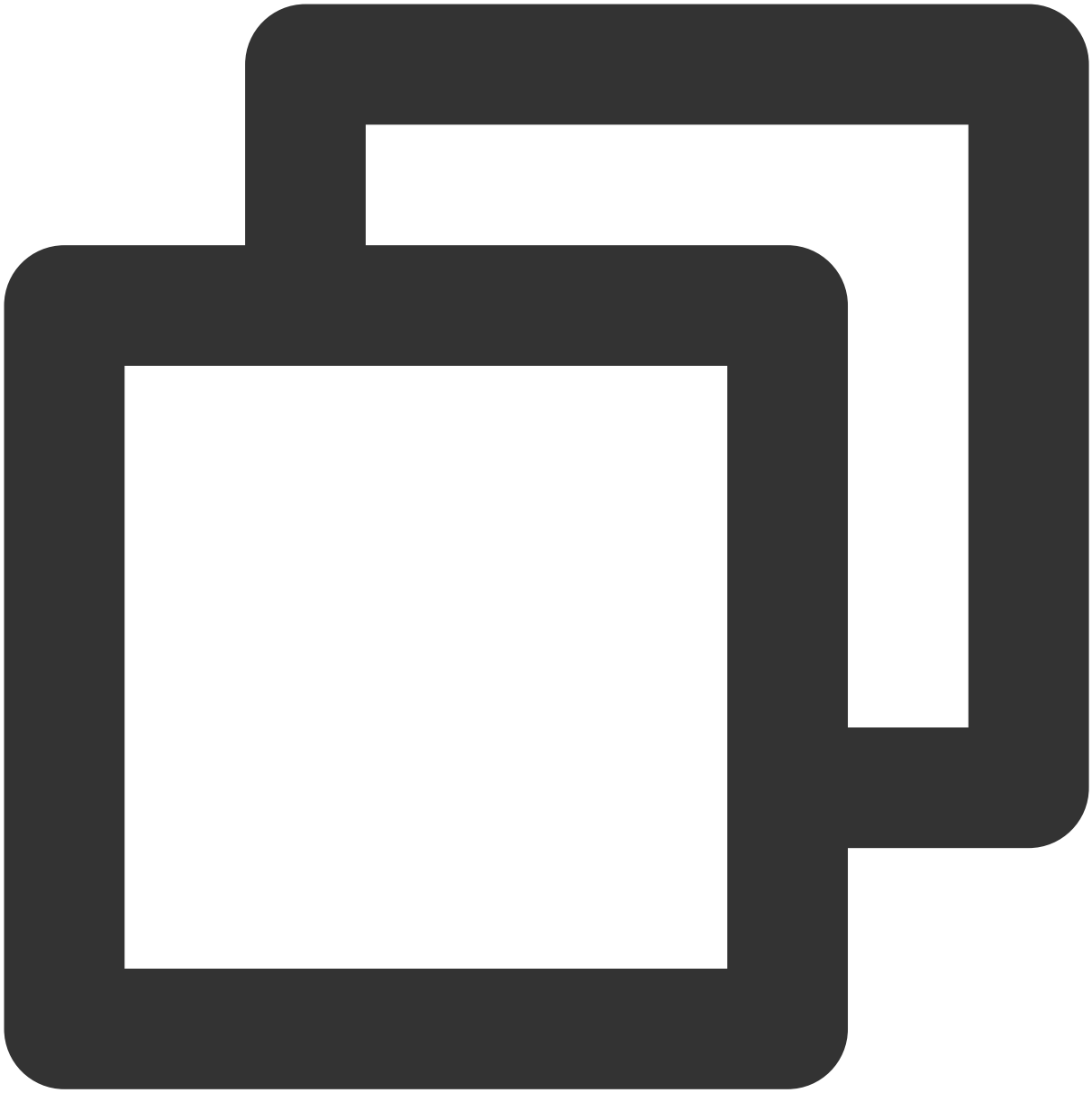


```
// 旧バージョン
import TRTCCalling from 'trtc-calling-js';

// 新バージョン
import { TUICallEngine } from "tuicall-engine-webrtc";
```

2. インスタンスの作成方法を修正します。サンプルコードは次のとおりです。詳細なインターフェースが必要な場合は、 createInstance()をご参照ください。

©2013-2022 Tencent Cloud. All rights reserved.



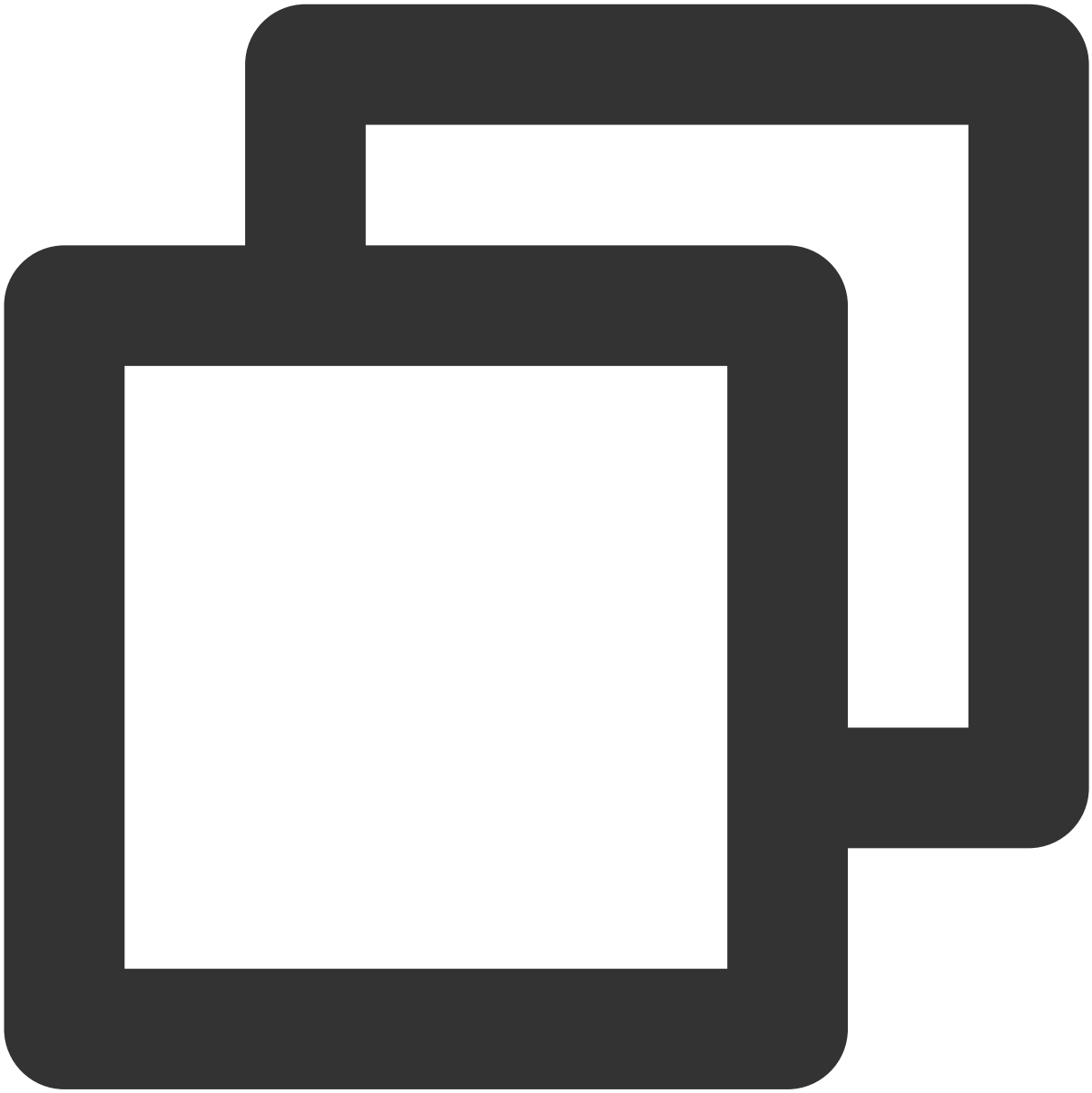
```
// 旧バージョン
new TRTCCalling({ SDKAppID });

// 新バージョン
TUICallEngine.createInstance({ SDKAppID });
// TUICallEngine.destroyInstance(); // インスタンスを削除する必要がある場合は、このインター
```

3. 監視イベント定数を置き換える

©2013-2022 Tencent Cloud. All rights reserved.

監視イベント定数アドレスをTRTCCalling.EVENTからTUICallEventに変更します。後者はtuicall-engine-webrtcパッケージから導入することができます。
このステップはエディタのグローバル検索を使用してクイック置換することをお勧めします。



```
// 旧バージョン
trtcCalling.on(TRTCCalling.EVNET.ERROR, this.handleError);
trtcCalling.on(TRTCCalling.EVNET.INVITED, this.handleNewInvitationReceived);
trtcCalling.on(TRTCCalling.EVNET.USER_ACCEPT, this.handleUserAccept);
...（その他の監視を含む）

// 新バージョン
```

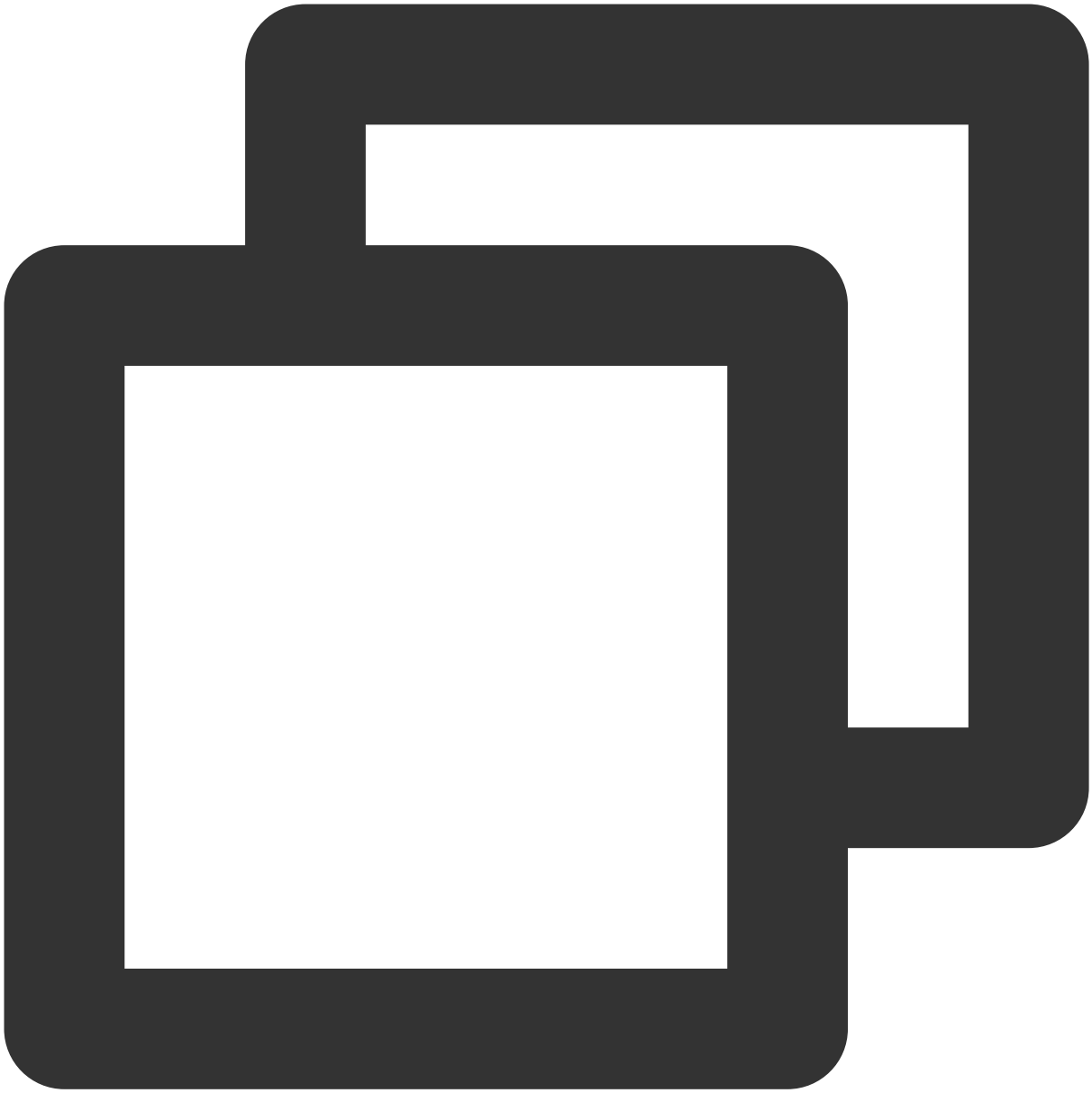
©2013-2022 Tencent Cloud. All rights reserved.

```
import { TUICallEvent } from "tuicall-engine-webrtc";
tuiCallEngine.on(TUICallEvent.ERROR, this.handleError); // tuiCallEngineは新しいインス
tuiCallEngine.on(TUICallEvent.INVITED, this.handleNewInvitationReceived);
tuiCallEngine.on(TUICallEvent.USER_ACCEPT, this.handleUserAccept);
...（その他の監視を含む）
```

4. 通話タイプ定数を置き換える

以前に通話タイプ定数を使用していない場合はこのステップをスキップできます。
通話タイプ定数アドレスをTRTCCalling.CALL_TYPEからTUICallTypeに変更します。後者はtuicall-engine-webrtcパッケージから導入することができます。
このステップはエディタのグローバル検索を使用してクイック置換することをお勧めします。

©2013-2022 Tencent Cloud. All rights reserved.



```
// 旧バージョン
this.TRTCCalling.CALL_TYPE.AUDIO_CALL // 音声タイプ
this.TRTCCalling.CALL_TYPE.VIDEO_CALL // ビデオタイプ

// 新バージョン
import { TUICallType } from "tuicall-engine-webrtc";
TUICallType.AUDIO_CALL // 音声タイプ
TUICallType.VIDEO_CALL // ビデオタイプ
```

5. その他の主要なインターフェースの変更

©2013-2022 Tencent Cloud. All rights reserved.

APIの意味	旧バージョンのTRTCCalling SDK	新バージョンのTUICallEngine SDK	備考
通話タイプの切り替え	trtcCalling.switchToAudioCall()	tuiCallEngine.switchCallMediaType()	名称変更は、パラメータによって切り替えタイプを選択します
マイクデバイスリストの取得	trtcCalling.getMicrophones()	tuiCallEngine.getDeviceList()	インターフェースの統合は、パラメータによってデバイスタイプを選択します
カメラデバイスリストの取得	trtcCalling.getCameras()	tuiCallEngine.getDeviceList()	インターフェースの統合は、パラメータによってデバイスタイプを選択します
マイクのオン/オフ	trtcCalling.setMicMute()	tuiCallEngine.closeMicrophone()	インターフェースの分割
マイクのオン/オフ	trtcCalling.setMicMute()	tuiCallEngine.openMicrophone()	インターフェースの分割
ユーザーのニックネームおよびプロフィール画像の設定	なし	tuiCallEngine.setSelfInfo()	ユーザーのニックネームおよびプロフィール画像の設定

©2013-2022 Tencent Cloud. All rights reserved.

AIノイズリダクションのオン/オフ	なし	tuiCallEngine.enableAIVoice()	4.12.1バージョン以降をサポートしています。具体的な使用についての詳細はAIノイズリダクションの使用をご参照ください

6. 補足説明

TUICallKitはIMオーディオビデオ通話機能パッケージをマッチングする必要があるため、「The package you purchased does not support this ability.」が発生する場合があります。その際は、IMコンソールをクリックし、対応するSDKAppIDアプリケーションの基本設定ページに進み、ページ右下隅のTRTCサービスをアクティブ化する機能エリアを見つけ、無料体験をクリックすると、TUICallKitの60日間無料トライアルサービスをアクティブ化することができます。正式なアプリケーションのリリースが必要な場合は、追加購入をクリックすると購入ページに進むことができます。
上記のAPIにアップグレードが完了すると、TUICallEngine SDKを正常に使用できるようになります。その他のAPIインターフェース呼び出しの問題が発生した場合、SDK APIドキュメントをご参照ください。

©2013-2022 Tencent Cloud. All rights reserved.

公開ログ（TUICallKit）

Web

最終更新日：：2023-02-17 16:34:15

Version 1.3.1 @2022.11.29

注意

このバージョンはSDK `tuicall-engine-webrtc@1.2.1` バージョンに依存しています。速やかに更新してください。

追加

スタイルの詳細を最適化します。

相手が通話に応答しない場合、相手を監視して通話タイプを変更することをサポートしています。

basic demoにデバイス検査機能を追加しました。

修正

通話終了時の内部ロジックによるエラーを修正しました。

Version 1.3.0 @2022.11.14

注意

このバージョンにアップグレードする際に、ご参照ください：アップグレードガイドライン。

追加

携帯電話H5の使用時に、自動的に縦画面スタイルに切り替えることをサポートしました。

電話をかけた時にローカルカメラをプレビューすることをサポートしました。

basic demoに電話をかける前のデバイス検査を追加しました。

修正

`TUICallKitServer.destroy()` を呼び出した後にtimインスタンスが完全に終了しない問題を修正しました。

相手が通話中に応答なしと表示される問題を修正しました。

vite環境下でtypescriptタイプのパッケージ化が成功しない問題を修正しました。

インターフェースの変更

`TUICallKitServer.call()` または `TUICallKitServer.groupCall()` を自発的に呼び出した時に、エラーが発生した場合、 `beforeCalling` コールバックを呼び出さなくなります。直接try catchを使用してエラーを捕捉してください。

©2013-2022 Tencent Cloud. All rights reserved.

Version 1.2.0 @2022.11.03

追加

新しいバージョンのTUICallEngine SDKに適応します。

Version 1.1.0 @2022.10.21

追加

通話中に、通話ページをフルスクリーンにすることができます。

通話中に、 `<TUICallKitMini/>` を使用して最小化することができます。

修正

既知の問題を修正し、安定性を向上させました。

Version 1.0.3 @2022.10.14

追加

basic demoにUserIDのクイックコピーを追加したため、ワンクリックで新しいウィンドウを開くことができます。

Version 1.0.2 @2022.09.30

追加

アクセスドキュメントを最適化し、デモ画像および詳細のガイドを追加しました。

修正

初めて入室した時に、デバイスのステータスビットが無効になる問題を修正しました。

webpackツールをパッケージ化する時に、アイコンのロードに失敗することがある問題を修正しました。

既知のスタイルの問題を修正しました。

Version 1.0.1 @2022.09.26

追加

電話をかけた時に、相手のマイクアイコンが非表示になります。

修正

basic demo SDKAppID入力ボックスが数字になる問題を修正しました。

©2013-2022 Tencent Cloud. All rights reserved.

Version 1.0.0 @2022.09.23

TUICallKit demoクイックスタート
TUICallKit クイックアクセス
TUICallKit API
TUICallKitインターフェースのカスタマイズガイド
TUICallKit (Web) に関するよくあるご質問

©2013-2022 Tencent Cloud. All rights reserved.

Android & iOS

最終更新日：：2022-11-21 17:43:39

説明：
新バージョン（1.1.0.103）のTUICallEngineインターフェースに関連する変更：

- 開発中に、maven latest更新メカニズムが原因でAndroid構築エラーが発生した場合は、次の2つの方法で解決できます。
  - TUICallKitを最新バージョンにアップグレードする。
  - tuicallkit/build.gradleのtuicallengineを1.0の固定バージョン：`com.tencent.liteav.tuikit:tuicallengine:1.0.0.53`に依存するよう変更する。
- 開発中に、pod updateの実行が原因でiOS構築エラーが発生した場合は、次の2つの方法で解決できます。
  - TUICallKitを最新バージョンにアップグレードする。
  - Podfileファイルに`pod 'TUICallEngine'、'1.0.0.53'`を追加する。

Version 1.1.0.103 @ 2022.09.30

- Android&iOS：グループ通話中のグループメンバー招待機能を最適化しました。
- Android&iOS：通話フローで、通話応答前にレコーディング、審査などの料金が発生しないように改善しました。
- Android&iOS：カスタムオーディオビデオ通話のオフラインプッシュメッセージをサポートしました。
- Android&iOS：一部の**TUICallEngine**インターフェースパラメータを変更しました。詳細については、call()、groupCall()、inviteUser()、およびコールバックインターフェースonCallReceived()をご参照ください。
- Android&iOS：グループ通話中に、少量のコールバックの異常が発生することがある問題を修正しました。
- Android&iOS：重複ログインおよびUserSigの期限切れによるステータス異常の問題を修正しました。
- iOS：TUCallKit Objective-CとSwiftの混合コンパイルの際に、`init`インターフェースエラーが発生する問題を修正しました。
- Android：Kotlinでのフローティングウィンドウ機能統合の際にコンパイルエラーが発生する問題を修正しました。

Version 1.0.0.53 @ 2022.08.15

新規リリース：

- Android&iOS：1v1オーディオビデオ通話、グループオーディオビデオ通話をサポートしました。
- Android&iOS：主要メーカーのオフライン通知をサポートしました。
- Android&iOS：プロフィール画像のカスタマイズ、ニックネームのカスタマイズをサポートしました。

©2013-2022 Tencent Cloud. All rights reserved.

- Android&iOS：通話中のフローティングウィンドウ有効化をサポートしました。
- Android&iOS：カスタム着信音の設定をサポートしました。
- Android&iOS：複数のプラットフォームにログインした状態での着信サービスをサポートしました。

©2013-2022 Tencent Cloud. All rights reserved.

よくある質問

Android

最終更新日：：2022-11-23 15:10:41

TUICallKitでIM SDKをインポートせず、TRTCだけを使用することはできますか。

できません。TUIKit全シリーズのコンポーネントはすべて、通信の基本サービス、例えば通話発信シグナル、通話中シグナルなどのコアロジックにTencent Cloud IM SDKを使用しています。他のIM製品をご購入済みの場合は、TUICallKitロジックを参照して適合させることも可能です。

「The package you purchased does not support this ability」というエラーが表示されました。

上記のエラーが表示された場合は、現在のアプリケーションのオーディオビデオ通話機能パッケージが期限切れまたはアクティブ化されていないことを表します。ステップ1：サービスのアクティブ化を参照してオーディオビデオ通話機能を取得またはアクティブ化し、引き続きTUICallKitコンポーネントをご利用ください。

TUICallKitがクラッシュし、クラッシュログが"No implementation found for xxxx"となっています。

スタック情報は次のとおりです。

```
java.lang.UnsatisfiedLinkError: No implementation found for void com.tencent.lite
av.base.Log.nativeWriteLogToNative(int, java.lang.String, java.lang.String) (trie
d Java_com_tencent_liteav_base_Log_nativeWriteLogToNative and Java_com_tencent_li
teav_base_Log_nativeWriteLogToNative__ILjava_lang_String_2Ljava_lang_String_2)
at com.tencent.liteav.base.Log.nativeWriteLogToNative(Native Method)
at com.tencent.liteav.base.Log.i(SourceFile:177)
at com.tencent.liteav.basic.log.TXCLog.i(SourceFile:36)
at com.tencent.liteav.trtccalling.model.impl.base.TRTCLogger.i(TRTCLogger.java:15
)
at com.tencent.liteav.trtccalling.model.impl.ServiceInitializer.init(ServiceIniti
alizer.java:36)
at com.tencent.liteav.trtccalling.model.impl.ServiceInitializer.onCreate(ServiceI
nitializer.java:101)
at android.content.ContentProvider.attachInfo(ContentProvider.java:2097)
at android.content.ContentProvider.attachInfo(ContentProvider.java:2070)
at android.app.ActivityThread.installProvider(ActivityThread.java:8168)
at android.app.ActivityThread.installContentProviders(ActivityThread.java:7709)
at android.app.ActivityThread.handleBindApplication(ActivityThread.java:7573)
at android.app.ActivityThread.access$2600(ActivityThread.java:260)
at android.app.ActivityThread$H.handleMessage(ActivityThread.java:2435)
at android.os.Handler.dispatchMessage(Handler.java:110)
at android.os.Looper.loop(Looper.java:219)
```

©2013-2022 Tencent Cloud. All rights reserved.

```
at android.app.ActivityThread.main(ActivityThread.java:8668)
at java.lang.reflect.Method.invoke(Native Method)
at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:513)
at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:1109)
```

注意：
TUICallkitは仮想マシンをサポートしていませんので、実機でご体験ください。実機で上記の異常が生じる原因は、TUICallKitの依存するTRTCなどのSDKの一部インターフェースにJNIを使用しているため、Android Studioがいくつかの条件でAPKコンパイルを行う際にNativeの.soライブラリをパッケージ化できず、このようなエラーが発生します。再Cleanを行ってください。

allowBackupの異常が発生しました。

エラーメッセージ：`Manifest merger failed : Attribute application@allowBackup value=(true) from AndroidManifest.xml`

問題の原因：TUICallKitが依存するIM SDKではデフォルトのallowBackupの値はfalseであり、これはアプリケーションのバックアップと復元機能が無効になっていることを意味します。

対処方法：プロジェクトの `AndroidManifest.xml` ファイルからallowBackup属性を削除するか、またはこの属性をfalseに変更し、バックアップと復元機能が無効にすることができます。また、 `AndroidManifest.xml` ファイルのapplicationノードに `tools:replace="android:allowBackup"` を追加することもでき、これはIM SDKの設定を上書きし、ご自身の設定を使用することを意味します。

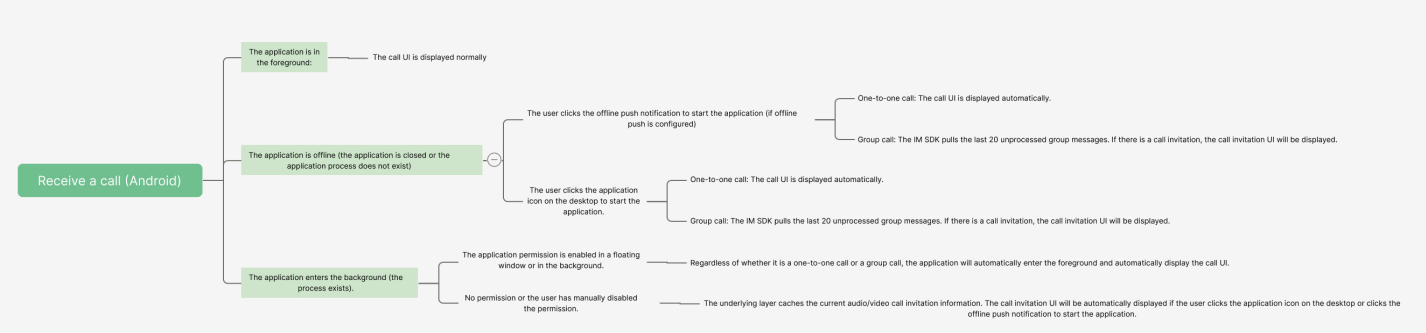
TUICallKitはオフラインプッシュをサポートしていますか。

サポートしています。AndroidのオフラインプッシュはTencent CloudのTUIOfflinePushコンポーネントを使用しています。詳細については、TUIOfflinePushをご参照ください。TUICallKitコンポーネントにもこのコンポーネントが接続されています。

通話招待のタイムアウト時間内に、被招待者がオフラインから再びオンラインとなった場合、通話画面をポップアップさせることはできますか。

©2013-2022 Tencent Cloud. All rights reserved.

Appの起動タイプによって状況がそれぞれ異なります。



オフラインから再びオンラインとなった際に通話画面がポップアップしなかった場合は、`onReceiveNewInvitation` ログをフィルタリングし、メッセージ履歴にプルしていないかどうかを確認してください。このログプリントがない場合は、お問い合わせください。

アプリケーションがバックエンドにあるとき、通話画面をフロントエンドに自動的にプルできません

アプリケーションをバックエンドからフロントエンドに自動的にプルする場合、Appが「バックエンド自動起動」または「フローティングウィンドウ」権限を有効にしているかどうかを確認する必要があります。

メーカーが異なる場合や、同じメーカーであってもAndroidのバージョンが異なる場合は、アプリケーションに許可する権限や権限名が一致しないことがありますのでご注意ください。例えば、Xiaomi 6ではバックエンドの画面ポップアップ権限を有効にすればよいだけですが、Redmiではバックエンド画面ポップアップとフローティングウィンドウの権限を同時に有効にする必要があります。

> 説明：
> テスト中に手動ですべての権限を有効化しても、通話画面のフロントエンドへの自動プルができない場合は、メーカー間の互換性の問題がある可能性があります。

©2013-2022 Tencent Cloud. All rights reserved.

iOS

最終更新日：：2022-11-21 16:34:37

「The package you purchased does not support this ability」というエラーが表示されました。

上記のエラーが表示された場合は、現在のアプリケーションのオーディオビデオ通話機能パッケージが期限切れまたはアクティブ化されていないことを表します。クイックアクセスのステップ1：サービスのアクティブ化を参照し、オーディオビデオ通話機能を取得またはアクティブ化することで、引き続きTUICallKitコンポーネントを使用することができます。

オーディオビデオ通話パッケージを購入するにはどうすればよいですか。

購入リンクオーディオビデオ通話SDK価格一覧をご参照ください。その他にご質問がありましたらページ右側をクリックし、パッケージのご購入前にお問い合わせください。

TUICallKitソースコードを変更するにはどうすればよいですか。

`CocoaPods` を使用してコンポーネントをインポートします。具体的な手順は次のとおりです。

1. プロジェクトの `Podfile` ファイルと同じ階層のディレクトリ下に `TUICallKit` フォルダを作成します。

2. クリックして**Github/TUICallKit**に進み、コードのクローン/ダウンロードを選択した後、**iOS**ディレクトリ下の `TUICallKit` 、 `Resources` フォルダおよび `TUICallKit.podspec` ファイルを、 `ステップ1` で作成した `TUICallKit` フォルダ下にコピーします。

3. `Podfile` ファイルに次の依存を追加します。

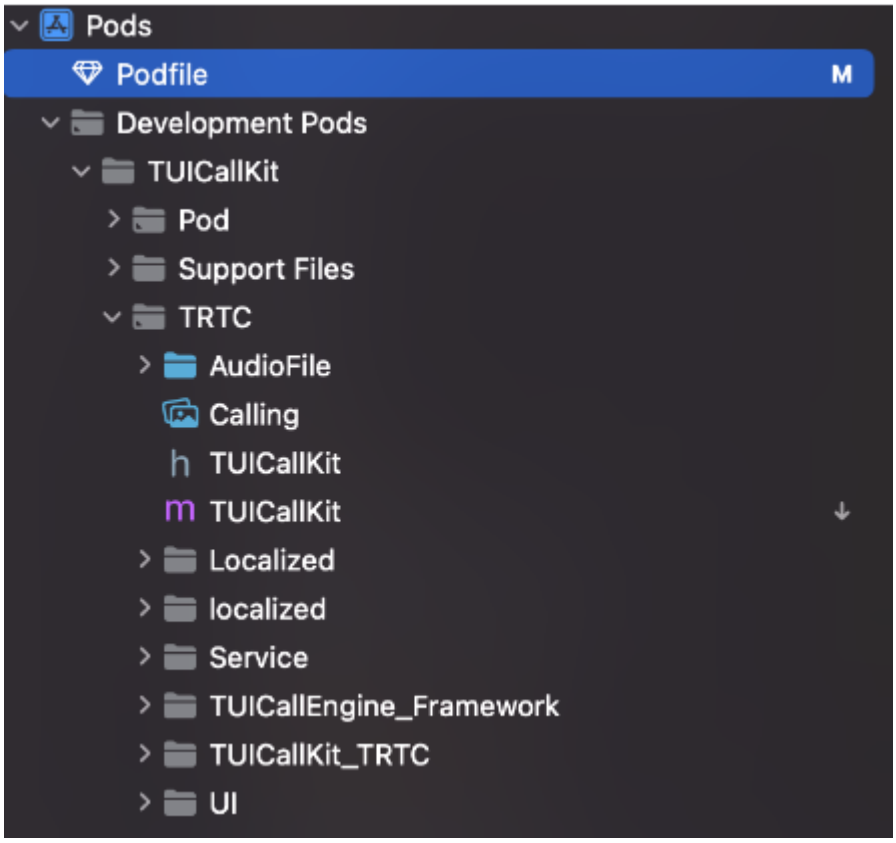
```
# :path => "TUICallKit.podspecを指定する相対パス"
pod 'TUICallKit', :path => "TUICallKit/TUICallKit.podspec"
```

4. `pod install` コマンドを実行し、インポートを完了します。

> 注意：
> `TUICallKit` 、 `Resources` フォルダと `TUICallKit.podspec` ファイルは同一のディレクトリ下にある必要があります。

TUICallKitコンポーネント統合後の効果：

©2013-2022 Tencent Cloud. All rights reserved.



説明：
TUICallKitコンポーネントが統合されると、フォルダの階層化表示がサポートされ、ソースコードの閲覧と変更がしやすくなります。

TUICallKitと自身で統合したオーディオビデオライブラリは競合しますか。

Tencent Cloudの `オーディオビデオライブラリ` は同時に統合できないため、シンボルの競合が発生する可能性があります。ケースごとに、次のように処理することができます。

1. `TXLiteAVSDK_TRTC` ライブラリを使用している場合は、シンボル競合は発生しません。 `Podfile` ファイルに次の依存を直接追加できます。

```
pod 'TUICallKit'
```

2. `TXLiteAVSDK_Professional` ライブラリを使用している場合は、シンボル競合が発生します。 `Podfile` ファイルに次の依存を追加できます。

```
pod 'TUICallKit/Professional'
```

©2013-2022 Tencent Cloud. All rights reserved.

3. `TXLiteAVSDK_Enterprise` ライブラリを使用している場合は、シンボル競合が発生します。 `TXLiteAVSDK_Professional` にアップグレードしてから `TUICallKit/Professional` を使用することをお勧めします。

TUICallKit統合後に実行エラー「ld: framework not found BoringSSL clang: error: linker command failed with exit code 1 sdk」が表示されました

`TUICallKit` の依存するオーディオビデオライブラリは現時点ではエミュレーターをサポートしていません。実機で実行またはデバッグを行ってください。

TUICallKitでIM SDKをインポートせず、TRTCだけを使用することはできますか。

できません。 `TUIKit` 全シリーズのコンポーネントはすべて、通信の基本サービス、例えば通話発信シグナル、通話中シグナルなどのコアロジックにTencent Cloud IM SDKを使用しています。他の `IM` 製品をご購入済みの場合は、 `TUICallKit` ロジックを参照して適合させることも可能です。

TUICallKitコンポーネントは着信音のカスタマイズをサポートしていますか。

サポートしています。TUICalling#setCallingBellを呼び出して行うことができます。

CocoaPodsをインストールするにはどうすればよいですか。

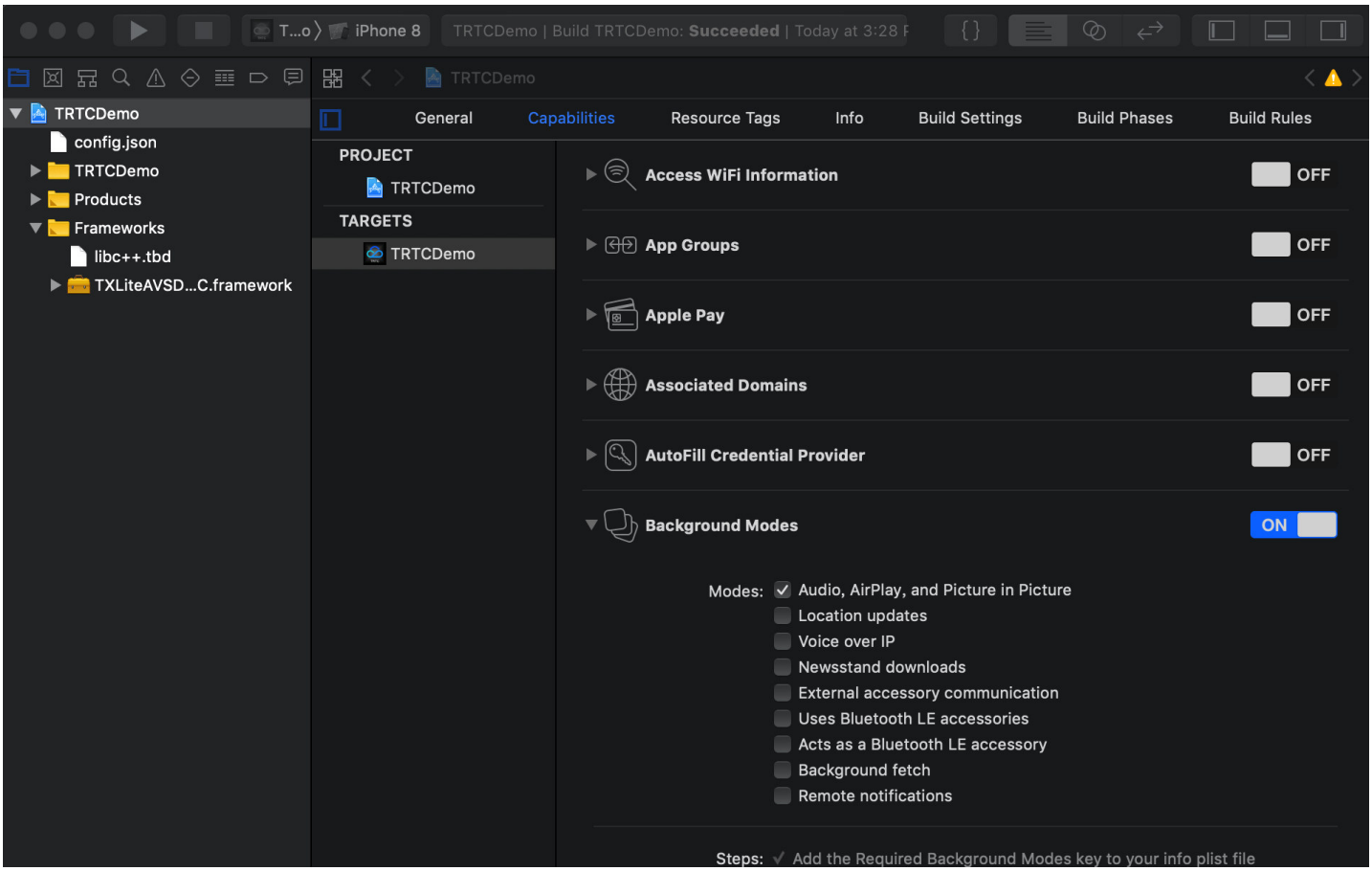
ターミナルポートに以下のコマンド（事前に `Mac` に `Ruby` 環境をインストールする必要があります）を入力します。

```
sudo gem install cocoapods
```

TUICallKitはバックエンドでの実行をサポートしていますか。

サポートしています。バックエンドに入っても引き続き関連機能を実行する必要がある場合は、現在のプロジェクトを選択できます。下図に示すように、 **Capabilities** の下に**Background Modes** を設定して **ON**を開いて **Audio，AirPlay and Picture in Picture** にチェックを入れます。

©2013-2022 Tencent Cloud. All rights reserved.



©2013-2022 Tencent Cloud. All rights reserved.

Web

最終更新日：：2022-11-21 16:34:38

1、基本環境についての質問

Web端末SDKはどのブラウザをサポートしていますか。

TRTC Web SDKのブラウザサポートの詳細については、[TRTC Web SDKのブラウザサポート状況をご参照ください。上記に記載がない環境については、現在のブラウザでTRTC機能テストを開き、WebRTCの機能が完全にサポートされているかどうかテストすることができます。

現在のネットワーク状況をリアルタイムに検出するにはどうすればよいですか。

具体的には、通話前のネットワーク品質テストをご参照ください。

ローカルの開発テストでは正常に使用できたのに、オンラインにデプロイしてIPでアクセスするとビデオ/音声通話が正常に行えなくなりましたが、なぜですか。

ユーザーに対するセキュリティ、プライバシーなどの観点から、ブラウザの制限ページは安全な環境下（例えば、`https`、`localhost`、`file://`などのプロトコル）でなければ、マイク、カメラのキャプチャが行えないようになっています。HTTPプロトコルは安全ではないため、ブラウザはHTTPプロトコルでのメディアデバイスのキャプチャを禁止する場合があります。

ローカルの開発テストではすべて正常でも、ウェブページのデプロイ後にカメラ、マイクを正常にキャプチャできなくなった場合は、ウェブページがHTTPプロトコルにデプロイされているかどうかを確認してください。もしデプロイされていなければ、HTTPSを使用してウェブページのデプロイを行ってください。その場合は適切なHTTPSセキュリティ証明書があることを確認してください。

より詳しい情報については、URLドメイン名およびプロトコル制限の説明をご参照ください。

接続中に、「is not included in the current tim's package」とのエラーが出ます。

原因：

ダウンロードしたTIMの依存パッケージのバージョンが低すぎる可能性があります。

TIM依存パッケージのバージョンが正しい場合は、SDKAppIDがオーディオビデオパッケージを購入していない、もしくは呼び出した機能がパッケージでサポートされていない可能性があります。

対処方法

TIMパッケージのバージョンを>=2.21.2にアップグレードします。

オーディオビデオパッケージをサポートしているSDKAppIDを購入します。

©2013-2022 Tencent Cloud. All rights reserved.

2、統合についての質問

tuiCallEngine.handup()で「uncaught (in promise) TypeError: cannot read property 'stop' of null」とのエラーが出ます。

原因：ユーザーがイベントの監視中に何度もhandup()を呼び出したため、hangupが実行を完了しないうちにまたトリガーされたことによるものです。

対処方法：handup()の実行は1回だけでよいです。イベント監視後の操作はTRTCCalling内部ですでに処理されているため、hangup()メソッドを再度実行する必要はなく、業務に関連する操作を行うだけで済みます。

接続中に、「TypeError: Cannot read property 'getVideoTracks' of null」とのエラーが出ます。

原因：ユーザーが受信時に、ユーザービデオおよびマイクの使用権限を取得していないために起こったものです。

対処方法：startRemoteView、startLocalViewなどのデバイス操作メソッドを使用している場合は、非同期のメソッドを使用することをお勧めします。TRTCCalingを最新バージョンにバージョンアップしてください。

sdkAppidをscript方式でインポートする際に、「TSignaling._onMessageReceived unknown bussinessID=undefined」と通知されます。

詳細：同一のsdkAppidをscript方式でインポートした場合、scriptでインポートしたもの同士は相互接続が可能ですが、npmでインポートしたもの、またはAndroid/iOSのものとは相互接続できず、かつ警告メッセージ `TSignaling._onMessageReceived unknown bussinessID=undefined` が返されます。

原因： `bussinessId=undefined` は、そのバージョンのtsignalingが旧バージョンであり、旧バージョンのシグナルに問題があることを表しています。

対処方法：tsignalingをバージョンアップします。また、インポートの過程では**新バージョンのtsignalingのファイル名が `tsignaling-js` になっているかに注意する必要があります**。

「Uncaught (in promise) RTCError: duplicated play() call observed, please stop() firstly <invalid_operation 0x1001="">」と通知されます。

原因：音声通話中に、startRemoteViewインターフェースを呼び出したためです。

対処方法：音声通話中に、startRemoteView操作をキャンセルします。

Error: tuiCallEngine.call - ユーザーデバイス権限の取得に失敗しました。

原因：TRTCCallingにデバイス権限がない、またはデバイスに対応していないためです。

対処方法

TRTCデバイス検査を使用してチェックを行います。

©2013-2022 Tencent Cloud. All rights reserved.

Chromeのウェブサイト設定（chrome://settings/content）にアクセスし、TRTCCallingを使用しているウェブサイトがカメラ/マイクの権限をオンにしているかどうかを確認します。

TUICallEngine webはオフラインメッセージの受信をサポートしていますか。

オフラインメッセージの受信はサポートしていません。

オフラインメッセージプッシュをサポートするには、call/groupCallのofflinePushInfoでプッシュしたいメッセージを追加してください。

Error: TRTCClient.getMediaDevicesAuth - failed to get user video steam - NotReadableError: Could not start video source?

原因：システムにブラウザのカメラをオンにする権限がありません。

対処方法：システムの設定でカメラを見つけ、対応するブラウザの権限を有効にします。

©2013-2022 Tencent Cloud. All rights reserved.

多人数ビデオミーティング
TUIRoom（Web）の統合

最終更新日：：2022-10-27 15:15:28

コンポーネントの説明

TUIRoomは、UIを含むオープンソースのオーディオビデオコンポーネントであり、TUIRoomの統合により、業務にオーディオビデオルーム、画面共有、チャットなどの機能を速やかに立ち上げることができます。Web側のTUIRoomの基本機能は下図のとおりです。

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになりますs。



TUIRoomオンラインリンクをクリックすると、TUIRoomのその他の機能を体験できます。
GithubをクリックしてTUIRoomコードをダウンロードし、TUIRoom Webデモプロジェクトのクイックスタートドキュメントを参照してTUIRoom Webデモプロジェクトをクイックスタートできます。
Web側のTUIRoomコンポーネントを既存のビジネスに統合する場合は、このドキュメントをご参照ください。

©2013-2022 Tencent Cloud. All rights reserved.

コンポーネントの統合

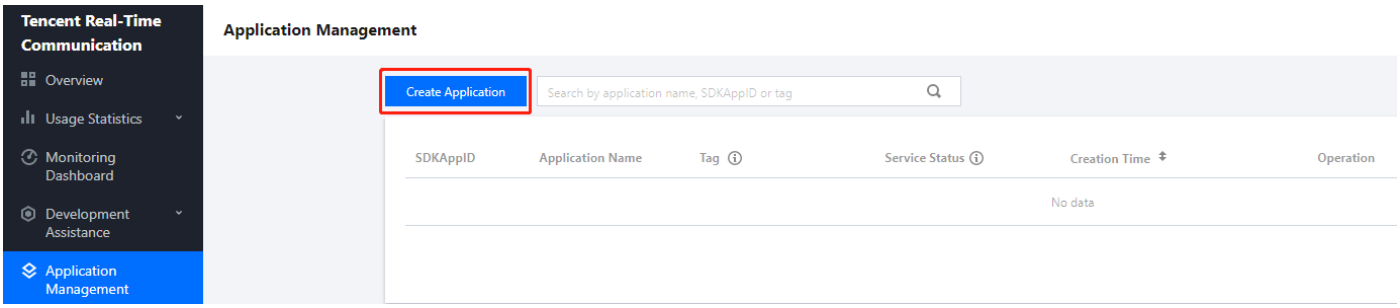
TUIRoomコンポーネントは、Vue3+TS+Pinia+Element Plus+SCSSを使用して開発されており、インポート先のプロジェクトにはVue3+TSテクノロジースタックが必要です。

ステップ1：Tencent Cloud TRTCとInstant Messagingサービスを有効化します

TUIRoomは、Tencent Cloud TRTCとInstant Messagingサービスをベースに開発されたものです。

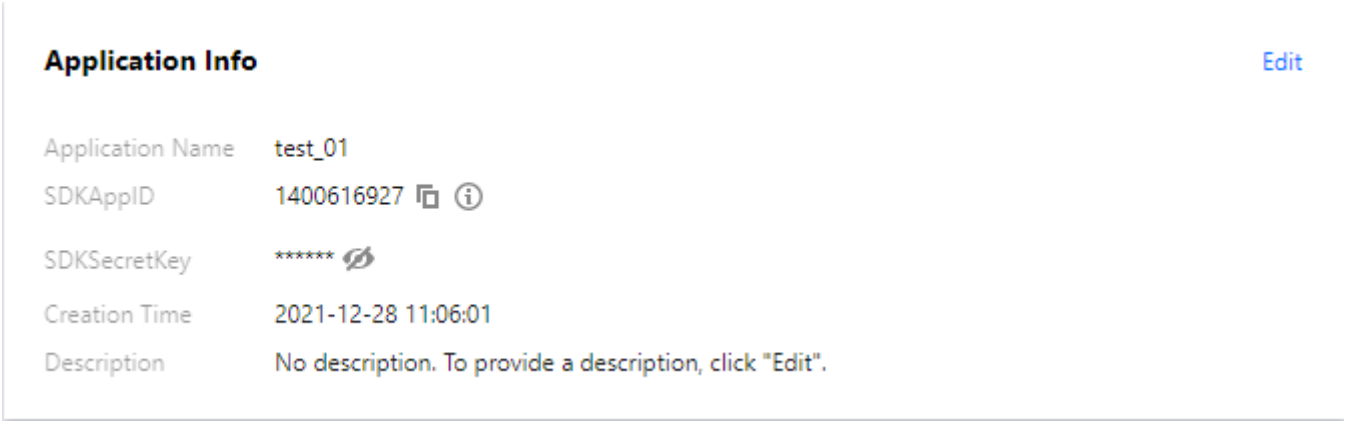
ステップ1. **TRTCアプリケーションを作成します**

- Tencent Cloudアカウントがない場合は、Tencent Cloudアカウントの登録を行ってください。
- 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。



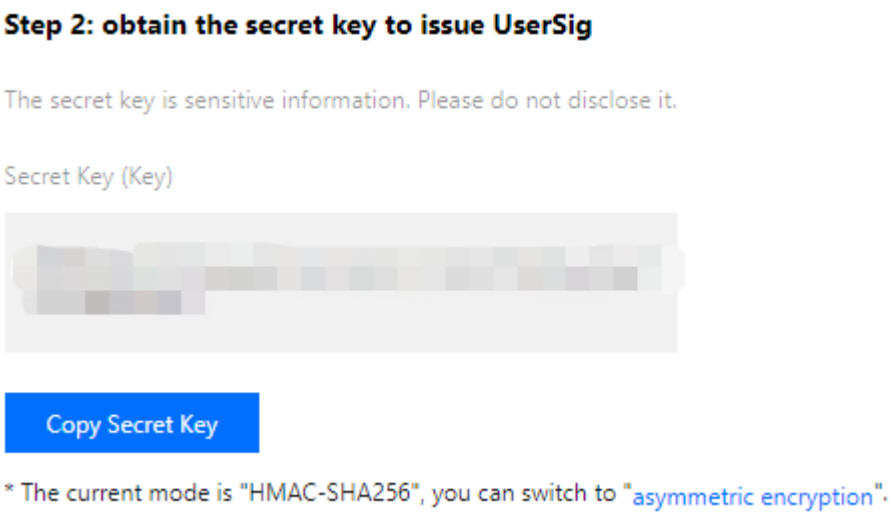
2. **TRTCキー情報およびキー情報の取得**

3. **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。SDKAppIDは、TRTCのアプリケーションIDであり、サービス分離のために使用され、すなわち、異なるSDKAppIDによる通話が相互に通じません。



©2013-2022 Tencent Cloud. All rights reserved.

4. **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。SecretKeyはTRTCのアプリケーションキーであり、SDKAppIDとペアで使用してください。TRTCサービスを正当に使用するための認証用チケットUserSigをチェックアウトします。



5. **UserSigの発行**
   UserSigとは、悪意ある攻撃者によるクラウドサービスの使用権の盗用を防ぐために、Tencent Cloudによって設計されたセキュリティ保護された署名です。TUIRoomを初期化するには、UserSigパラメータを提供してください。
   - デバッグスタート段階でuserSigを発行する方法については、デバッグスタート段階でのUserSigの計算方法をご参照ください。
   - 本番環境でuserSigを発行する方法については、本番環境でのUserSigの計算方法をご参照ください。

ステップ2：TUIRoomコンポーネントのダウンロードとコピー

1. Githubをクリックして、TUIRoomウェアハウスコードをクローンまたはダウンロードします。
2. ビジネスサイドで既存のVue3 + TSプロジェクトを開き、ViteとWebpackのパッケージ方式をサポートします。Vue3 + TSのプロジェクトがない場合、以下のいずれかの方法を選択して、テンプレートプロジェクトを生成することができます。
   - Vue3 + Vite + TSのテンプレートプロジェクトの生成
   - Vue3 + Webpack + TSテンプレートプロジェクトの生成

```
npm create vite@latest TUIRoom-demo -- --template vue
```

©2013-2022 Tencent Cloud. All rights reserved.

> 注意
> テンプレートプロジェクトスクリプトの生成を実行する場合、ステップ1で直接リターンし、ステップ2でVueを選択し、ステップ3でvue-tsを選択します。

Vue3 + Vite + TSテンプレートプロジェクトの生成に成功したら、以下のスクリプトを実行します。

```
cd TUIRoom-demo
npm install
npm run dev
```

3. `TUIRoom/Web/src/TUIRoom` フォルダを既存のプロジェクトsrc/ディレクトリにコピーします。

ステップ3：TUIRoomコンポーネントのインポート

1. ページでTUIRoomコンポーネントを参照します。例：TUIRoomコンポーネントを `App.vue` コンポーネントにインポートします。
   - TUIRoomコンポーネントは、ユーザーをキャスターロールと一般メンバーロールに分けます。コンポーネントは外部にinit、createRoom、enterRoomの各メソッドを提供します。
   - キャスターと一般メンバーはinitメソッドを使用してTUIRoomコンポーネントの適用とユーザーデータを初期化できます。キャスターはcreateRoomメソッドを使用してルームを作成して参加します。一般メンバーはenterRoomメソッドを使用してキャスターが作成したルームに参加できます。

```
<template>
<room ref="TUIRoomRef"></room>
</template>
<script setup lang="ts">
import { ref, onMounted } from 'vue';
// TUIRoomコンポーネントをインポートします。インポートパスが正しいことを確認してください
import Room from './TUIRoom/index.vue';
// TUIRoomコンポーネントのメソッドを呼び出すために、TUIRoomコンポーネント要素を取得します
const TUIRoomRef = ref();
onMounted(async () => {
// TUIRoomコンポーネントの初期化
// キャスターはルームを作成する前にTUIRoomコンポーネントを初期化してください
// 一般メンバーは、ルームに参加する前にTUIRoomコンポーネントを初期化してください
await TUIRoomRef.value.init({
// sdkAppIdを取得するにはステップ1をご参照ください
sdkAppId: 0,
// ビジネスにおけるユーザーの一意の識別ID
userId: '',
// ローカル開発デバッグは、userSigをhttps://console.intl.cloud.tencent.com/trtc/usersigtoolページにすばやく生成します。userSigとuserIdは1対1で対応していることに注意して
```

©2013-2022 Tencent Cloud. All rights reserved.

```
ください
userSig: '',
// ユーザーがビジネスで使用するニックネーム
userName: '',
// ユーザーがビジネスで使用するアバターへのリンク
userAvatar: '',
// 画面共有のためにユーザーが使用する一意のID。shareUserId=`share_${userId}`が必要です。画面共有機能が要求されない場合は渡す必要がありません
shareUserId: '',
// このドキュメントのステップ1>3を参照し、sdkAppIdとshareUserIdを使用してshareUserSigを発行してください
shareUserSig: '',
})
// デフォルトではルームの作成が実行され、実際のインポートはオンデマンドでhandleCreateRoomメソッドが実行されます
await handleCreateRoom();
})
// キャスターがルームを作成します。このメソッドはルームが作成されたときにのみ呼び出されます
async function handleCreateRoom() {
// roomIdはユーザーが参加するルーム番号です。roomIdタイプはnumber型が必要です
// roomModeは、'FreeSpeech'（自由発言モード）と'ApplySpeech'（挙手発言モード）の2モードを含み、デフォルトは'FreeSpeech'です。現在、自由発言モードだけをサポートします
// roomParamは、ユーザがルームに参加したときのデフォルトの動作を指定します（デフォルトでマイクをオンにするかどうか、デフォルトでカメラをオンにするかどうか、デフォルトのメディアデバイスID）
const roomId = 123456;
const roomMode = 'FreeSpeech';
const roomParam = {
isOpenCamera: true,
isOpenMicrophone: true,
}
await TUIRoomRef.value.createRoom(roomId, roomMode, roomParam);
}
// 一般メンバーがルームに参加します。このメソッドは作成されたルームに一般メンバーが参加するときに呼び出されます
async function handleEnterRoom() {
// roomIdはユーザーが参加するルーム番号です。roomIdタイプはnumber型が必要です
// roomParamは、ユーザがルームに参加したときのデフォルトの動作を指定します（デフォルトでマイクをオンにするかどうか、デフォルトでカメラをオンにするかどうか、デフォルトのメディアデバイスID）
const roomId = 123456;
const roomParam = {
isOpenCamera: true,
isOpenMicrophone: true,
}
await TUIRoomRef.value.enterRoom(roomId, roomParam);
}
```

©2013-2022 Tencent Cloud. All rights reserved.

```
</script>
<style>
html, body {
width: 100%;
height: 100%;
margin: 0;
}
#app {
width: 100%;
height: 100%;
}
</style>
```

注意：
ページで上記のコードをコピーした後、TUIRoomインターフェースのパラメータを実際のデータに変更する必要があります。

ステップ4：開発環境の設定

TUIRoomコンポーネントがインポートされたら、プロジェクトが正常に機能するように、次の設定を行ってください

- Vue3 + Vite + TSプロジェクトの開発環境の設定
- Vue3+Webpack+Tsプロジェクトの開発環境の設定

1. 依存関係のインストール

- 開発環境の依存関係をインストールします。

```
npm install sass typescript unplugin-auto-import unplugin-vue-components -S -D
```

- 本番環境の依存関係をインストールします。

```
npm install element-plus events mitt pinia rtc-beauty-plugin tim-js-sdk trtc-js-sdk tsignaling -S
```

2. Piniaの登録

TUIRoomはPiniaを使用してルームのデータ管理を行います。Piniaをプロジェクトエントリファイルに登録してください。プロジェクトエントリファイルは `src/main.ts` ファイルです。

©2013-2022 Tencent Cloud. All rights reserved.

```
// src/main.tsファイル
import { createPinia } from 'pinia';
const app = createApp(App);
// pinia登録
app.use(createPinia());
app.mount('#app');
```

3. **element-plusをオンデマンドでインポートするための設定**

- TUIRoomはelement-plus UIコンポーネントを使用します。すべてのelement-plusコンポーネントがインポートされないようにするには、 `vite.config.ts` でelement-plusコンポーネントをオンデマンドでロードするように設定してください。

> 注意
> 以下の設定項目は増分設定です。すでに存在するVite設定項目を削除しないでください。

```
// vite.config.ts
import AutoImport from 'unplugin-auto-import/vite';
import Components from 'unplugin-vue-components/vite';
import { ElementPlusResolver } from 'unplugin-vue-components/resolvers';
export default defineConfig({
// ...
plugins: [
AutoImport({
resolvers: [ElementPlusResolver()],
}),
Components({
resolvers: [ElementPlusResolver({
importStyle: 'sass',
})],
}),
],
css: {
preprocessorOptions: {
scss: {
// ...
additionalData: `
@use "./src/TUIRoom/assets/style/element.scss" as *;
`,
},
```

©2013-2022 Tencent Cloud. All rights reserved.

```
},
},
});
```

- また、UI付きのelement-plus コンポーネントがスタイルを正しく表示するためには、エントリファイル `src/main.ts` にelement-plusコンポーネントスタイルをロードしてください。

```
// src/main.ts
import 'element-plus/theme-chalk/el-message.css';
import 'element-plus/theme-chalk/el-message-box.css';
```

ステップ5：開発環境の実行

コンソールで開発環境実行スクリプトを実行し、ブラウザを使用してTUIRoomを含むページを開くと、ページ内でTUIRoomコンポーネントを使用できます。

- ステップ2のスクリプトを使用してVue + Vite + TSプロジェクトを生成する場合は、次の事項を行う必要があります。

1. 開発環境コマンドを実行します。

```
npm run dev
```

2. ブラウザでページ `http://localhost:3000/` を開きます。

> 注意：
> TUIRoomはオンデマンドでelement-plusコンポーネントをインポートするため、開発環境のルーティングページが最初にロードされたときの反応が遅くなり、element-plusがオンデマンドでロードされるのを待つと正常に機能するようになります。element-plusのオンデマンドロードは、パッケージング後のページロードには影響しません。

3. TUIRoomコンポーネント機能を体験します。

- ステップ2のスクリプトを使用してVue + Webpack + TSプロジェクトを生成する場合は、次の事項を行う必要があります。

1. 開発環境コマンドを実行します。

```
npm run serve
```

©2013-2022 Tencent Cloud. All rights reserved.

2. ブラウザで `http://localhost:8080/` のページを開きます

> 注意：
> 実行中、src/TUIRoomディレクトリでeslintエラーが発生した場合、.eslintignoreファイルに/src/TUIRoomのパスを追加すると、eslintチェックをブロックできます。

3. TUIRoomコンポーネント機能を体験します。

ステップ6：本番環境のデプロイ

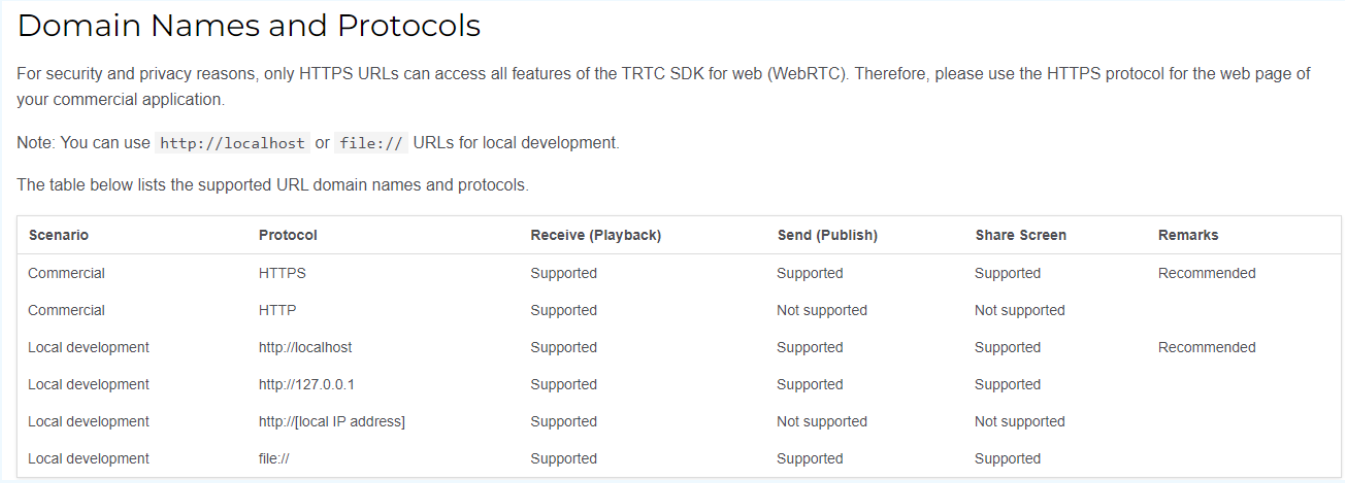
1. distファイルのパッケージ

```
npm run build
```

> 説明：
> 実際のパッケージコマンドについてはpackage.jsonファイルをご確認ください。

2. distファイルのサーバーへのデプロイ

> 注意：
> 本番環境ではHTTPSドメイン名を使用する必要があります。



Domain Names and Protocols

For security and privacy reasons, only HTTPS URLs can access all features of the TRTC SDK for web (WebRTC). Therefore, please use the HTTPS protocol for the web page of your commercial application.

Note: You can use `http://localhost` or `file://` URLs for local development.

The table below lists the supported URL domain names and protocols.

Scenario	Protocol	Receive (Playback)	Send (Publish)	Share Screen	Remarks
Commercial	HTTPS	Supported	Supported	Supported	Recommended
Commercial	HTTP	Supported	Not supported	Not supported	
Local development	http://localhost	Supported	Supported	Supported	Recommended
Local development	http://127.0.0.1	Supported	Supported	Supported	
Local development	http://[local IP address]	Supported	Not supported	Not supported	
Local development	file://	Supported	Supported	Supported	

©2013-2022 Tencent Cloud. All rights reserved.

付録：TUIRoom API

TUIRoomインターフェース

init

TUIRoomデータを初期化します。TUIRoomを使用するすべてのユーザーはinitメソッドを呼び出してください。

```
TUIRoomRef.value.init(roomData);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomData	object	
roomData.sdkAppId	number	お客様のSDKAppId
roomData.userId	string	ユーザーの唯一ID
roomData.userSig	string	ユーザーのUserSig
roomData.userName	string	ユーザーのニックネーム
roomData.userAvatar	string	ユーザーのプロフィール写真
roomData.shareUserId	string	非必須。ユーザーが画面共有を行うUserIdです。 shareUserId=`share_${userId}`が要求され、画面共有機能がない場合は渡されません
roomData.shareUserSig	string	非必須。ユーザーが画面共有を行うUserSig

createRoom

キャスターがルームを作成します。

```
TUIRoomRef.value.createRoom(roomId, roomMode, 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	number	ルームID

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomMode	string	ルームモードです。'FreeSpeech'（自由発言モード）と'ApplySpeech'（挙手発言モード）を含み、デフォルトは'FreeSpeech'です。現在、自由発言モードだけをサポートします
roomParam	Object	非必須
roomParam.isOpenCamera	string	非必須。入室はカメラをオンにするかどうか。デフォルトはオフです
roomParam.isOpenMicrophone	string	非必須。入室はマイクをオンにするかどうか。デフォルトはオフです
roomParam.defaultCameraId	string	非必須。デフォルトはカメラ装置のID
roomParam.defaultMicrophoneId	string	非必須。デフォルトはマイク装置のID
roomParam.defaultSpeakerId	String	非必須。デフォルトはスピーカー装置のID

enterRoom

一般メンバーがルームに参加します。

```
TUIRoomRef.value.enterRoom(roomId, 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	number	ルームID
roomParam	Object	非必須
roomParam.isOpenCamera	string	非必須。入室はカメラをオンにするかどうか。デフォルトはオフです
roomParam.isOpenMicrophone	string	非必須。入室はマイクをオンにするかどうか。デフォルトはオフです
roomParam.defaultCameraId	string	非必須。デフォルトはカメラ装置のID
roomParam.defaultMicrophoneId	string	非必須。デフォルトはマイク装置のID
roomParam.defaultSpeakerId	String	非必須。デフォルトはスピーカー装置のID

©2013-2022 Tencent Cloud. All rights reserved.

TUIRoomイベント

onRoomCreate

ルーム作成のコールバック。

```
<template>
<room ref="TUIRoomRef" @on-room-create="handleRoomCreate"></room>
</template>
<script setup lang="ts">
// TUIRoomコンポーネントをインポートします。インポートパスが正しいことを確認してください
import Room from './TUIRoom/index.vue';

function handleRoomCreate(info) {
if (info.code === 0) {
console.log('ルーム作成成功')
}
}
</script>
```

onRoomEnter

入室コールバック。

```
<template>
<room ref="TUIRoomRef" @on-room-enter="handleRoomEnter"></room>
</template>
<script setup lang="ts">
// TUIRoomコンポーネントをインポートします。インポートパスが正しいことを確認してください
import Room from './TUIRoom/index.vue';

function handleRoomEnter(info) {
if (info.code === 0) {
console.log('入室成功')
}
}
</script>
```

onRoomDestory

キャスターがルームを廃棄する通知。

```
<template>
<room ref="TUIRoomRef" @on-room-destory="handleRoomDestory"></room>
</template>
<script setup lang="ts">
```

©2013-2022 Tencent Cloud. All rights reserved.

```
// TUIRoomコンポーネントをインポートします。インポートパスが正しいことを確認してください
import Room from './TUIRoom/index.vue';

function handleRoomDestory(info) {
if (info.code === 0) {
console.log('キャスターが廃棄することに成功しました')
}
}
</script>
```

onRoomExit

一般メンバーが退室する通知。

```
<template>
<room ref="TUIRoomRef" @on-room-exit="handleRoomExit"></room>
</template>
<script setup lang="ts">
// TUIRoomコンポーネントをインポートします。インポートパスが正しいことを確認してください
import Room from './TUIRoom/index.vue';

function handleRoomExit(info) {
if (info.code === 0) {
console.log('一般メンバーが退室することに成功しました')
}
}
</script>
```

onKickOff

一般メンバーがキャスターにルームからキックアウトされたときの通知です。

```
<template>
<room ref="TUIRoomRef" @on-kick-off="handleKickOff"></room>
</template>
<script setup lang="ts">
// TUIRoomコンポーネントをインポートします。インポートパスが正しいことを確認してください
import Room from './TUIRoom/index.vue';

function handleKickOff(info) {
if (info.code === 0) {
console.log('一般メンバーがキャスターにルームからキックアウト')
}
}
</script>
```

©2013-2022 Tencent Cloud. All rights reserved.

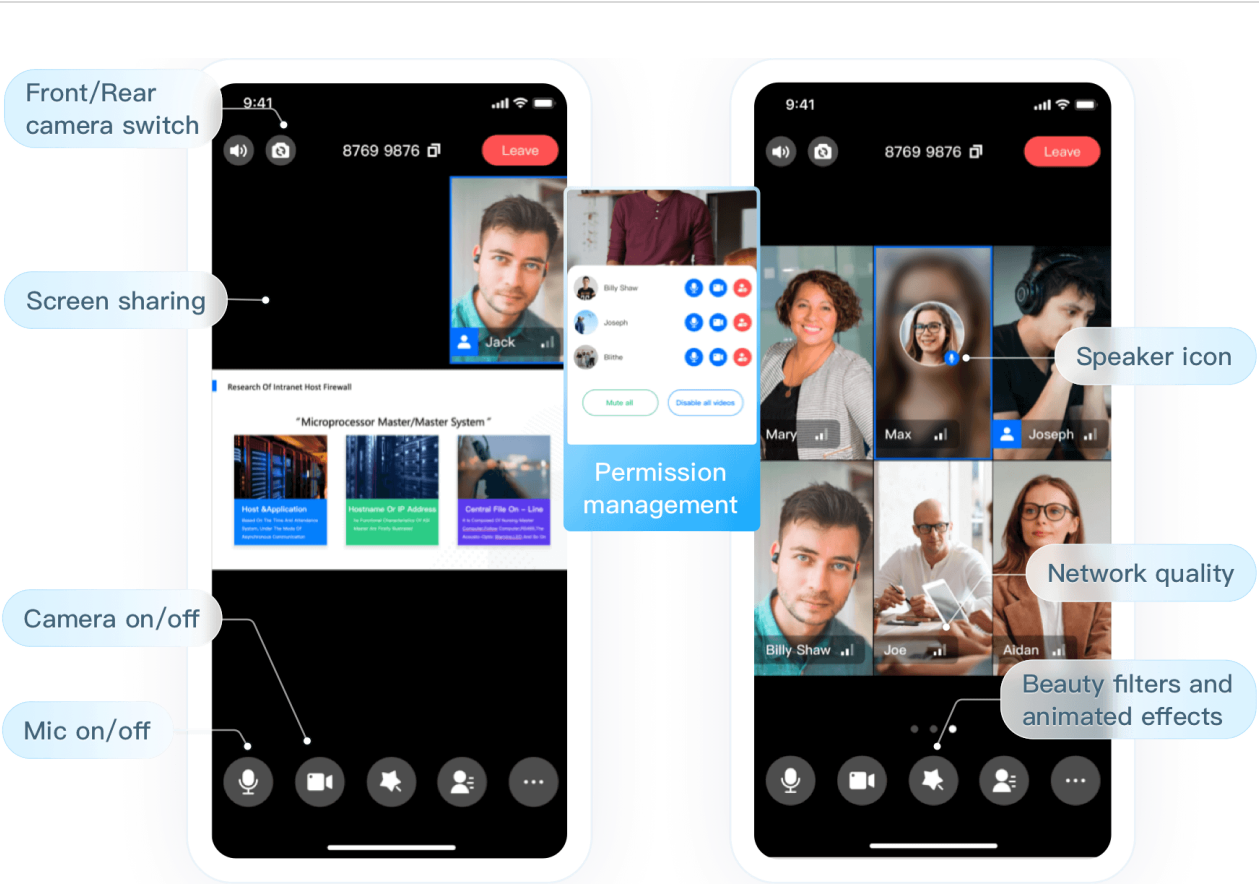
TUIRoom (Android)の統合

最終更新日：：2022-09-06 14:42:23

コンポーネントの説明

TUIRoomはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIRoomコンポーネントを統合することにより、数行のコードを書くだけで、Appに画面共有、美顔、低遅延ビデオ通話などを組み込むことができます。TUIRoomはまた、iOS、Windows、Macなどのプラットフォームでもサポートしています。基本機能は下図のとおりです：

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。



©2013-2022 Tencent Cloud. All rights reserved.

コンポーネントの統合

ステップ1：TUIRoomコンポーネントのダウンロードとインポート

クリックしてGithubに進み、コードのクローン/ダウンロードを選択した後、Android下のtuiroom、debug、tuibeautyディレクトリをプロジェクトにコピーし、次のようにインポート動作を完了します：

1. `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 ':tuiroom'
include ':debug'
include ':tuibeauty'
```

2. appの `build.gradle` ファイルにtuiroom、debug、tuibeautyに対する依存関係を追加します：

```
api project(':tuiroom')
api project(':debug')
api project(':tuibeauty')
```

3. ルートディレクトリの `build.gradle` ファイルに `TRTC SDK` および `IM SDK` の依存関係を追加します：

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

ステップ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システムではマイクの権限などを動的に申請してください）：

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

©2013-2022 Tencent Cloud. All rights reserved.

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：TUIコンポーネントリポジトリの作成と初期化

```
// 1.コンポーネントのログイン
TUILogin.addLoginListener(new TUILoginListener() {
@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（例：アカウントが他のデバイスでログインしている）
}
@Override
public void onUserSigExpired() { // userSig期限切れ通知
}
});
TUILogin.login(context, "あなたのSDKAppId", "あなたのuserId", "あなたのuserSig", null
);

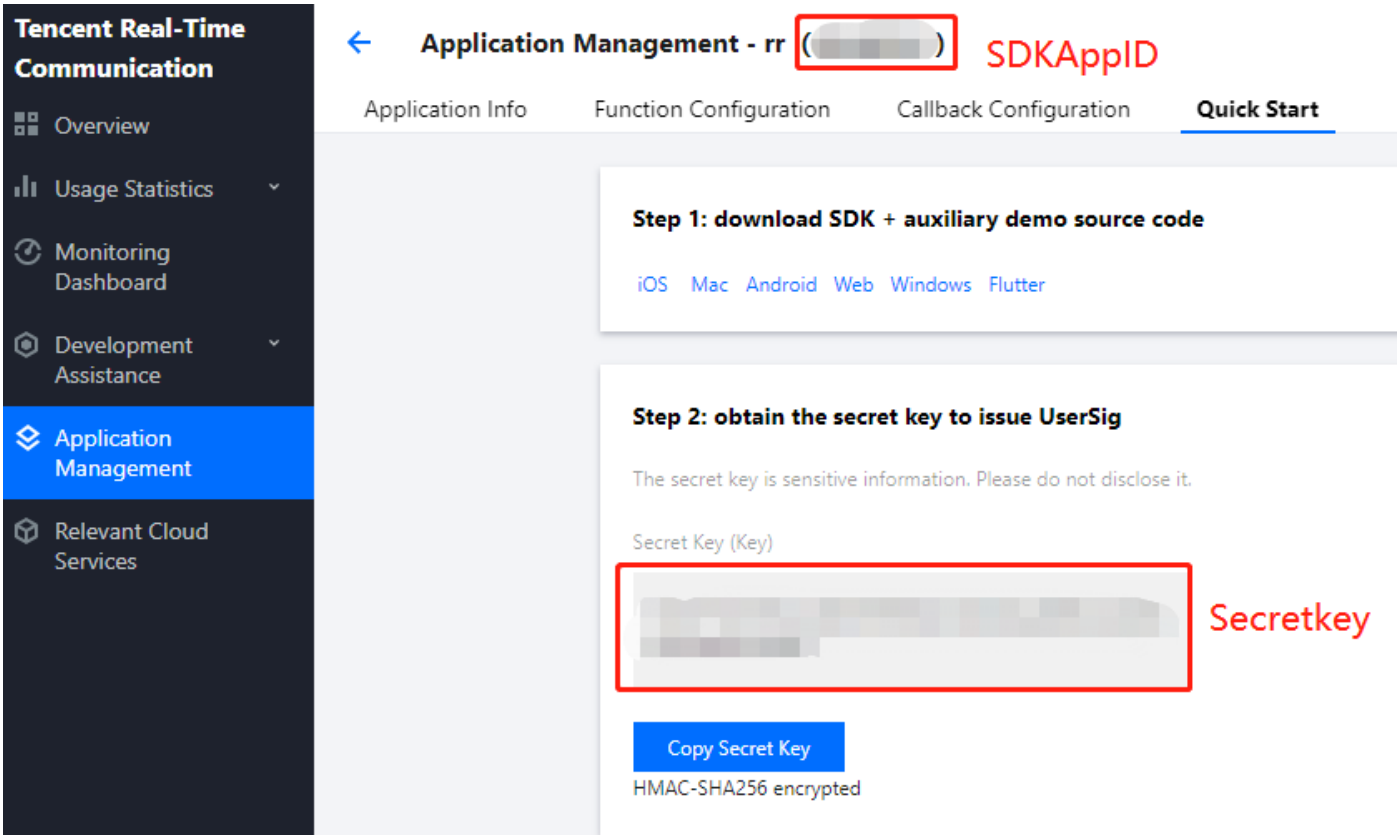
// 2.TUIRoomインスタンスの初期化
TUIRoom tuiRoom = TUIRoom.sharedInstance(this);
```

パラメータの説明

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション**

©2013-2022 Tencent Cloud. All rights reserved.

情報をクリックすると、SDKAppID情報が次の図のように表示されます：



- **Secretkey**：**TRTC アプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます。
- **userId**：現在のユーザーID。文字列タイプであり、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**：SDKAppId、userId，Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のuserSigがオンラインで直接生成されます。UserSigの計算、使用方法をご参照ください。

ステップ4：多人数オーディオビデオインタラクションの実装

1. **管理者が多人数オーディオビデオインタラクティブルームを作成できるようにします。**

```
tuiRoom.createRoom(12345, TUIRoomCoreDef.SpeechMode.FREE_SPEECH, true, true);
```

2. **他のメンバーがオーディオビデオルームに参加できるようにします。**

```
tuiRoom.enterRoom(12345, true, true);
```

©2013-2022 Tencent Cloud. All rights reserved.

ステップ5：ルーム管理（オプション）

1. 管理者によるルーム解散 TUIRoomCore#destroyRoom。

```
// 1.管理者が呼び出してルームを解散
mTUIRoomCore.destroyRoom(new TUIRoomCoreCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {

}
});
メンバー側は、ルームの解散を通知するonDestroyRoomコールバックメッセージを受信します
@Override
public void onDestroyRoom() {
//管理者が解散し、ルームから退出
}
```

2. メンバーの退室 TUIRoomCore#leaveRoom。

```
// 1.管理者以外による退室の呼び出し
mTUIRoomCore.leaveRoom(new TUIRoomCoreCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {

}
});
//メンバー側は、退室者があったことを通知するonRemoteUserLeaveコールバックメッセージを受信します
@Override
public void onRemoteUserLeave(String userId) {
Log.d(TAG, "onRemoteUserLeave userId: " + userId);
}
```

ステップ6：画面共有（オプション）

画面共有TUIRoomCore#startScreenCaptureを実装します。

```
// 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$TXScreenCaptureAssistantActi
```

©2013-2022 Tencent Cloud. All rights reserved.

```
vity"
android:theme="@android:style/Theme.Translucent" />
</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:" + getPackageName()));
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() {
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.TRTCScreenShareParam
s();
mTUIRoom.stopCameraPreview();
mTUIRoom.startScreenCapture(encParams, params);
}
```

よくあるご質問

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。

©2013-2022 Tencent Cloud. All rights reserved.

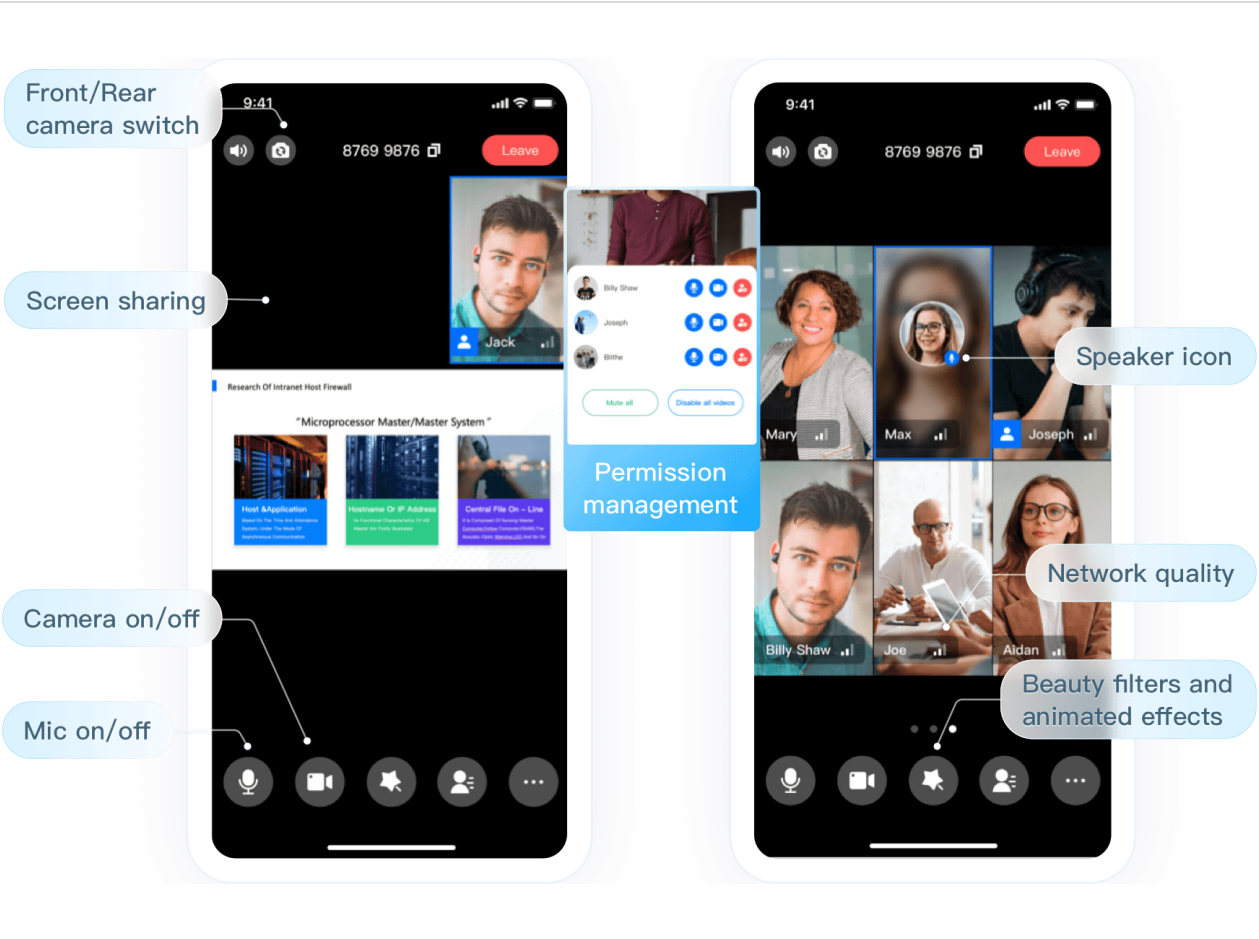
TUIRoom (iOS)の統合

最終更新日：：2022-07-19 16:28:38

コンポーネントの説明

TUIRoomはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIRoomコンポーネントを統合することにより、数行のコードを書くだけで、Appに画面共有、美顔、低遅延ビデオ通話などを組み込むことができます。TUIRoomはまた、Android、Windows、Macなどのプラットフォームでもサポートしています。基本機能は下図のとおりです：

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。



©2013-2022 Tencent Cloud. All rights reserved.

コンポーネントの統合

ステップ1：TUIRoom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. プロジェクトの `Podfile` ファイルと同じ階層のディレクトリ下に `TUIRoom` フォルダを作成します。
2. クリックして**Github/TUIRoom**に進み、コードのクローン/ダウンロードを選択した後、**TUIRoom/iOS/**ディレクトリ下の `Source` 、 `Resources` 、 `TUIBeauty` 、 `TXAppBasic`フォルダ、 TUIRoom.podspec `ファイルを、` ステップ1`で作成したTUIRoomフォルダ下にコピーします。
3. 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。

```
# :path => "TUIRoom.podspecを指定する相対パス"
pod 'TUIRoom', :path => "./TUIRoom/TUIRoom.podspec", :subspecs => ["TRTC"]
# :path => "TXAppBasic.podspecを指定する相対パス"
pod 'TXAppBasic', :path => "./TUIRoom/TXAppBasic/"
# :path => "TUIBeauty.podspecを指定する相対パス"
pod 'TUIBeauty', :path => "./TUIRoom/TUIBeauty/"
```

注意：
`Source` 、 `Resources` フォルダと `TUIRoom.podspec` ファイルは同一のディレクトリ下にある必要があります。

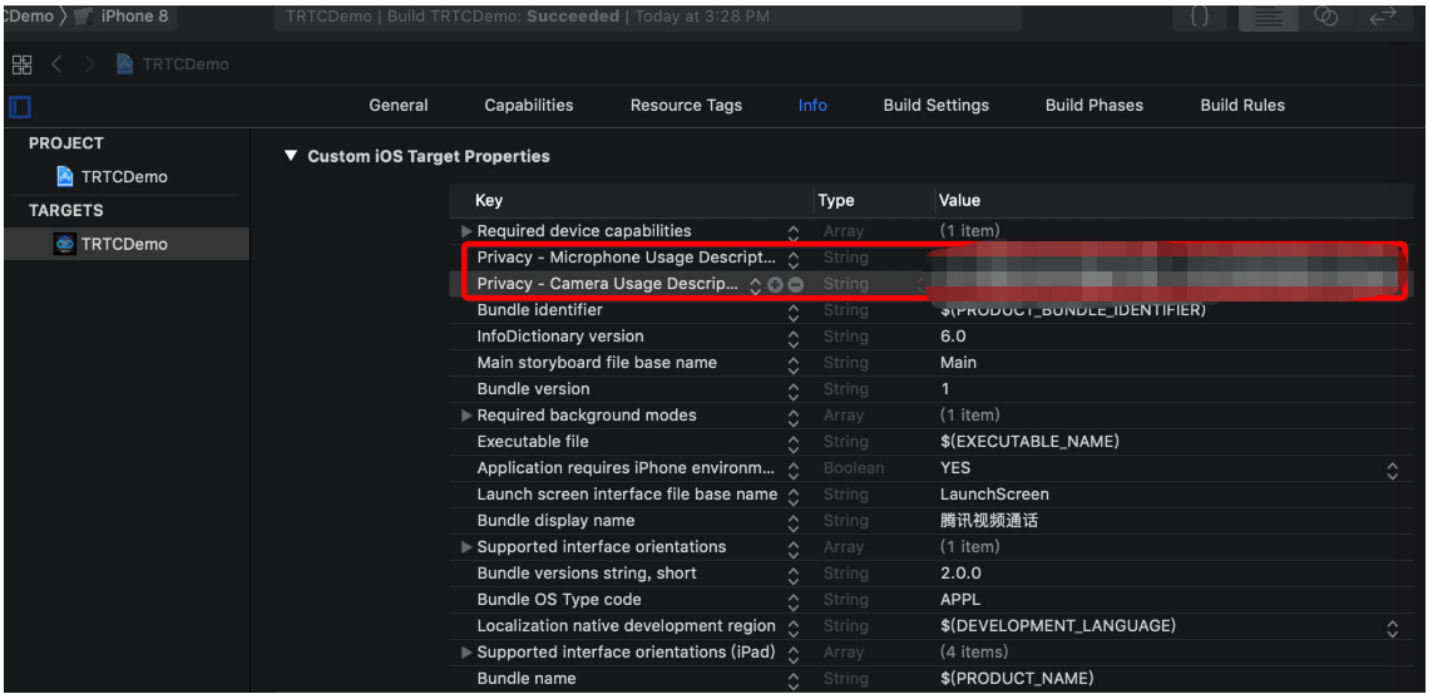
- TXAppBasic.podspecはTXAppBasicフォルダ下にあります。
- TUIBeauty.podspecはTCBeautyKitフォルダ下にあります。

ステップ2：権限の設定

オーディオビデオ機能を使用するには、マイクおよびカメラの使用権限を付与する必要があります。AppのInfo.plistで以下の2項を追加します。これらはシステムが権限付与ダイアログボックスをポップアップする際に表示される、マイクとカメラのメッセージにそれぞれ対応します。

```
<key>NSCameraUsageDescription</key>
<string>RoomAppはカメラへのアクセス権限が必要です。有効にしないとレコーディングしたビデオの画面は出ません</string>
<key>NSMicrophoneUsageDescription</key>
<string>RoomAppはマイクへのアクセス権限が必要です。有効にないと、レコーディングしたビデオの音声は出ません</string>
```

©2013-2022 Tencent Cloud. All rights reserved.



ステップ3：TUIコンポーネントリポジトリの作成と初期化

```
@import TUIRoom;
@import TUICore;

// 1.コンポーネントのログイン
[TUILogin login:@"あなたのSDKAppID" userID:@あなたのUserID" userSig:@"あなたのUserSig" succ:^{

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

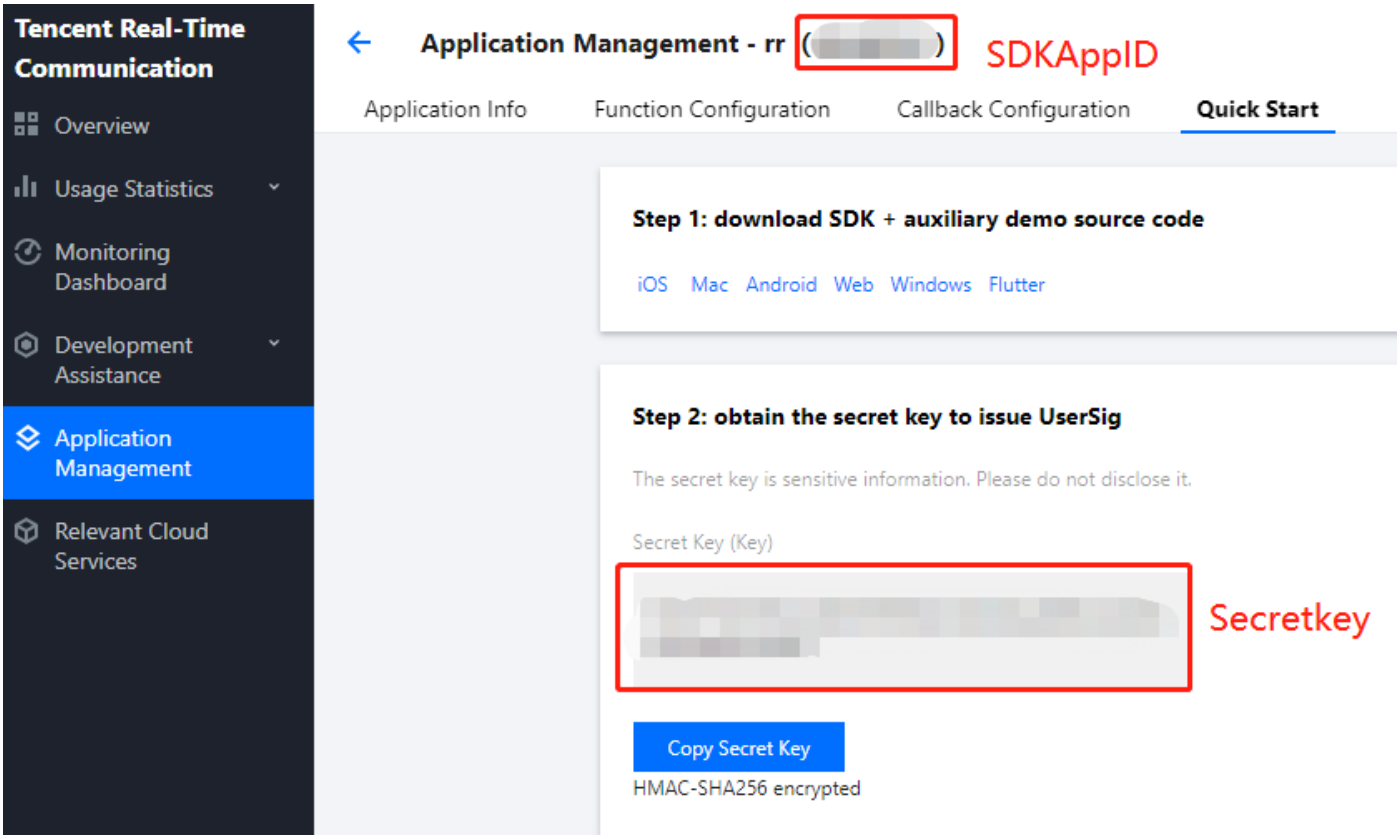
}];
// 2.TUIRoomインスタンスの初期化
TUIRoom *tuiRoom = [TUIRoom sharedInstance];
<dx-code-holder data-codeindex="2"></dx-code-holder>
```

パラメータの説明：

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション**

©2013-2022 Tencent Cloud. All rights reserved.

情報をクリックすると、SDKAppID情報が次の図のように表示されます：



- **SecretKey**：**TRTC アプリケーションキー**であり、SDKAppIDに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上図のように表示されます。
- **UserID**：現在のユーザーのIDです。文字列タイプであり、長さは32バイト以内とし、特殊文字の使用はサポートしていません。英語または数字の使用をお勧めします。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ができます。
- **UserSig**：SDKAppId、UserID，SecretKeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のUserSigがオンラインで直接生成されます。また当社のTUIRoomデモプロジェクトを参照してご自身で計算することもできます。その他の情報については、UserSigの計算、使用方法をご参照ください。

ステップ4：多人数オーディオビデオインタラクションの実装

1. **管理者が多人数オーディオビデオインタラクティブルームを作成できるようにします。**
   - Objective-C ObjectiveC
   - Swift Swift

```
@import TUIRoom;

[tuiRoom createRoomWithRoomId:12345 speechMode:TUIRoomFreeSpeech isOpenCamera:YES isOpenMicrophone:YES];
```

©2013-2022 Tencent Cloud. All rights reserved.

ステップ5：ルーム管理（オプション）

1. 管理者によるルーム解散 TUIRoomCore#destroyRoom。

```
@import TUIRoom;

[[TUIRoomCore shareInstance] destroyRoom:^(NSInteger code, NSString * _Nonnull
message) {

}];
<dx-code-holder data-codeindex="4"></dx-code-holder>
```

2. メンバーの退室 TUIRoomCore#leaveRoom。

```
@import TUIRoom;

[[TUIRoomCore shareInstance] leaveRoom:^(NSInteger code, NSString * _Nonnull m
essage) {

}];
<dx-code-holder data-codeindex="5"></dx-code-holder>
```

ステップ6：画面共有（オプション）

画面共有TUIRoomCore#startScreenCaptureを実装します。画面共有プロジェクトの設定については、リアルタイム画面共有(iOS)をご参照ください。

```
@import TUIRoom;
@import TXLiteAVSDK_Professional;

TRTCVideoEncParam *params = [[TRTCVideoEncParam alloc] init];
params.videoResolution = TRTCVideoResolution_1280_720;
params.resMode = TRTCVideoResolutionModePortrait;
params.videoFps = 10;
params.enableAdjustRes = NO;
params.videoBitrate = 1500;

[[TUIRoomCore shareInstance] startScreenCapture:param];
<dx-code-holder data-codeindex="6"></dx-code-holder>
```

よくあるご質問

©2013-2022 Tencent Cloud. All rights reserved.

CocoaPodsをインストールするにはどうすればよいですか？

端末のウィンドウに次のコマンドを入力します（事前にMac にRuby環境をインストールしてください）。

```
:::
::: Swift Swift
import TUIRoom
import TUICore
// 1.コンポーネントのログイン
TUILogin.login("あなたのSDKAppID", userID: "あなたのUserID", userSig: "あなたのUserSi
g") {

} fail: { code, msg in

}
// 2.TUIRoomインスタンスの初期化
let tuiRoom = TUIRoom.sharedInstance
```

説明：
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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIRoom (Flutter)の統合

最終更新日：：2023-02-17 16:22:19

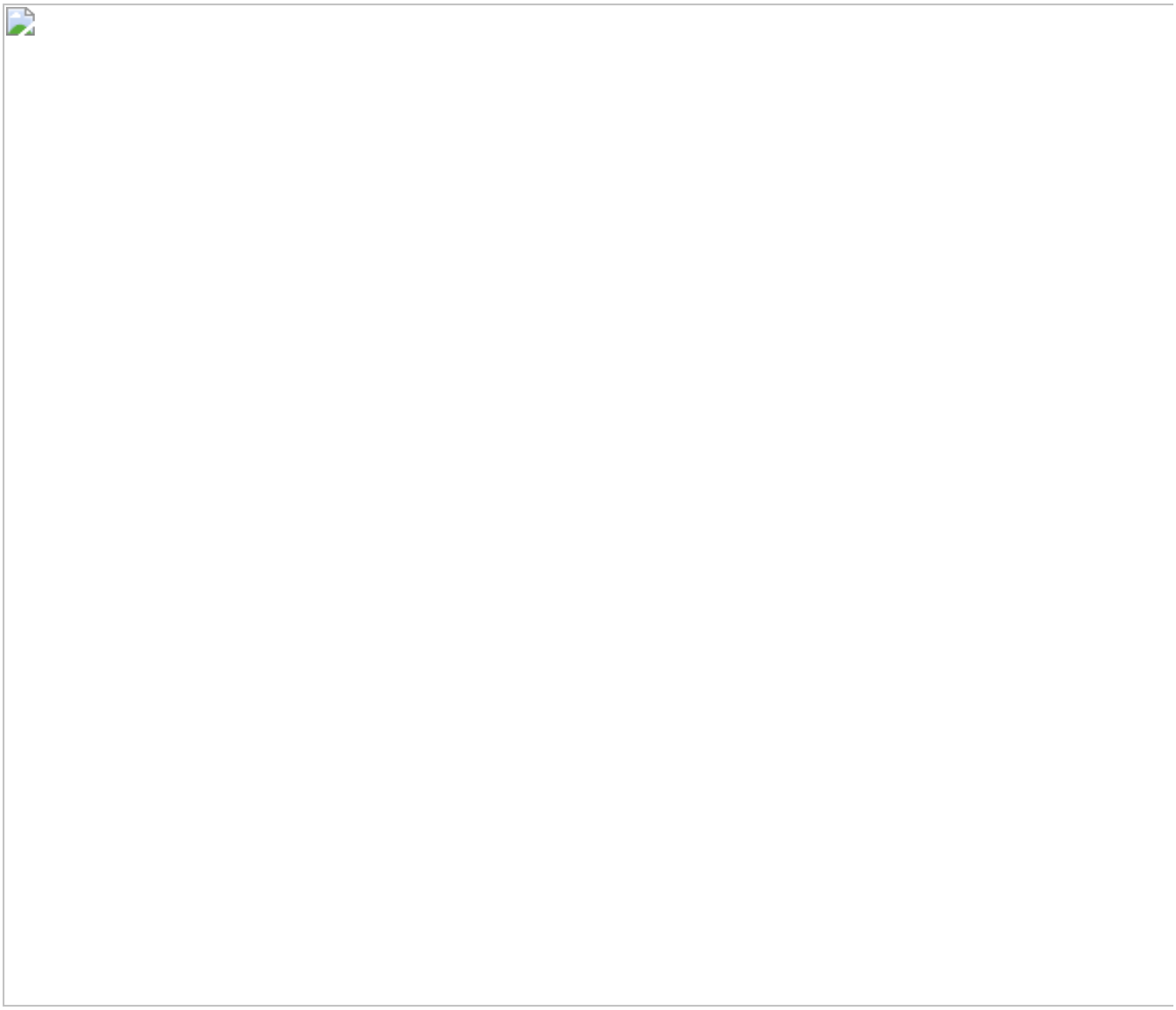
当社のAppをダウンロードしてインストールし、多人数オーディオビデオルームの効果をご体験いただけます。これには、画面共有、美顔、低遅延ミーティングなど、TRTCの多人数オーディオビデオルームシーン関連の機能が含まれています。
説明：
TUIKitシリーズコンポーネントはTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMサービスの課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

AppのUIの再利用

ステップ1：新規アプリケーションの作成

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。

©2013-2022 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。

ステップ2：Appソースコードのダウンロード

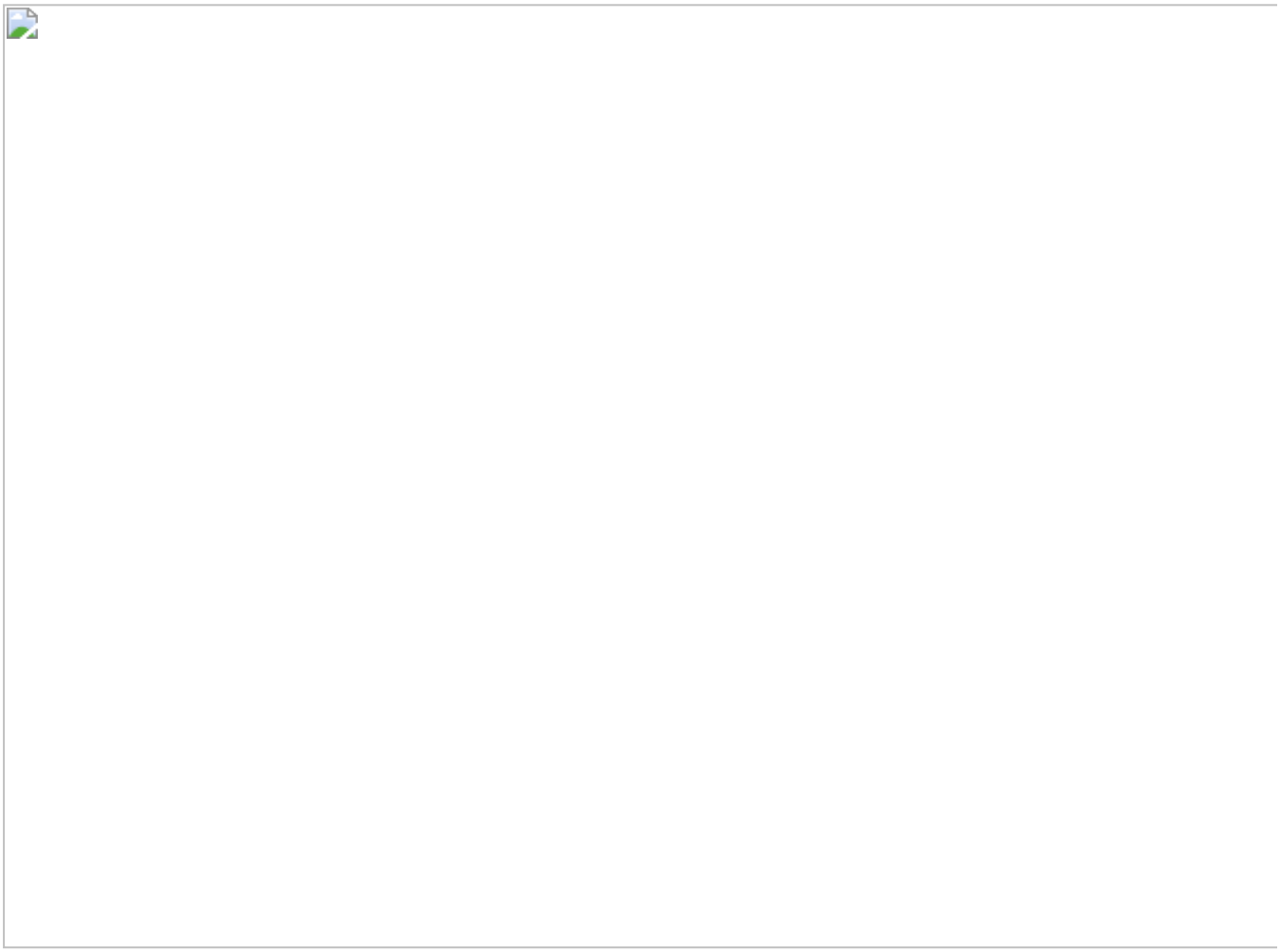
TRTCFlutterScenesDemoクリックして当該ページに入り、ソースコードをCloneまたはダウンロードします。

ステップ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。

©2013-2022 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: A placeholder by default. Set it to the actual `SDKAppID`.
SECRETKEY: A placeholder by default. Set it to the actual secret key.



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向けのインターフェースを提供します。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。詳細はサーバーでUserSigを生成するをご参照ください。

©2013-2022 Tencent Cloud. All rights reserved.

ステップ4：コンパイルと実行

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
1. 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。
2. コンパイルを行い、Demoプロジェクトを実行します。
1. `flutter run` を実行します。
2. 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。

ステップ5：Demoソースコードの修正

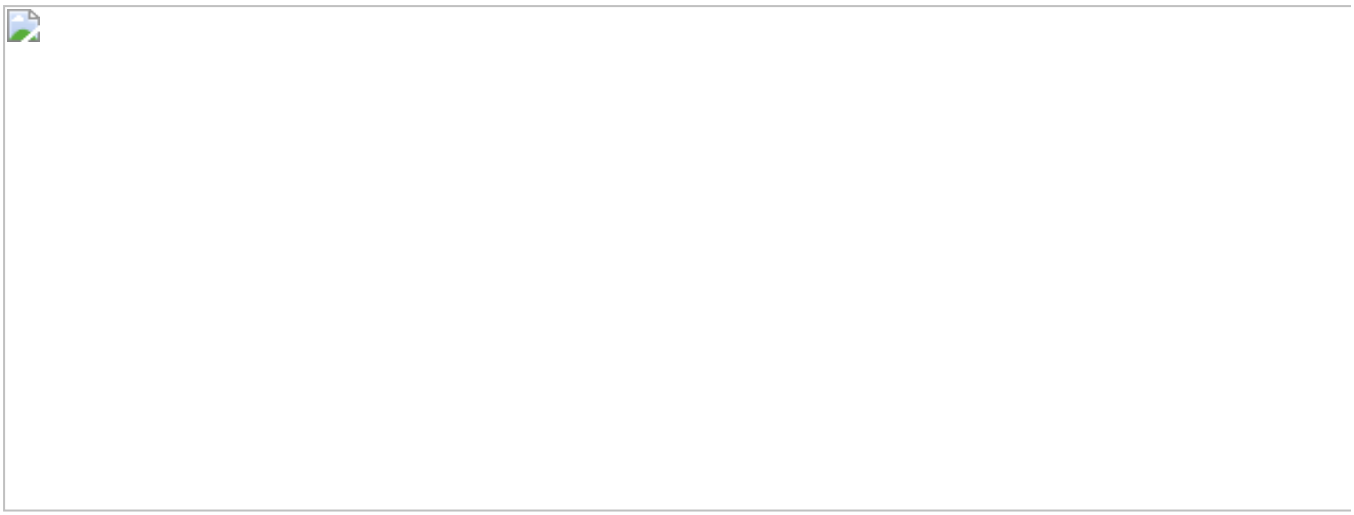
ソースコードのTRTCMeetingDemoフォルダには、uiとmodelの2つのサブフォルダが含まれ、uiフォルダの中はいずれもインターフェースコードです。以下の表にはファイルまたはフォルダおよび対応するUIをリストアップし、二次調整に便利です：

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

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。

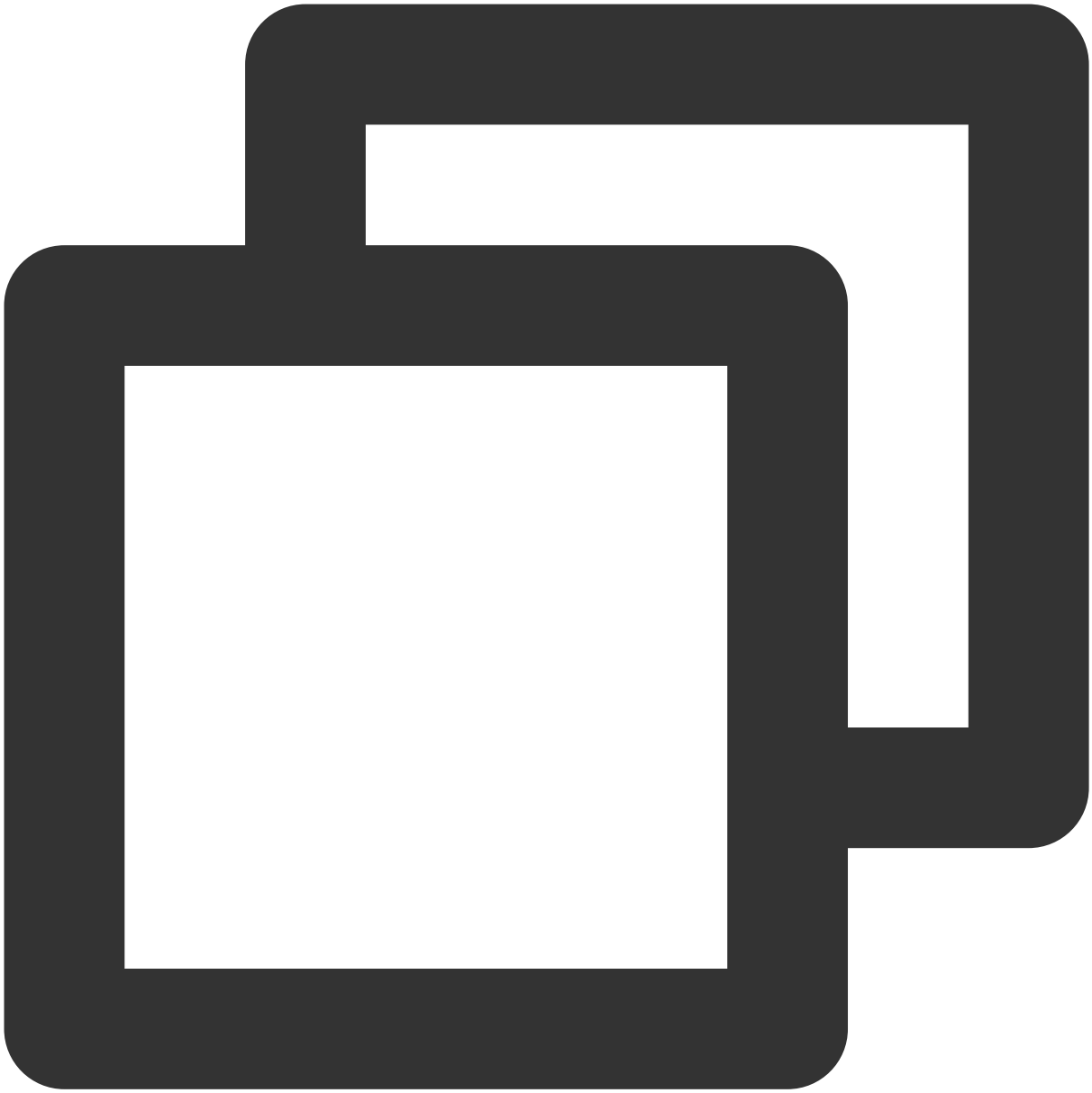
©2013-2022 Tencent Cloud. All rights reserved.



ステップ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：

©2013-2022 Tencent Cloud. All rights reserved.



```
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
```

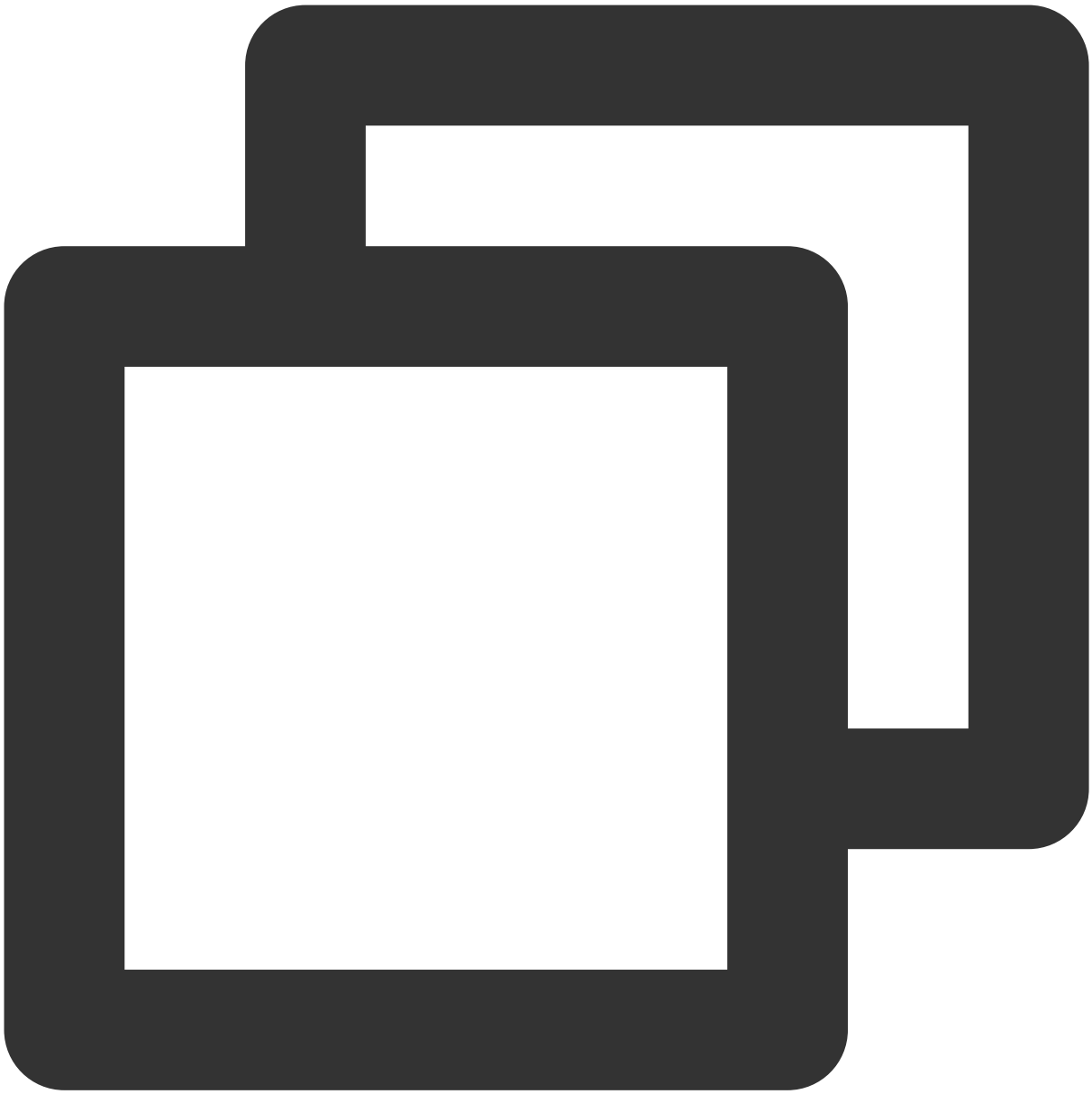
ステップ2：権限の設定および難読化ルール

iOS 端末

Android 端末

©2013-2022 Tencent Cloud. All rights reserved.

`Info.plist` にカメラとマイクの権限申請を追加してください：



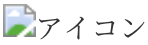
```
<key>NSMicrophoneUsageDescription</key>
<string>通常の音声通話が行えるようにマイクの権限を承認します</string>
```

1. `/android/app/src/main/AndroidManifest.xml` ファイルを開きます。
2. `xmlns:tools="http://schemas.android.com/tools"` をmanifestの中に追加します。
3. `tools:replace="android:label"をapplicationの中に追加します。

explain

この手順を実行しないと、Android Manifest merge failedコンパイルの失敗という問題が発生します。

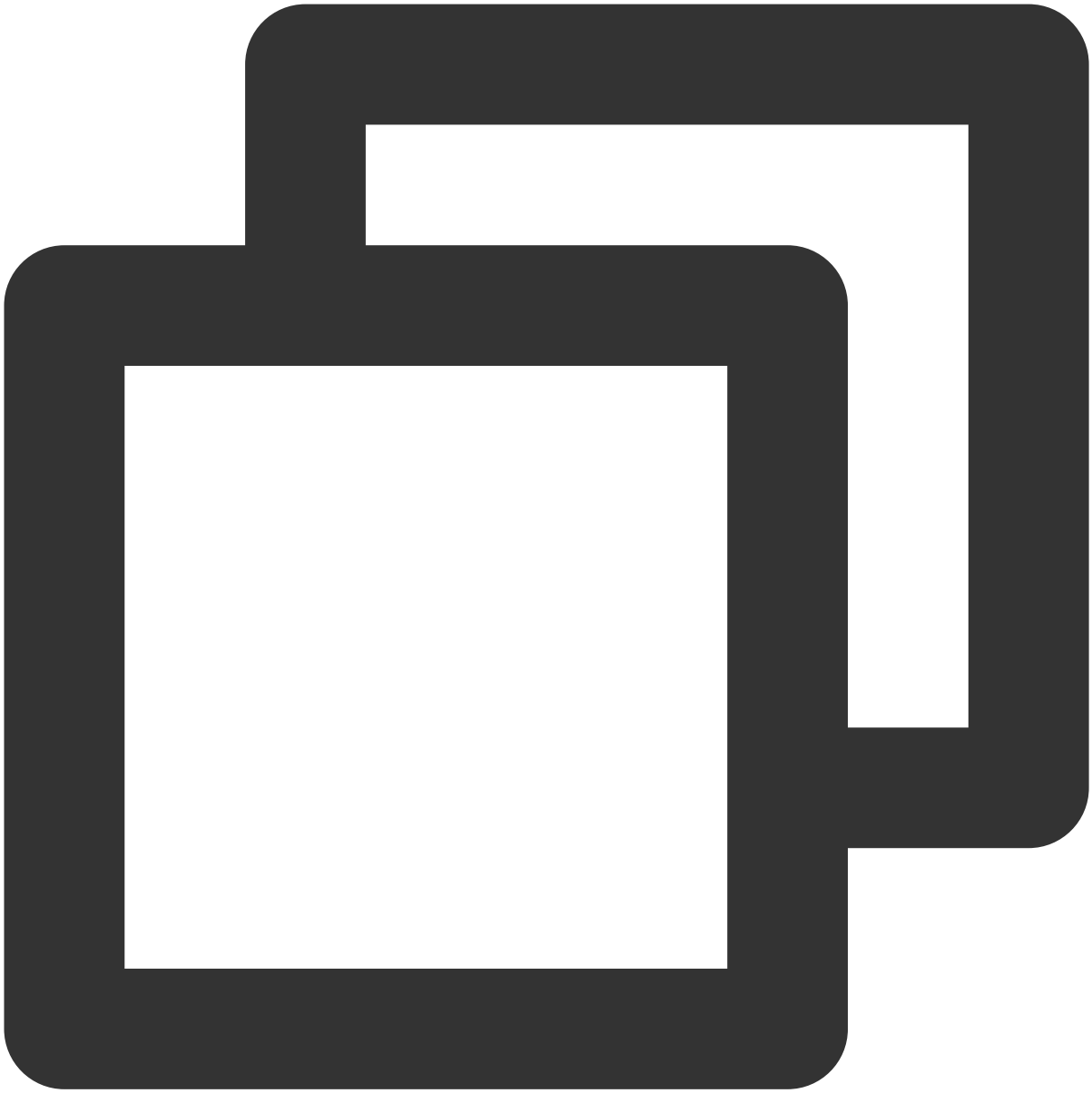
©2013-2022 Tencent Cloud. All rights reserved.



ステップ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：

©2013-2022 Tencent Cloud. All rights reserved.



```
lib/TRTCMeetingDemo/model/
```

ステップ4：コンポーネントの作成およびログイン

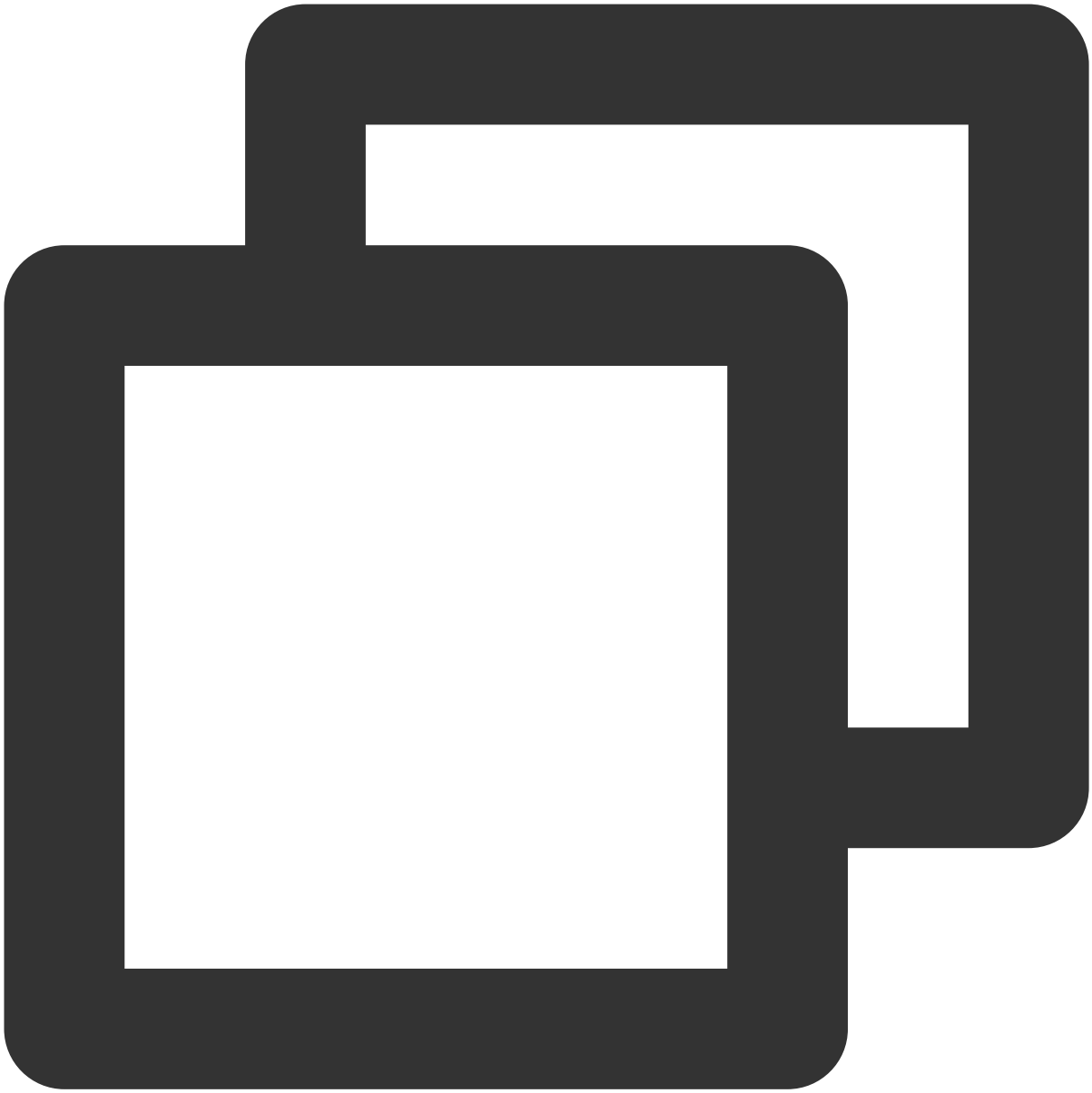
1. `sharedInstance` インターフェースを呼び出すと、TRTCMeetingコンポーネントのインスタンスオブジェクトを作成できます。
2. `registerListener` 関数を呼び出して、コンポーネントのイベント通知を登録します。

©2013-2022 Tencent Cloud. All rights reserved.

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。文字列タイプであり、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。

©2013-2022 Tencent Cloud. All rights reserved.



```
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
}
```

©2013-2022 Tencent Cloud. All rights reserved.

ステップ5：多人数ミーティングの新規作成

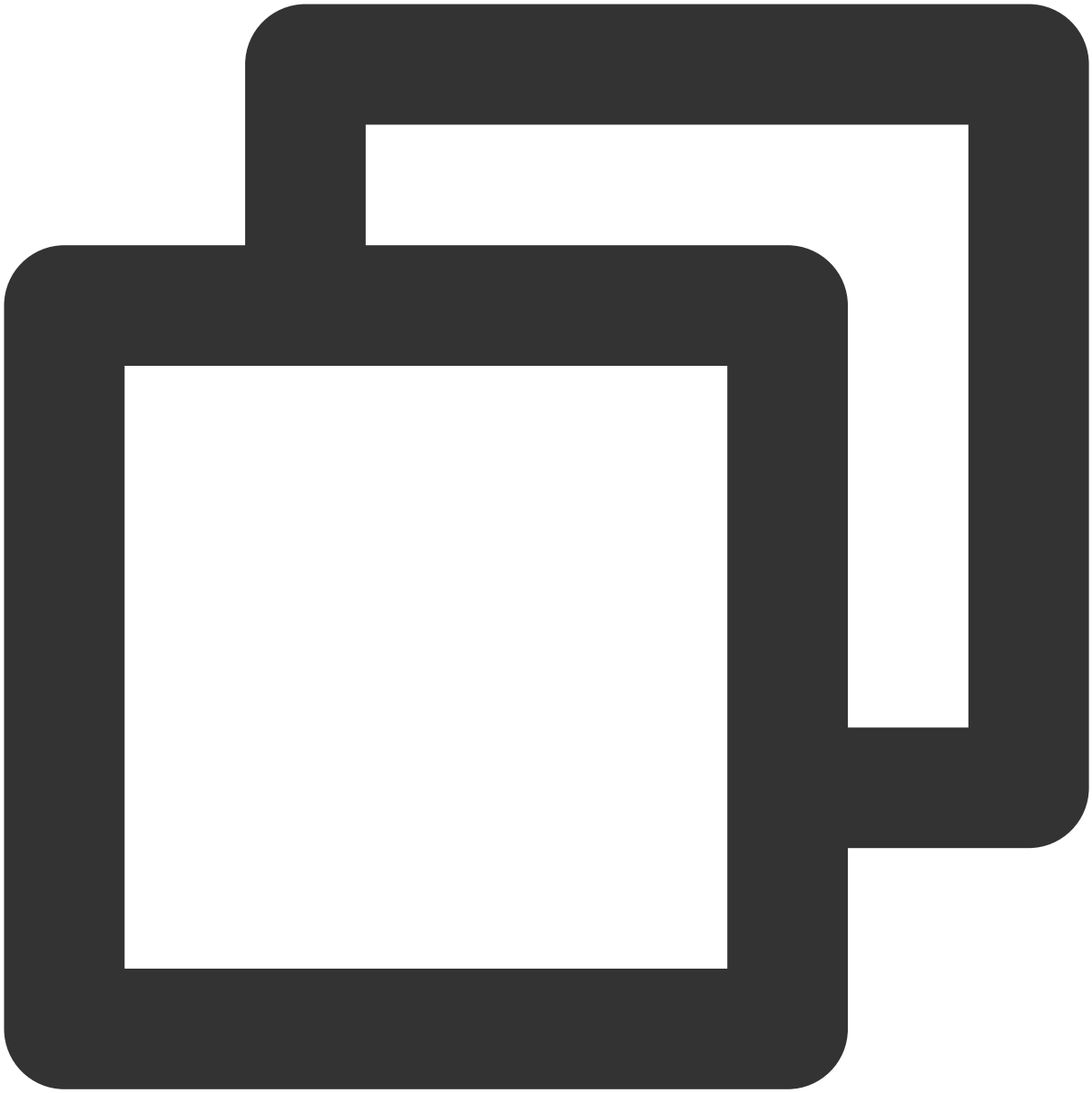
1. キャスターは、手順4でログインした後、 `setSelfProfile` を呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. キャスターがcreateMeetingを呼び出し、新しいミーティングルームを作成します。
3. キャスターは、 `startCameraPreview` を呼び出してビデオ画面をキャプチャすることができ、 `startMicrophone` を呼び出して音声をキャプチャすることもできます。
4. キャスターに美顔のニーズがある場合は、インターフェース上に美顔調節ボタンを設定して呼び出すことができ、getBeautyManager を介して美顔設定を行います。

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。

©2013-2022 Tencent Cloud. All rights reserved.



©2013-2022 Tencent Cloud. All rights reserved.



```
// 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
    trtcMeeting.startCameraPreview(true, TRTCCloudVideoViewId);
    trtcMeeting.startMicrophone();
    // 4. 美顔設定
```

©2013-2022 Tencent Cloud. All rights reserved.

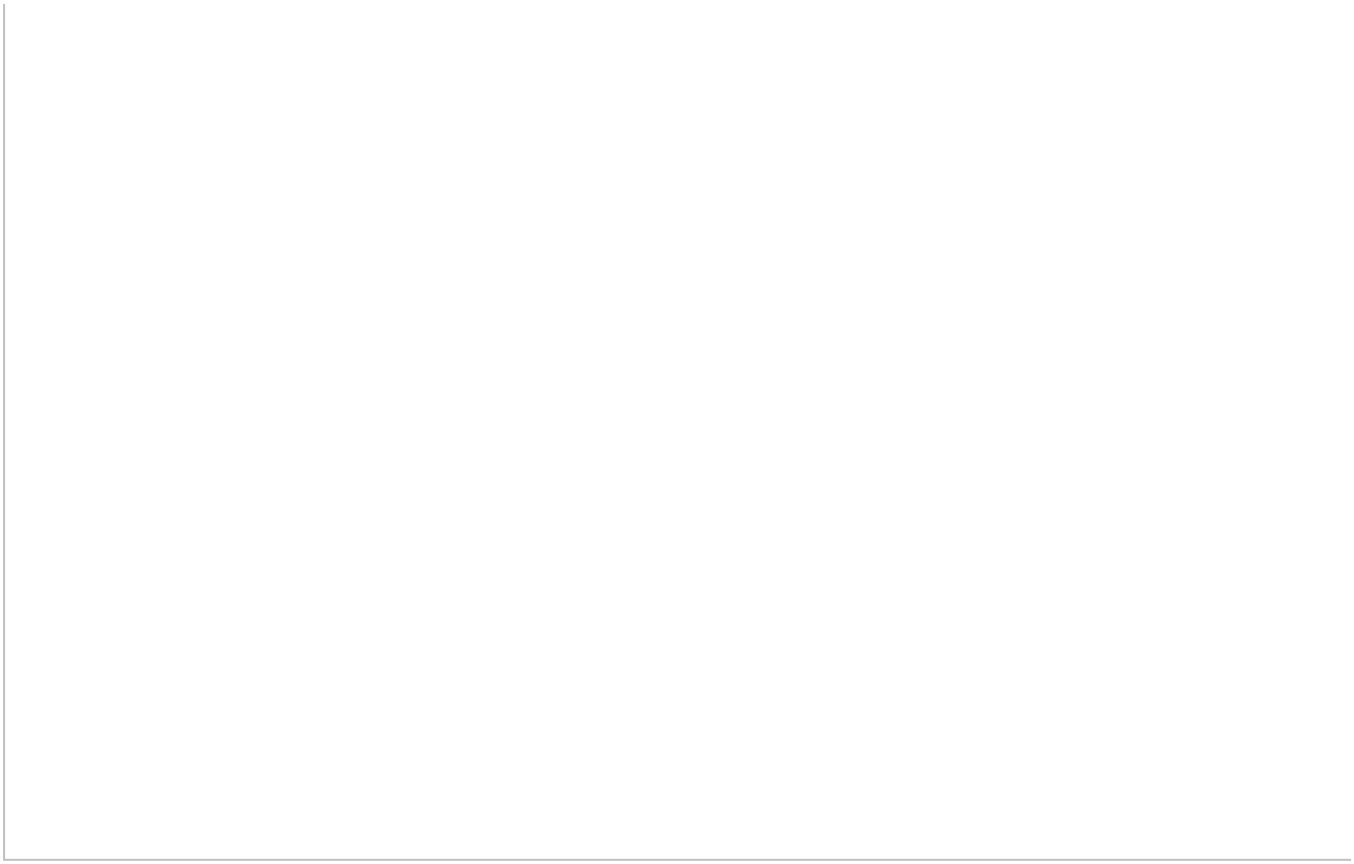
```
    trtcMeeting.getBeautyManager().setBeautyStyle(TRTCCloudDef.TRTC_BEAUTY_STYLE_PI
    trtcMeeting.getBeautyManager().setBeautyLevel(6);
}
```

ステップ6：参加者の多人数ミーティングへの参加

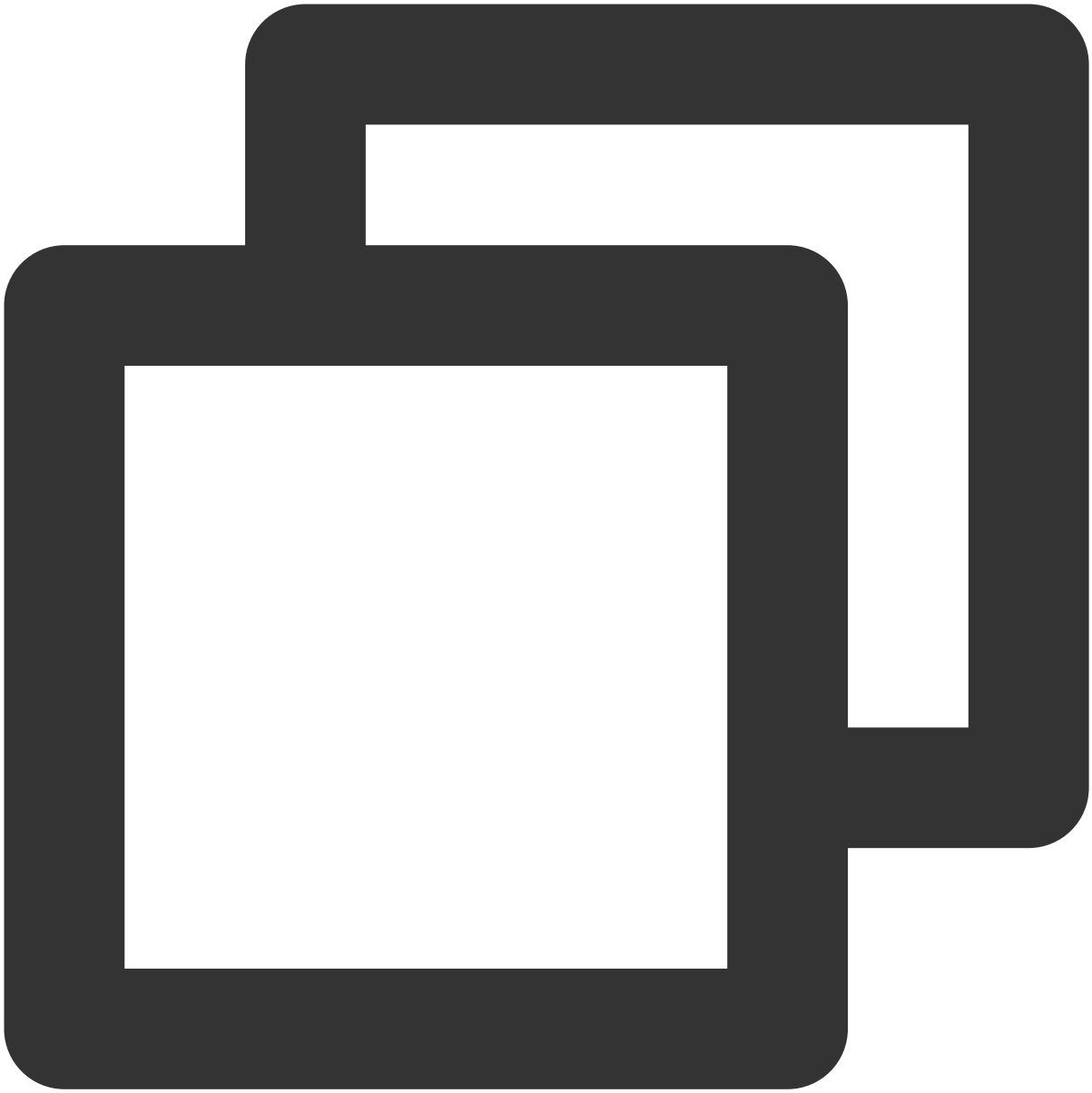
1. 参加者は、手順4でログインした後、setSelfProfileを呼び出して自身のニックネームおよびプロフィール画像を設定することができます。
2. 参加者は、enterMeeting呼び出してミーティングルーム番号を渡すと、ミーティングルームに参加できます。
3. 参加者は、startCameraPreviewを呼び出してビデオ画面をキャプチャすることができ、またstartMicrophoneを呼び出して音声をキャプチャすることもできます。
4. 他の参加者がカメラを起動すると、onUserVideoAvailableのイベントを受け取ります。この時、startRemoteViewを呼び出してuserIdを渡すと再生を開始できます。



©2013-2022 Tencent Cloud. All rights reserved.



©2013-2022 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
    trtcMeeting.startCameraPreview(true, TRTCCloudVideoViewId);
    trtcMeeting.startMicrophone();
    // 4. 美顔設定
```

©2013-2022 Tencent Cloud. All rights reserved.

```
    trtcMeeting.getBeautyManager().setBeautyStyle(TRTCCloudDef.TRTC_BEAUTY_STYLE_PI
    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;
    }
}
```

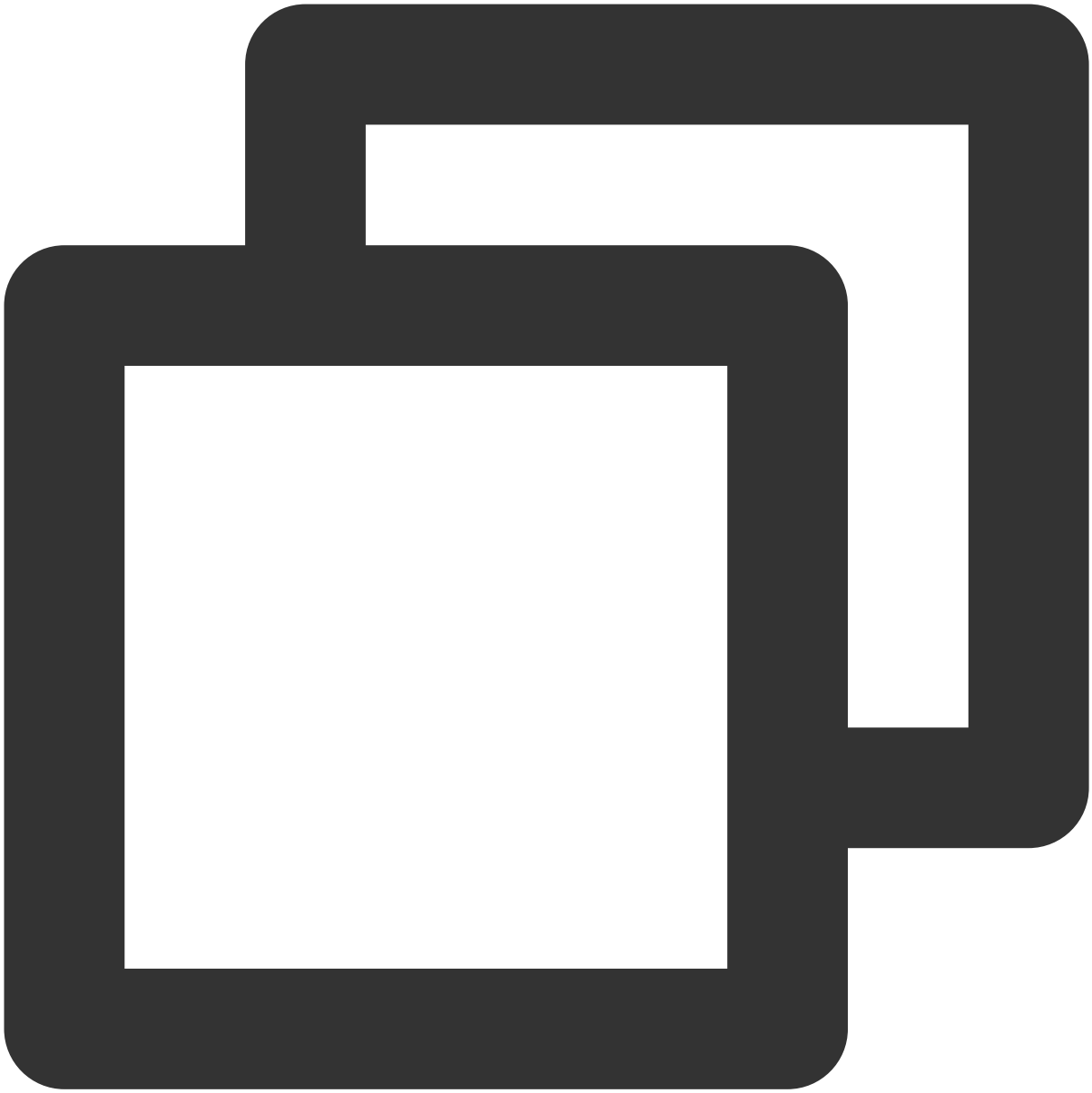
ステップ7：画面共有

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してください。
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。

©2013-2022 Tencent Cloud. All rights reserved.

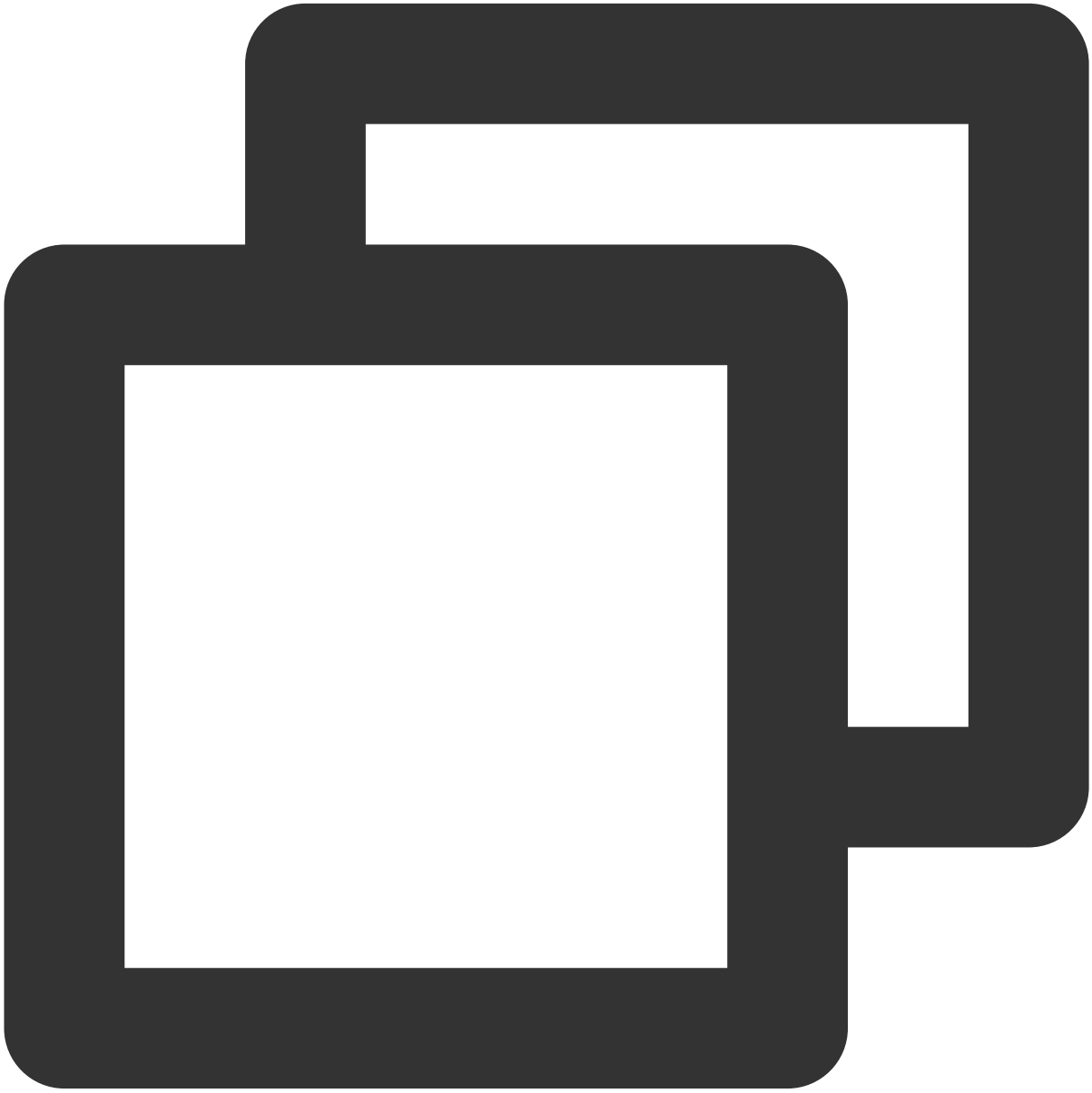


```
await trtcMeeting.stopCameraPreview();
trtcMeeting.startScreenCapture(
    videoFps: 10,
    videoBitrate: 1600,
    videoResolution: TRTCCloudDef.TRTC_VIDEO_RESOLUTION_1280_720,
    videoResolutionMode: TRTCCloudDef.TRTC_VIDEO_RESOLUTION_MODE_PORTRAIT,
    appGroup: iosAppGroup,
);
```

©2013-2022 Tencent Cloud. All rights reserved.

ステップ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。



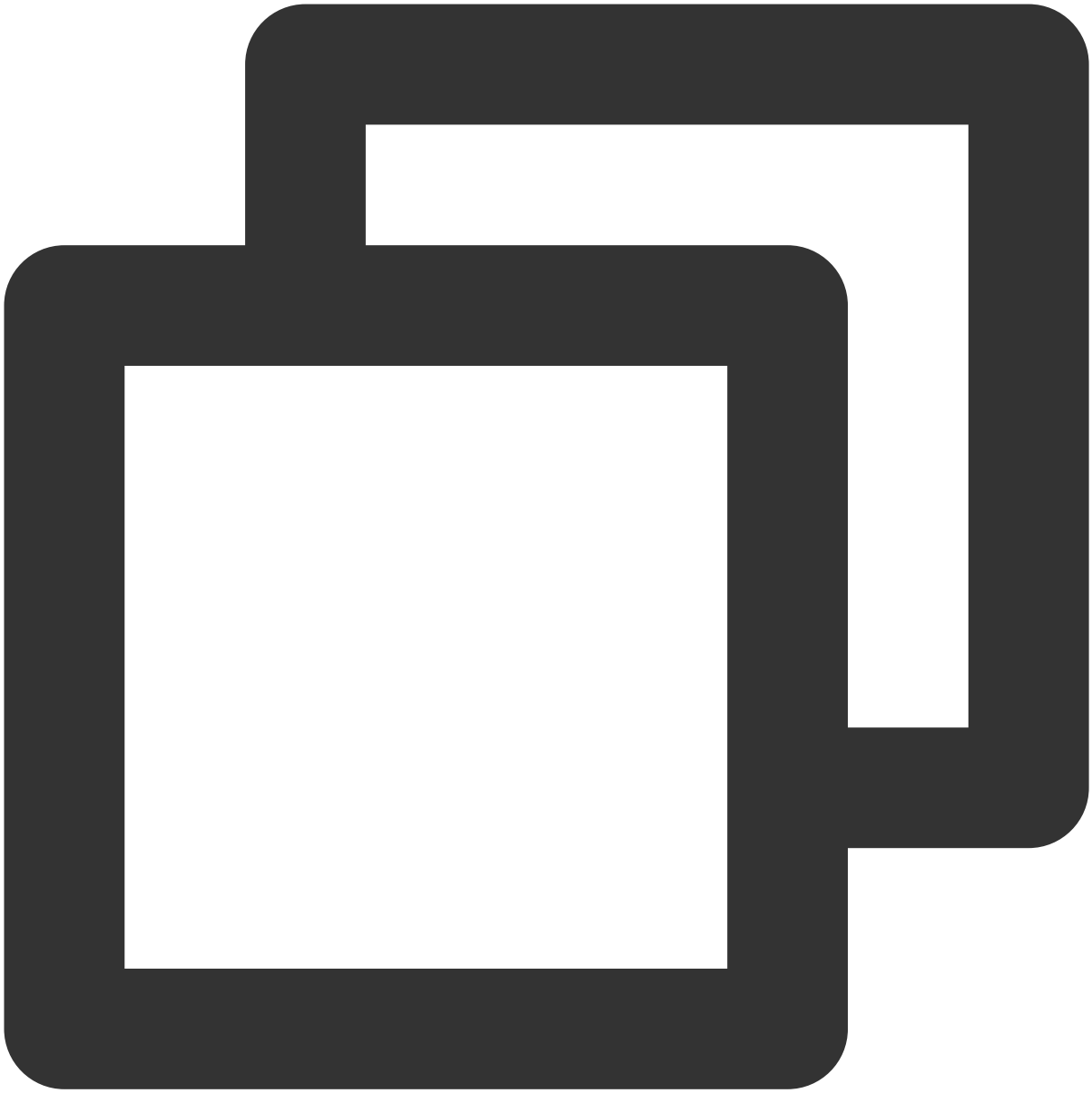
```
// 発信側：テキストメッセージの発信
trtcMeeting.sendRoomTextMsg('Hello Word!');
// 受信側：テキストメッセージのモニタリング
trtcMeeting.registerListener(onListener);
```

©2013-2022 Tencent Cloud. All rights reserved.

```
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（シグナリング）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）。

©2013-2022 Tencent Cloud. All rights reserved.



```
// 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
                print(param['userName'] +'から' + 'のミュートの通知：' + param['message
```

©2013-2022 Tencent Cloud. All rights reserved.

```
                trtcMeeting.muteLocalAudio(message == '1');
            }
            break;
        }
    }
```

©2013-2022 Tencent Cloud. All rights reserved.

TUIRoom (Windows&Mac)の統合

最終更新日：：2022-07-19 16:21:31

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、デジタル化のモデルチェンジを加速し、競争力を向上させます。
WindowsまたはMacプラットフォームのAppをダウンロード、インストールして体験できます。

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

デモンストレーション



ソリューションの優位性

©2013-2022 Tencent Cloud. All rights reserved.

- 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。

統合ガイド

多人数オーディオビデオルーム機能をスピーディーに統合するため、ここでは2種類の統合方法をお勧めしています。ご自身に合った方法を選択して二次開発にお役立てください。

- 外部プロセス起動
- カスタムUIの実装

環境の準備

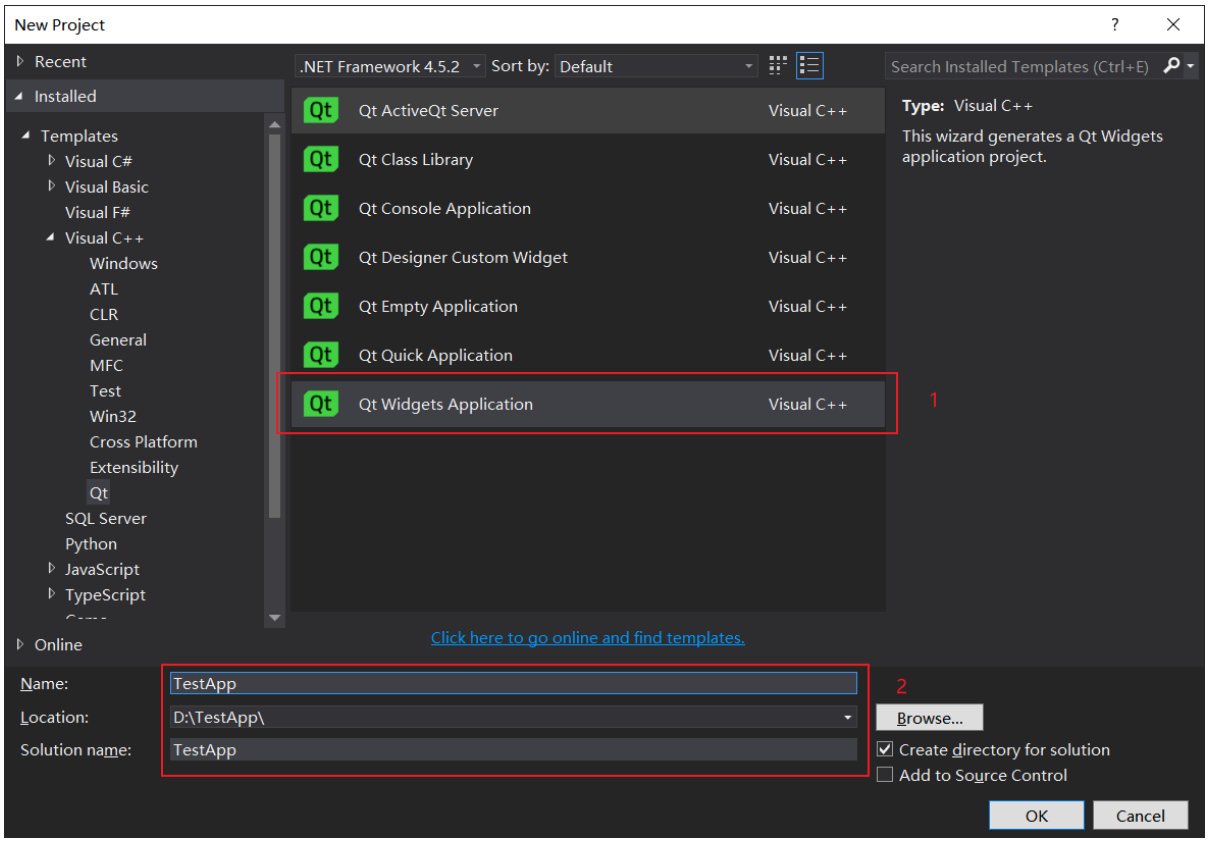
- **Windows環境** ：
  - Visual Studio 2015以上の統合開発環境。
  - QT5.9.1以上のバージョンのQT開発データベース。
  - VS下のQT開発プラグインQt Visual Studio Tools 2.2.0以上。
  - サポートする最低システム要件：Windows 8。
  - 正常な開発が行える統合開発環境であることを確実にしてください。
- **Mac環境**：
  - QT5.9.1およびそれ以上のバージョンのQT開発データベース。
  - QtCreator統合開発環境。 QTのインストール時に、QtCreatorの同時インストールを選択してください。バージョンはQTの公式インストールパッケージに従います。
  - QtCreator統合開発環境で正常な開発が行えることを確実にしてください。

外部プロセス起動

1. **RoomAppプログラムのコンパイル**
   - 外部プロセスによるRoomApp起動方式を使用する場合は、元のRoomAppの実行プログラムに依存するため、事前のコンパイルが必要です。
   - クリックしてRoomAppに進み、ソースコードをCloneし、プロジェクトを設定し、RoomAppをコンパイルして生成できます。
2. **TestAppプロジェクトの新規作成**
   - Windowsプラットフォーム
   - Macプラットフォーム

©2013-2022 Tencent Cloud. All rights reserved.

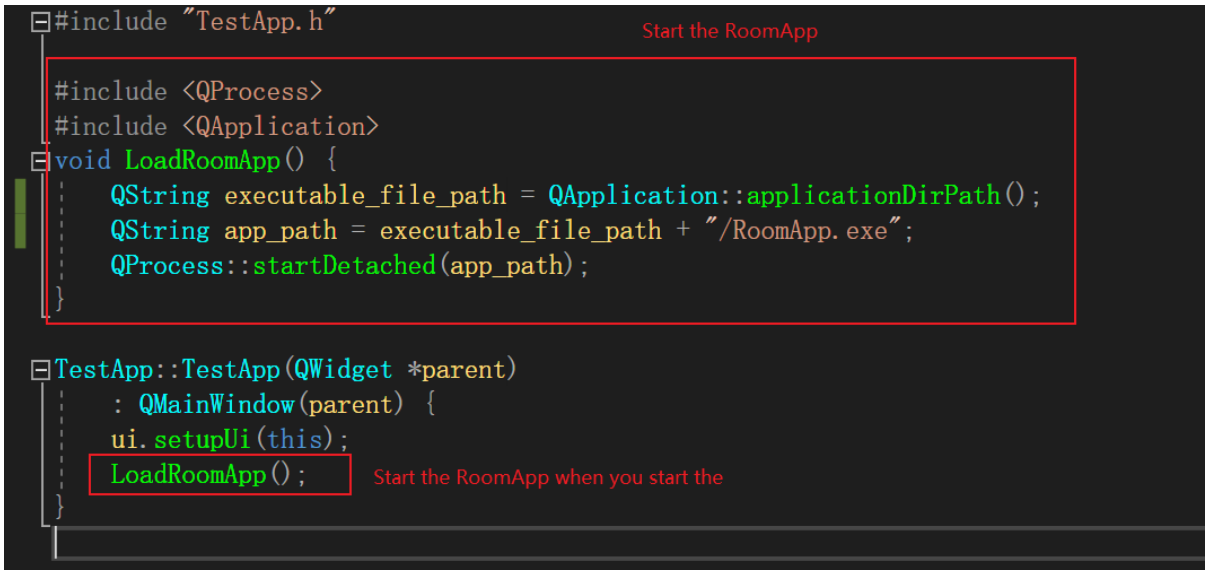
i. VSを開き、Qt Widgets Applicationプロジェクトタイプを選択し、TestAppプロジェクトを作成します。



ii. プロセスを起動するプログラムを作成し、適切な位置にLoadRoomApp関数を呼び出します。

```
#include <QProcess>
#include <QApplication>
void LoadRoomApp() {
QString executable_file_path = QApplication::applicationDirPath();
QString app_path = executable_file_path + "/RoomApp.exe";
QProcess::startDetached(app_path);
}
```

©2013-2022 Tencent Cloud. All rights reserved.



iii. プロジェクトをコンパイルし、RoomAppをコンパイルした成果物を現在実行可能なプログラムディレクトリにコピーします。release x86プログラムを例にとると、次のようになります：
`TUIRoom\Windows-Mac\RoomApp\bin\Win32\Release` ディレクトリ下のすべてのファイルを現在のプログラムディレクトリ下にコピーします。

iv. プログラムを実行します。TestAppの起動と同時にRoomAppを起動します。

カスタムUIの実装

- 当社が提供するAppで直接変更して適用するか、あるいはAppのソースコードのModuleを使用してカスタムUIを実装することができます
- 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ムUIを実装できます
- Appディレクトリには UIに関連する設計とロジックが含まれており、必要に応じてRoomAppソースコードを変更し、二次開発を行うことができます。主な機能ポイントは次のとおりです：

機能ポイント	ファイルディレクトリ
トップページログイン	Windows-Mac\RoomApp\App\LoginViewController.cpp
デバイス検査	Windows-Mac\RoomApp\App\PresetDeviceController.cpp
ホームページ	Windows-Mac\RoomApp\App\MainWindow.cpp
配信側リスト	Windows-Mac\RoomApp\App\StageListController.cpp

©2013-2022 Tencent Cloud. All rights reserved.

機能ポイント	ファイルディレクトリ
メンバーリスト	Windows-Mac\RoomApp\App\MemberListViewController.cpp
設定ページ	Windows-Mac\RoomApp\App\SettingViewController.cpp
チャットルーム	Windows-Mac\RoomApp\App\ChatRoomViewController.cpp
画面共有	Windows-Mac\RoomApp\App\ScreenShareWindow.cpp
下部ツールバー	Windows-Mac\RoomApp\App\BottomBarController.cpp

よくあるご質問

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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIRoom (Electron)の統合

最終更新日：：2022-10-31 14:41:31

コンポーネントの説明

TUIRoomは、UIを含むオープンソースオーディオビデオコンポーネントです。TUIRoomを統合することにより、ビジネスにおいて、オーディオビデオルーム、画面共有、チャットなどの機能を速やかにオンラインにすることができます。Electron端末のTUIRoomの基本機能は下図のとおりです。

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブ化すると、関連するIM SDKの体験版がデフォルトでアクティブ化されます。これは100 DAUのみをサポートします。



オンライン体験リンクMac OS版およびWindows版をクリックしてダウンロードすると、TUIRoom Electronのより多くの機能を体験することができます。
GithubをクリックしてTUIRoomのコードをダウンロードし、TUIRoom Electronデモプロジェクトクイックスタートのドキュメントを参照し、TUIRoom Electronデモプロジェクトをクイックスタートすることもできます。

©2013-2022 Tencent Cloud. All rights reserved.

現在の業務でElectron端末のTUIRoomコンポーネントを統合する必要がある場合は、こちらのドキュメントをご参照ください。

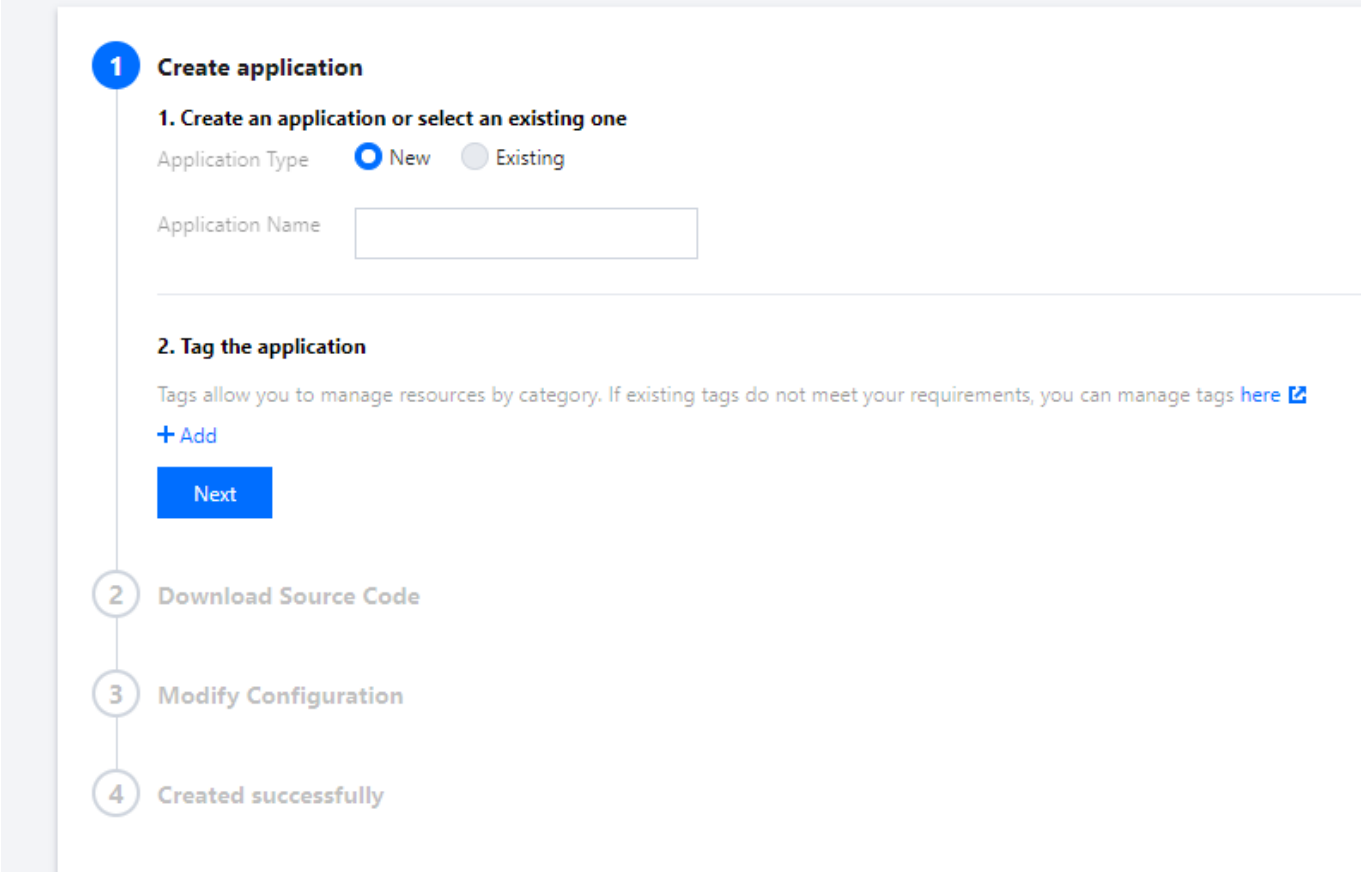
コンポーネントの統合

TUIRoomコンポーネントはVue3 + TS + Pinia + Element Plus + SCSSを使用して開発されています。プロジェクトの統合には、Electron + Vue3 + TSの技術スタックを使用する必要があります。

ステップ1： Tencent Cloud TRTCおよびIMサービスのアクティブ化

TUIRoomは、Tencent Cloud TRTCとIMサービスをベースに開発されています。

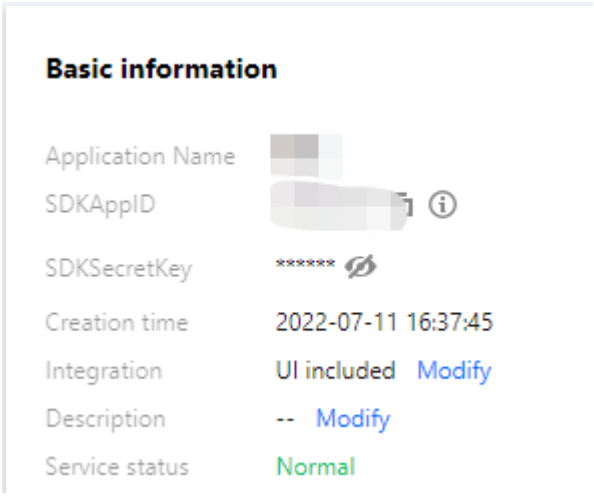
1. **TRTCアプリケーションの作成**
   - Tencent Cloudアカウントをまだお持ちでない場合は、Tencent Cloudアカウントの登録を行い、実名認証を完了してください。
   - TRTCコンソール で、**アプリケーション管理>アプリケーションの作成** をクリックし、新たなアプリケーションを作成します。



2. **TRTCアプリケーションおよびキー情報の取得**

©2013-2022 Tencent Cloud. All rights reserved.

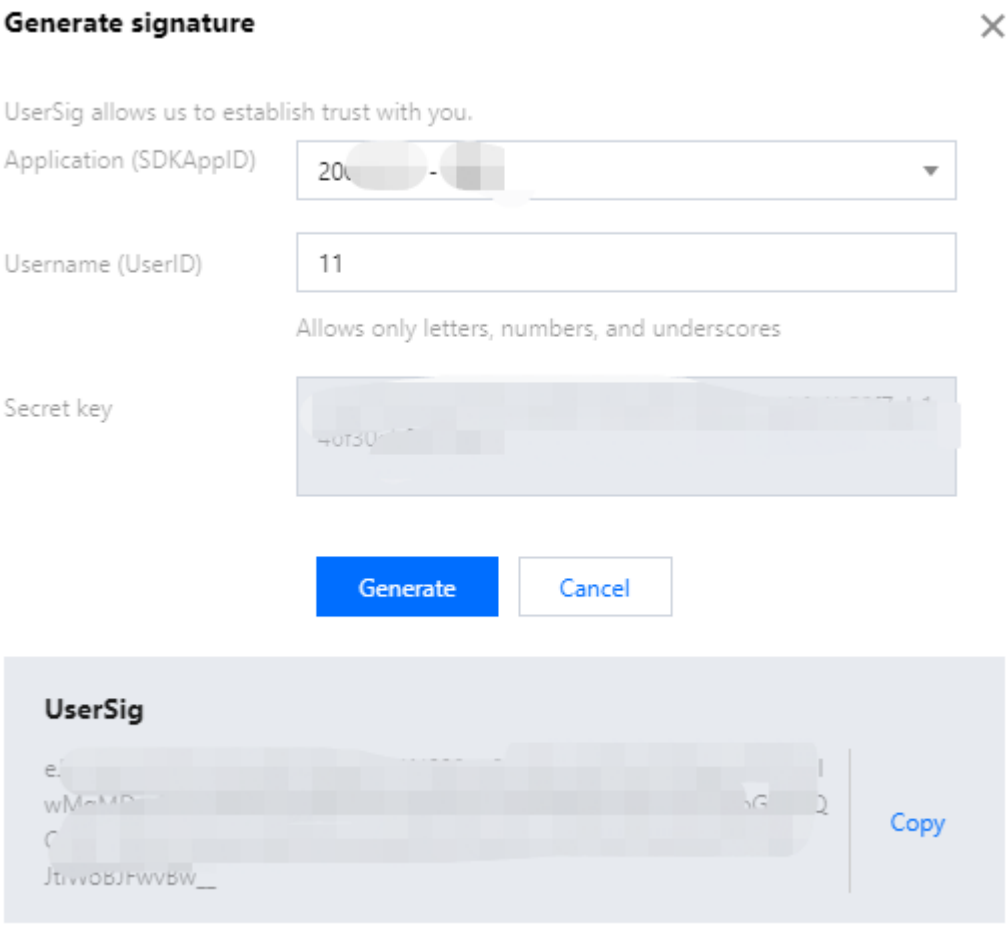
3. **アプリケーション管理 > アプリケーション情報**でSDKAppID の情報を取得します。SDKAppIDはTRTCのアプリケーションIDです。業務の分離、すなわち異なるSDKAppIDの通話が互いに接続できないようにするために用いられます。



4. **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。SecretKeyはTRTCのアプリケーションキーです。SDKAppIDとペアで使用する必要があり、TRTCサービスを正当に使用す

©2013-2022 Tencent Cloud. All rights reserved.

るための認証用証明書UserSigの発行に用いられます。



5. **UserSigの発行**

   UserSigとは、悪意ある攻撃者によるクラウドサービスの使用権の盗用を防ぐために、Tencent Cloudによって設計された、セキュリティ保護された署名です。TUIRoomの初期化にはUserSigパラメータの提供が必要です。

   - デバッグスタート段階でuserSigを発行する方法については、デバッグスタート段階でのUserSigの計算方法をご参照ください。
   - 本番環境でuserSigを発行する方法については、本番環境でのUserSigの計算方法をご参照ください。

ステップ2：TUIRoomコンポーネントのダウンロードとコピー

1. 業務側にすでにあるElectron + Vue3 + TSプロジェクトを開きます。Electron + Vue3 + TSプロジェクトがない場合は、このテンプレートGithubによってElectron + Vue3 + TSのテンプレートプロジェクトを生成することができます。

> 注意：

- このドキュメントで説明する統合手順はelectron-vite-vueテンプレートプロジェクトのバージョン1.0.0をベースにしています。
- electron-vite-vueテンプレートプロジェクトの最新バージョンではディレクトリ構造が調整されています。最新バージョンを使用される場合は、このドキュメントを参照してご自身でディレクトリの調整と設定を行うことができます。

2. テンプレートプロジェクトの生成に成功後、次のスクリプトを実行します。

```
cd electron-vite-vue
npm install
npm run dev
```

3. Githubをクリックし、TUIRoomリポジトリコードをクローンまたはダウンロードし、`TUIRoom/Electron/packages/renderer/src/TUIRoom` フォルダを既存のプロジェクトの `packages/renderer/src/` ディレクトリ下にコピーします。

ステップ3：TUIRoomコンポーネントのインポート

1. ページ内にTUIRoomコンポーネントをインポートします。例えば、 `App.vue` コンポーネントにTUIRoomコンポーネントをインポートします。
   - TUIRoomコンポーネントはユーザーをキャスターロールと一般メンバーロールに区分します。コンポーネントは外部に対しinit、createRoom、enterRoomのメソッドを提供します。
   - キャスターと一般メンバーはinitメソッドによってTUIRoomコンポーネントに対し、アプリケーションとユーザーデータの初期化を行うことができます。キャスターはcreateRoom メソッドによってルームを作成し入室することができます。一般メンバーはenterRoomメソッドによって、キャスターが作成したルームに入室することができます。

```
<template>
<room ref="TUIRoomRef"></room>
</template>
<script setup lang="ts">
import { ref, onMounted } from 'vue';
// TUIRoomコンポーネントをインポートします。インポートパスが正しいかどうかよく確認してください
import Room from './TUIRoom/index.vue';
// TUIRoomコンポーネント要素を取得し、TUIRoomコンポーネントの呼び出しメソッドに用います
const TUIRoomRef = ref();
onMounted(async () => {
// TUIRoomコンポーネントを初期化します
// キャスターはルームを作成する前にTUIRoomコンポーネントを初期化する必要があります
// 一般メンバーは入室前にTUIRoomコンポーネントを初期化する必要があります
```

©2013-2022 Tencent Cloud. All rights reserved.

```
await TUIRoomRef.value.init({
// sdkAppIdを取得します。ステップ1をご参照ください
sdkAppId: 0,
業務// ユーザーの業務における固有の表示IDです
userId: '',
// ローカル開発デバッグの場合はhttps://console.intl.cloud.tencent.com/trtc/usersigtoolページでuserSigをすぐに発行できます。userSigとuserIdは対になる関係であることに注意してください
userSig: '',
// ユーザーが業務で使用するニックネーム
userName: '',
// ユーザーが業務で使用するプロフィール画像リンク
userAvatar: '',
// ユーザーが画面共有に用いる固有のIdであり、shareUserId = `share_${userId}`である必要があります。画面共有機能が必要なければ渡さなくても結構です
shareUserId: '',
// 本文ステップ1 > 第3段階を参照し、sdkAppIdとshareUserIdを使用してshareUserSigを発行してください
shareUserSig: '',
})
// デフォルトではルーム作成が実行されます。実際の統合では必要に応じ、タイミングを見てhandleCreateRoomメソッドを実行することができます
await handleCreateRoom();
})
// キャスターがルームを作成します。このメソッドはルーム作成時にのみ呼び出します
async function handleCreateRoom() {
// roomIdはユーザーが入室するルームナンバーです。roomIdのタイプはnumberである必要があります
// roomModeには'FreeSpeech'(自由発言モード) と'ApplySpeech'(挙手発言モード)の2種類があります。デフォルトでは'FreeSpeech'です。現在は自由発言モードのみサポートされていますのでご注意ください
// roomParamはユーザーの入室時のデフォルト動作(デフォルトでマイクをオンにするかどうか、デフォルトでカメラをオンにするかどうか、デフォルトのメディアデバイスId)を指定するものです
const roomId = 123456;
const roomMode = 'FreeSpeech';
const roomParam = {
isOpenCamera: true,
isOpenMicrophone: true,
}
await TUIRoomRef.value.createRoom(roomId, roomMode, roomParam);
}
// 一般メンバーが入室します。このメソッドは一般メンバーが作成済みのルームに入室する際にのみ呼び出します
async function handleEnterRoom() {
// roomIdはユーザーが入室するルームナンバーです。roomIdのタイプはnumberである必要があります
// roomParamはユーザーの入室時のデフォルト動作(デフォルトでマイクをオンにするかどうか、デ
```

```
フォルトでカメラをオンにするかどうか、デフォルトのメディアデバイスId)を指定するものです
const roomId = 123456;
const roomParam = {
isOpenCamera: true,
isOpenMicrophone: true,
}
await TUIRoomRef.value.enterRoom(roomId, roomParam);
}
</script>
<style>
html, body {
width: 100%;
height: 100%;
margin: 0;
}
#app {
width: 100%;
height: 100%;
}
</style>
```

注意：
ページ内で上記のコードをコピーした後、TUIRoomインターフェースのパラメータを実際のデータに変更する必要があります。

ステップ4：本番環境の設定

TUIRoomコンポーネントをインポートした後、プロジェクトが正常に実行されることを確認するため、次の設定を行います。

1. **依存のインストール**

- 開発環境の依存をインストールします。

```
npm install sass typescript unplugin-auto-import unplugin-vue-components -S -D
```

- 本番環境の依存をインストールします。

```
npm install element-plus events mitt pinia trtc-electron-sdk tim-js-sdk tsignal
ing -S
```

©2013-2022 Tencent Cloud. All rights reserved.

2. **Piniaの登録**

TUIRoomはPiniaを使用してルームデータの管理を行うため、プロジェクトのエントリーファイルでPiniaの登録を行う必要があります。プロジェクトのエントリーファイルは `packages/renderer/src/main.ts` ファイルです。

```
// src/main.tsファイル
import { createPinia } from 'pinia';
const app = createApp(App);
// Piniaの登録
createApp(App)
.use(createPinia())
.mount('#app')
.$nextTick(window.removeLoading)
```

3. **element-plusを必要に応じてインポートするよう設定**

- TUIRoomはelement-plus UIコンポーネントを使用します。すべてのelement-plusコンポーネントをインポートしてしまわないように、 `packages/renderer/vite.config.ts` で、element-plusコンポーネントを必要に応じてロードするように設定しておく必要があります。

> 注意：
> 以下の設定項目は増分設定です。すでに存在するVite設定項目を削除しないでください。

```
// vite.config.ts
import AutoImport from 'unplugin-auto-import/vite';
import Components from 'unplugin-vue-components/vite';
import { ElementPlusResolver } from 'unplugin-vue-components/resolvers';
const path = require('path');
export default defineConfig({
// ...
plugins: [
AutoImport({
resolvers: [ElementPlusResolver()],
}),
Components({
resolvers: [ElementPlusResolver({
importStyle: 'sass',
})],
}),
],
css: {
```

©2013-2022 Tencent Cloud. All rights reserved.

```
preprocessorOptions: {
scss: {
additionalData: `
@use '${path.resolve(__dirname, 'src/TUIRoom/assets/style/element.scss')}' as *;
`,
},
},
},
});
```

- また、element-**plus**に付帯するUIコンポーネントがスタイルを正常に表示できるようにするため、エントリーファイル`packages/renderer/src/main.ts`でelement-**plus**コンポーネントスタイルをロードする必要があります。

```
// src/main.ts
import 'element-plus/theme-chalk/el-message.css'
import 'element-plus/theme-chalk/el-message-box.css'
```

4. **trtc-electron-sdkのインポート**

UIレイヤーでimport方式により `trtc-electron-sdk` をインポートするためには、コードスタイルを統一します。これを行わない場合はrequire方式でのインポートが必要になります。 `packages/renderer/vite.config.ts` での設定が必要です。

> 注意：
> 以下の設定項目でresolveの内容を置き換えます。

```
// vite.config.ts
export default defineConfig({
// ...
plugins: [
resolve(
{
"trtc-electron-sdk": `
const TRTCCloud = require("trtc-electron-sdk");
const TRTCParams = TRTCCloud.TRTCParams;
const TRTCAppScene = TRTCCloud.TRTCAppScene;
const TRTCVideoStreamType = TRTCCloud.TRTCVideoStreamType;
const TRTCScreenCaptureSourceType = TRTCCloud.TRTCScreenCaptureSourceType;
const TRTCVideoEncParam = TRTCCloud.TRTCVideoEncParam;
const Rect = TRTCCloud.Rect;
```

©2013-2022 Tencent Cloud. All rights reserved.

```
const TRTCAudioQuality = TRTCCloud.TRTCAudioQuality;
const TRTCScreenCaptureSourceInfo = TRTCCloud.TRTCScreenCaptureSourceInfo;
const TRTCDeviceInfo = TRTCCloud.TRTCDeviceInfo;
const TRTCVideoQosPreference = TRTCCloud.TRTCVideoQosPreference;
const TRTCQualityInfo = TRTCCloud.TRTCQualityInfo;
const TRTCStatistics = TRTCCloud.TRTCStatistics;
const TRTCVolumeInfo = TRTCCloud.TRTCVolumeInfo;
const TRTCDeviceType = TRTCCloud.TRTCDeviceType;
const TRTCDeviceState = TRTCCloud.TRTCDeviceState;
const TRTCBeautyStyle = TRTCCloud.TRTCBeautyStyle;
const TRTCVideoResolution = TRTCCloud.TRTCVideoResolution;
const TRTCVideoResolutionMode = TRTCCloud.TRTCVideoResolutionMode;
const TRTCVideoMirrorType = TRTCCloud.TRTCVideoMirrorType;
const TRTCVideoRotation = TRTCCloud.TRTCVideoRotation;
const TRTCVideoFillMode = TRTCCloud.TRTCVideoFillMode;
export {
TRTCParams,
TRTCAppScene,
TRTCVideoStreamType,
TRTCScreenCaptureSourceType,
TRTCVideoEncParam,
Rect,
TRTCAudioQuality,
TRTCScreenCaptureSourceInfo,
TRTCDeviceInfo,
TRTCVideoQosPreference,
TRTCQualityInfo,
TRTCStatistics,
TRTCVolumeInfo,
TRTCDeviceType,
TRTCDeviceState,
TRTCBeautyStyle,
TRTCVideoResolution,
TRTCVideoResolutionMode,
TRTCVideoMirrorType,
TRTCVideoRotation,
TRTCVideoFillMode,
};
export default TRTCCloud.default;
`,
}
),
]
// ...
});
```

©2013-2022 Tencent Cloud. All rights reserved.

5. **env.d.tsファイル設定**
   - env.d.tsファイル設定は `packages/renderer/src/env.d.ts` で行う必要があります。

> 注意：
> 以下の設定項目は増分設定です。すでに存在するenv.d.tsファイル設定を削除しないでください。

```
// env.d.ts
declare module 'tsignaling/tsignaling-js' {
import TSignaling from 'tsignaling/tsignaling-js';
export default TSignaling;
}
declare module 'tim-js-sdk' {
import TIM from 'tim-js-sdk';
export default TIM;
}
```

6. **プロジェクト内にimport動的ロードが存在する場合は、アーキテクチャ設定を変更し、esモジュールをパッケージ化して生成する必要があります**
   - esモジュールをパッケージ化して生成するには、 `packages/renderer/vite.config.ts` 内で設定を行う必要があります。

> 注意：
> プロジェクト内にimport動的ロードが存在しない場合は、この設定を行わないでください。以下の設定項目は増分設定です。すでに存在するVite設定項目を削除しないでください。

```
// vite.config.ts
export default defineConfig({
// ...
build: {
rollupOptions: {
output: {
format: 'es'
}
}
},
});
```

©2013-2022 Tencent Cloud. All rights reserved.

ステップ5：開発環境の実行

コンソールの実行開発環境でスクリプトを実行し、TUIRoomの含まれるページをブラウザで開くと、そのページでTUIRoomコンポーネントを使用することができます。
ステップ2のスクリプトを使用してElectron + Vue3 + TSプロジェクトを作成する場合は、次の事項を行う必要があります。

1. 開発環境コマンドを実行します。

```
npm run dev
```

注意：
TUIRoomはelement-plusコンポーネントを必要に応じてインポートするため、開発環境のルーティングページでの初回ロード時に反応がやや遅くなりますが、必要なelement-plusのロードが完了すると正常に使用できるようになります。element-plusの必要に応じてのロードがパッケージ化後のページロードに影響することはありません。
2. TUIRoomコンポーネント機能を体験します。

ステップ6：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、次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 `release` ディレクトリにあり、インストールの実行が可能です。

```
npm run build
```

注意：
Mac PCを使用したMacインストールパッケージの作成、Windows PCを使用したWindowsインストールパッケージの作成のみ可能です。

付録：TUIRoom API

TUIRoomインターフェース

init

TUIRoomデータを初期化します。TUIRoomを使用するすべてのユーザーはinitメソッドを呼び出す必要があります。

©2013-2022 Tencent Cloud. All rights reserved.

```
TUIRoomRef.value.init(roomData);
```

パラメータは下表に示すとおりです。

パラメータ	タイプ	意味
roomData	object	
roomData.sdkAppId	number	お客様のSDKAppId
roomData.userId	string	ユーザーの固有ID
roomData.userSig	string	ユーザーのUserSig
roomData.userName	string	ユーザーのニックネーム
roomData.userAvatar	string	ユーザーのプロフィール画像
roomData.shareUserId	string	任意入力。ユーザーが画面共有を行う際のUserIdであり、`shareUserId = share_${userId}`である必要があります。画面共有機能がなければ渡さなくても結構です
roomData.shareUserSig	string	任意入力。ユーザーが画面共有を行う際のUserSig

createRoom

キャスターがルームを作成します。

```
TUIRoomRef.value.createRoom(roomId, roomMode, 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	number	ルームID
roomMode	string	ルームモード。'FreeSpeech'（自由発言モード）と'ApplySpeech'（挙手発言モード）があり、デフォルトでは'FreeSpeech'です。現在は自由発言モードのみサポートされていますのでご注意ください
roomParam	Object	任意入力
roomParam.isOpenCamera	string	任意入力。入室時にカメラをオンにするかどうか。デフォルトではオフ

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
roomParam.isOpenMicrophone	string	任意入力。入室時にマイクをオンにするかどうか。デフォルトではオフ
roomParam.defaultCameraId	string	任意入力。デフォルトのカメラデバイスID
roomParam.defaultMicrophoneId	string	任意入力。デフォルトのマイクデバイスID
roomParam.defaultSpeakerId	String	任意入力。デフォルトのスピーカーデバイスID

enterRoom

一般メンバーが入室します。

```
TUIRoomRef.value.enterRoom(roomId, 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	number	ルームID
roomParam	Object	任意入力
roomParam.isOpenCamera	string	任意入力。入室時にカメラをオンにするかどうか。デフォルトではオフ
roomParam.isOpenMicrophone	string	任意入力。入室時にマイクをオンにするかどうか。デフォルトではオフ
roomParam.defaultCameraId	string	任意入力。デフォルトではカメラデバイスID
roomParam.defaultMicrophoneId	string	任意入力。デフォルトではマイクデバイスID
roomParam.defaultSpeakerId	String	任意入力。デフォルトではスピーカーデバイスID

TUIRoomイベント

onRoomCreate

ルーム作成のコールバックです。

```
<template>
<room ref="TUIRoomRef" @on-room-create="handleRoomCreate"></room>
</template>
<script setup lang="ts">
```

©2013-2022 Tencent Cloud. All rights reserved.

```
// TUIRoomコンポーネントをインポートします。インポートパスが正しいかどうかよく確認してください
import Room from './TUIRoom/index.vue';

function handleRoomCreate(info) {
if (info.code === 0) {
console.log('ルーム作成成功')
}
}
</script>
```

onRoomEnter

入室コールバックです。

```
<template>
<room ref="TUIRoomRef" @on-room-enter="handleRoomEnter"></room>
</template>
<script setup lang="ts">
// TUIRoomコンポーネントをインポートします。インポートパスが正しいかどうかよく確認してください
import Room from './TUIRoom/index.vue';

function handleRoomEnter(info) {
if (info.code === 0) {
console.log('入室成功')
}
}
</script>
```

onRoomDestory

キャスターのルーム破棄通知。

```
<template>
<room ref="TUIRoomRef" @on-room-destory="handleRoomDestory"></room>
</template>
<script setup lang="ts">
// TUIRoomコンポーネントをインポートします。インポートパスが正しいかどうかよく確認してください
import Room from './TUIRoom/index.vue';

function handleRoomDestory(info) {
if (info.code === 0) {
console.log('キャスターが破棄に成功')
}
}
</script>
```

©2013-2022 Tencent Cloud. All rights reserved.

onRoomExit

一般メンバーのルーム退出通知。

```
<template>
<room ref="TUIRoomRef" @on-room-exit="handleRoomExit"></room>
</template>
<script setup lang="ts">
// TUIRoomコンポーネントをインポートします。インポートパスが正しいかどうかよく確認してください
import Room from './TUIRoom/index.vue';

function handleRoomExit(info) {
if (info.code === 0) {
console.log('一般メンバーのルーム退出成功')
}
}
</script>
```

©2013-2022 Tencent Cloud. All rights reserved.

TUIRoom APIのクエリー
TUIRoom(Android)

最終更新日：：2022-10-31 14:41:31

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブ化すると、関連するIM SDKの体験版がデフォルトでアクティブ化されます。これは100 DAUのみをサポートします。

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(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

©2013-2022 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）。
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）。
leaveRoom	退室（参加者が呼び出し）。
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	ローカル画面のイメージプレビューモードを設定します。
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

©2013-2022 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	参加者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	参加者がキャスターの発言申請に同意/拒否します。
sendSpeechApplication	参加者が発言を申請します。
replySpeechApplication	キャスターが参加者の発言申請に同意/拒否します。
forbidSpeechApplication	キャスターが発言申請を禁止します。
sendOffSpeaker	キャスターが参加者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	参加者は発言を停止し、視聴者になります。

©2013-2022 Tencent Cloud. All rights reserved.

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。

美顔フィルター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。

©2013-2022 Tencent Cloud. All rights reserved.

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。
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申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	参加者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、参加者が受信する場合のコールバック。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
onCallingRollStopped	キャスターが点呼を終了し、参加者が受信する場合のコールバック。
onMemberReplyCallingRoll	参加者が点呼に応答し、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	参加者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。

©2013-2022 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, TUIRoomCoreC
allback.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

©2013-2022 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（参加者が呼び出し）。

```
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。

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、roomId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。
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）。

```
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。

©2013-2022 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<TUIRoomCoreDef.UserInfo> getRoomUsers();
```

getUserInfo

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.Action
Callback 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	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。

©2013-2022 Tencent Cloud. All rights reserved.

transferRoomMaster

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。

©2013-2022 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

ローカル画面のイメージプレビューモードを設定します。

```
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。

©2013-2022 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

©2013-2022 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.ActionCa
llback 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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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);
```

©2013-2022 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	無効にするかどうか。
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

©2013-2022 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。

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

参加者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

キャスターが参加者の発言を要請します。

```
void sendSpeechInvitation(String userId, TUIRoomCoreCallback.InvitationCallback c
allback);
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

cancelSpeechInvitation

キャスターが参加者の発言要請をキャンセルします。

```
void cancelSpeechInvitation(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。

replySpeechInvitation

参加者がキャスターの発言要請に同意/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

参加者が発言を申請します。

```
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

参加者が発言申請をキャンセルします。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。

replySpeechApplication

キャスターが参加者の発言申請に同意/拒否します。

```
void replySpeechApplication(boolean agree, String userId, TUIRoomCoreCallback.Act
ionCallback 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 c
allback);
```

パラ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

キャスターが参加者に発言の停止を命令します。

```
void sendOffSpeaker(String userId, TUIRoomCoreCallback.ActionCallback callback);
```

©2013-2022 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。

sendOffAllSpeakers

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

参加者が発言を停止し、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.TR
TCScreenShareParams screenShareParams);
```

パラメータは下表に示すとおりです。

©2013-2022 Tencent Cloud. All rights reserved.

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。
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();
```

美顔フィルター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。

©2013-2022 Tencent Cloud. All rights reserved.

- 「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。

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

©2013-2022 Tencent Cloud. All rights reserved.

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);
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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
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);
```

©2013-2022 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です。

```
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。

©2013-2022 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。

©2013-2022 Tencent Cloud. All rights reserved.

```
void onRemoteUserScreenVideoAvailable(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	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。

©2013-2022 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);
```

©2013-2022 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申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。

©2013-2022 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

参加者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、参加者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、参加者が受信する場合のコールバック。

```
void onCallingRollStopped(String userId);
```

onMemberReplyCallingRoll

参加者が点呼に応答し、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。

©2013-2022 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

©2013-2022 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<TRTCCloudDef.TRTCQuality> 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	アップストリームネットワークの品質。
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

©2013-2022 Tencent Cloud. All rights reserved.

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(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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIRoom API（iOS）

最終更新日：：2022-10-31 14:41:31

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブ化すると、関連するIM SDKの体験版がデフォルトでアクティブ化されます。これは100 DAUのみをサポートします。

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

©2013-2022 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）。
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）。
leaveRoom	退室（参加者が呼び出し）。
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	ローカル画面のイメージプレビューモードを設定します。
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

©2013-2022 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	参加者がキャスターの点呼に応答します。
sendSpeechInvitation	キャスターが参加者の発言を要請します。
cancelSpeechInvitation	キャスターが参加者の発言要請をキャンセルします。
replySpeechInvitation	参加者がキャスターの発言申請に同意/拒否します。
sendSpeechApplication	参加者が発言を申請します。
replySpeechApplication	キャスターが参加者の発言申請に同意/拒否します。
forbidSpeechApplication	キャスターが発言申請を禁止します。
sendOffSpeaker	キャスターが参加者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	参加者は発言を停止し、視聴者になります。

©2013-2022 Tencent Cloud. All rights reserved.

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。

美顔フィルター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。

©2013-2022 Tencent Cloud. All rights reserved.

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。
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申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	参加者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、参加者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、参加者が受信する場合のコールバック。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
onMemberReplyCallingRoll	参加者が点呼に応答し、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	参加者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

©2013-2022 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<TUIRoomCoreDelegate>)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;
```

©2013-2022 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（参加者が呼び出し）。

```
- (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。

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、roomId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。
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）。

```
- (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。

©2013-2022 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<TUIRoomUserInfo *> *)getRoomUsers;
```

getUserInfo

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。

©2013-2022 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

©2013-2022 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

ローカル画面のイメージプレビューモードを設定します。

```
- (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
```

©2013-2022 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。

©2013-2022 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;
```

©2013-2022 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。

©2013-2022 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。

©2013-2022 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

参加者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

キャスターが参加者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

キャスターが参加者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。

©2013-2022 Tencent Cloud. All rights reserved.

replySpeechInvitation

参加者がキャスターの発言要請に同意/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

参加者が発言を申請します。

```
- (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

参加者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

キャスターが参加者の発言申請に同意/拒否します。

```
- (void)replySpeechApplication:(BOOL)agree
userId:(NSString *)userId
```

©2013-2022 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

キャスターが参加者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。

©2013-2022 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

参加者が発言を停止し、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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (void)stopScreenCapture API_AVAILABLE(ios(11.0));
```

美顔フィルター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。

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;
```

©2013-2022 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。

©2013-2022 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。

©2013-2022 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

©2013-2022 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です。

```
- (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。

©2013-2022 Tencent Cloud. All rights reserved.

onRoomMasterChanged

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。

©2013-2022 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。

©2013-2022 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。

©2013-2022 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申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。

©2013-2022 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

参加者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、参加者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、参加者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。

©2013-2022 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

参加者が点呼に応答し、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。

©2013-2022 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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (void)onNetworkQuality:(TRTCQualityInfo *)localQuality remoteQuality:(NSArray<TRTCQualityInfo *> *)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。

©2013-2022 Tencent Cloud. All rights reserved.

TRTCMeeting API（Flutter）

最終更新日：：2022-07-19 16:21:31

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

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。

©2013-2022 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。

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）。
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

©2013-2022 Tencent Cloud. All rights reserved.

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。

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

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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

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一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。

©2013-2022 Tencent Cloud. All rights reserved.

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

destroySharedInstance

TRTCMeetingシングルトンオブジェクトを破棄します。

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

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してください。

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<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

©2013-2022 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。
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

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	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<ActionCallback> 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

©2013-2022 Tencent Cloud. All rights reserved.

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、一括管理してください。

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

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<ActionCallback> 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、一括管理してください。

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<ActionCallback> 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。

©2013-2022 Tencent Cloud. All rights reserved.

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<ActionCallback> 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<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<String>	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<UserListCallback> 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：

©2013-2022 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。
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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

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

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);
```

©2013-2022 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	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);
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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

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);
```

©2013-2022 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
volume	int	集音音量、値：0 - 100、デフォルト値：100。

setAudioPlayoutVolume

再生音量の設定。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。

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：

©2013-2022 Tencent Cloud. All rights reserved.

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

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。

©2013-2022 Tencent Cloud. All rights reserved.

- 「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。

```
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	コマンドワードです。開発者がカスタマイズするものであり、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

©2013-2022 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。

パラ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。
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。

©2013-2022 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	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。

パラ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。

©2013-2022 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
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。

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

©2013-2022 Tencent Cloud. All rights reserved.

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。
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。

©2013-2022 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	コマンドワードです。開発者がカスタマイズするものであり、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一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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIRoom(Windows&Mac)

最終更新日：：2022-10-31 14:41:31

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブ化すると、関連するIM SDKの体験版がデフォルトでアクティブ化されます。これは100 DAUのみをサポートします。

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

©2013-2022 Tencent Cloud. All rights reserved.

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）。
LeaveRoom	退室（参加者が呼び出し）。
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	ローカル画面のイメージプレビューモードを設定します。

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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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
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	キャスターが参加者の発言を要請します。
CancelSpeechInvitation	キャスターが参加者の発言要請をキャンセルします。
ReplySpeechInvitation	参加者がキャスターの発言申請に同意/拒否します。
SendSpeechApplication	参加者が発言を申請します。
CancelSpeechApplication	参加者が発言申請をキャンセルします。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
ReplySpeechApplication	キャスターが参加者の発言申請に同意/拒否します。
ForbidSpeechApplication	キャスターが発言申請を禁止します。
SendOffSpeaker	キャスターが参加者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	参加者は発言を停止し、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

©2013-2022 Tencent Cloud. All rights reserved.

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

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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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。

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申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	キャスターが発言申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	参加者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、参加者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、参加者が受信する場合のコールバック。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
OnMemberReplyCallingRoll	参加者が点呼に応答し、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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
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）。

```
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。ルームの作成の成否は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

©2013-2022 Tencent Cloud. All rights reserved.

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。
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<TUIUserInfo> GetRoomUsers() = 0;
```

GetUserInfo

メンバー情報を取得します。

```
virtual const TUIUserInfo* GetUserInfo(const std::string& user_id) = 0;
```

©2013-2022 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。

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

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;
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。

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& vie
w,
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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, lite
av::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
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

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
virtual int MuteUserMicrophone(const std::string& user_id, bool mute, Callback ca
llback) = 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。

©2013-2022 Tencent Cloud. All rights reserved.

```
virtual int MuteUserCamera(const std::string& user_id, bool mute, Callback callba
ck) = 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;
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

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;
```

ReplyCallingRoll

参加者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

キャスターが参加者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。

©2013-2022 Tencent Cloud. All rights reserved.

CancelSpeechInvitation

キャスターが参加者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

参加者がキャスターの発言要請に同意/拒否します。

```
virtual int ReplySpeechInvitation(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
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

参加者が発言を申請します。

```
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

参加者が発言申請をキャンセルします。

©2013-2022 Tencent Cloud. All rights reserved.

```
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

キャスターが参加者の発言申請に同意/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。
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

キャスターが参加者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

©2013-2022 Tencent Cloud. All rights reserved.

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

参加者が発言を停止し、視聴者になります。

```
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

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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。

©2013-2022 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。

```
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

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

©2013-2022 Tencent Cloud. All rights reserved.

```
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。
message	string	エラー情報。

OnCreateRoom

ルーム作成のコールバックです。

```
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です。

```
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。

©2013-2022 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	エラー情報。

OnEnterRoom

入室コールバックです。

```
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。

OnExitRoom

退室コールバックです。

```
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 st
ream_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

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

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 availab
le) = 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。

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 availab
le) = 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。

©2013-2022 Tencent Cloud. All rights reserved.

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
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

©2013-2022 Tencent Cloud. All rights reserved.

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。
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;
```

©2013-2022 Tencent Cloud. All rights reserved.

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申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
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;
```

©2013-2022 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。

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

参加者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。

```
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、参加者が受信する場合のコールバック。

```
virtual void OnCallingRollStopped() = 0;
```

OnMemberReplyCallingRoll

参加者が点呼に応答し、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。

©2013-2022 Tencent Cloud. All rights reserved.

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;
```

パラ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

©2013-2022 Tencent Cloud. All rights reserved.

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, litea
v::TRTCQualityInfo* 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
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) {}
```

©2013-2022 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
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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
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

©2013-2022 Tencent Cloud. All rights reserved.

音声インタラクティブライブストリーミング TUIVoiceRoom (Android)の統合

最終更新日：：2022-09-06 14:19:42

コンポーネントの説明

TUIVoiceRoomはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIVoiceRoomコンポーネントを統合することにより、数行のコードを書くだけで、Appに「多人数ボイスチャット」などのシーンを組み込むことができます。TUIVoiceRoomはiOSなどのプラットフォームをサポートしています。基本機能は下図のとおりです：

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

©2013-2022 Tencent Cloud. All rights reserved.



コンポーネントの統合

ステップ1：TUIVoiceRoomコンポーネントのダウンロードとインポート

クリックしてGithubに進み、コードのクローン/ダウンロードを選択した後、Android/Sourceディレクトリをプロジェクトにコピーし、次のようにインポート動作を完了します：

- `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'
```

- appのbuild.gradleファイルにSourceに対する依存関係を追加します：

```
api project(':Source')
```

- ルートディレクトリの `build.gradle` ファイルに `TRTC SDK` および `IM SDK` の依存関係を追加します

©2013-2022 Tencent Cloud. All rights reserved.

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

ステップ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システムではマイクの権限などを動的に申請してください）：

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

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.** { *; }
```

ステップ3：初期化およびログイン

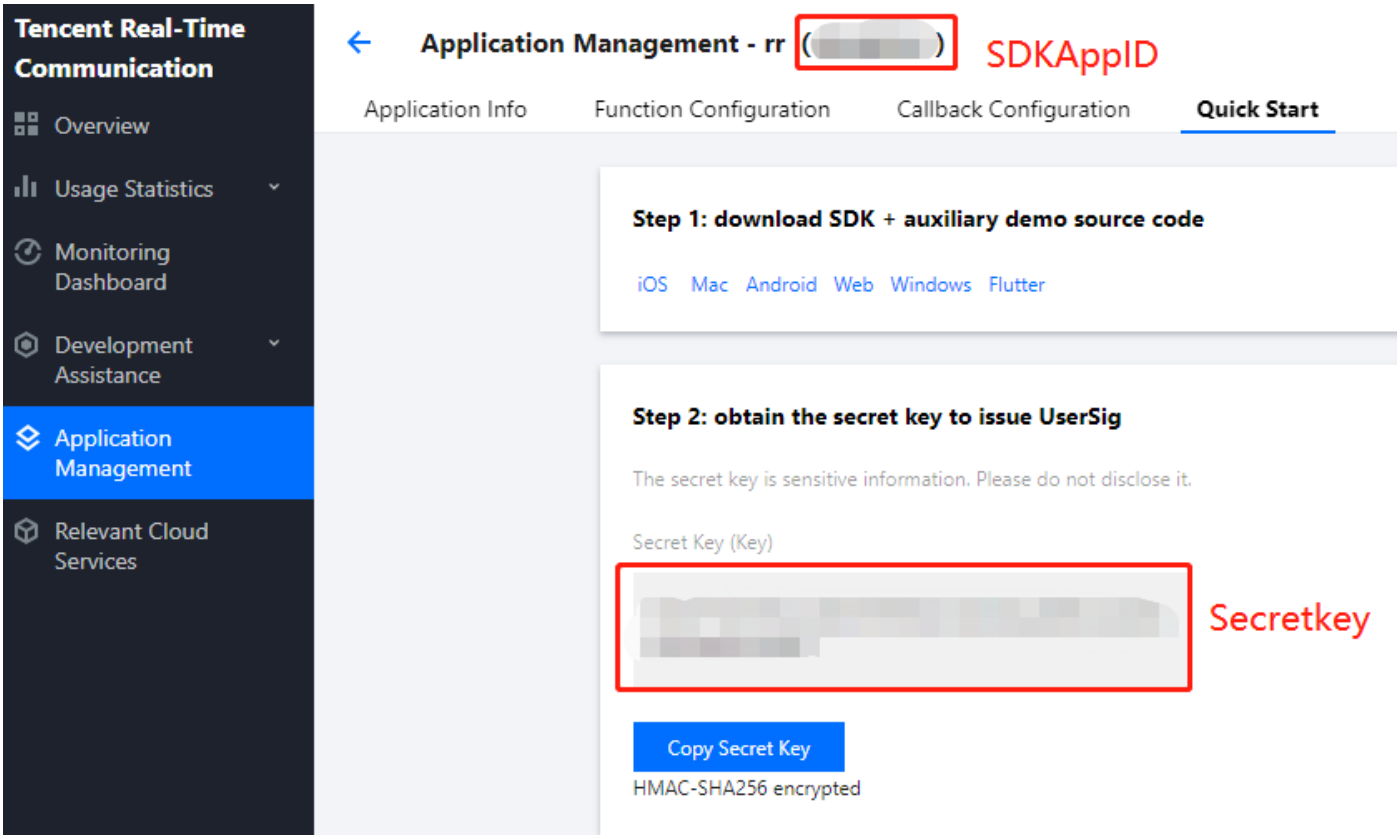
```
// 1.初期化
TRTCVoiceRoom mTRTCVoiceRoom = TRTCVoiceRoom.sharedInstance(this);
mTRTCVoiceRoom.setDelegate(new TRTCVoiceRoomDelegate() {
);
// 2.ログイン
mTRTCVoiceRoom.login(SDKAppID, userId, userSig, new TRTCVoiceRoomCallback.ActionC
allback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//ログイン成功
}
}
});
```

パラメータの説明：

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション**

©2013-2022 Tencent Cloud. All rights reserved.

情報をクリックすると、SDKAppID情報が次の図のように表示されます：



- **Secretkey**：**TRTC アプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます。
- **userId**：現在のユーザーID。文字列タイプであり、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**：SDKAppId、userId，Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のuserSigがオンラインで直接生成されます。UserSigの計算、使用方法をご参照ください。

ステップ4：ボイスチャットルームの実装

1. **管理者によるボイスチャットルーム作成の実装TRTCVoiceRoom#createRoom**

```
// 1.管理者が呼び出してルームを作成
int roomId = 12345; //ルームid
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
```

©2013-2022 Tencent Cloud. All rights reserved.

```
roomParam.coverUrl = "ルームカバー図のURL ";
mTRTCVoiceRoom.createRoom(roomId, roomParam, new TRTCVoiceRoomCallback.ActionCa
llback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//作成に成功
}
}
});
```

2. リスナーのボイスチャットルーム入室の実装TRTCVoiceRoom#enterRoom

```
// 1.リスナーが呼び出して入室
mTRTCVoiceRoom.enterRoom(roomId, new TRTCVoiceRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//入室に成功
}
}
});
```

3. リスナーの自主的なマイク・オンの実装TRTCVoiceRoom#enterRoom

```
// 1: リスナーが呼び出してマイク・オン
int seatIndex = 2; //マイクのindex
mTRTCVoiceRoom.enterSeat(seatIndex, new TRTCVoiceRoomCallback.ActionCallback()
{
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//操作に成功しました
}
}
});
// 2. onSeatListChangeコールバックを受信し、マイクリストを更新します
@Override
public void onSeatListChange(final List<TRTCVoiceRoomDef.SeatInfo> seatInfoLis
t) {
}
```

4. 管理者による視聴者発言招待の実装TRTCVoiceRoom#pickSeat

©2013-2022 Tencent Cloud. All rights reserved.

```
// 1: 管理者が呼び出して、視聴者が発言できるように招待
int seatIndex = 2; //マイクのindex
String userId = "123"; //マイク・オンが必要なユーザーid
mTRTCVoiceRoom.pickSeat(1, userId, new TRTCVoiceRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//操作に成功しました
}
}
});
// 2. onSeatListChangeコールバックを受信し、マイクリストを更新します
@Override
public void onSeatListChange(final List<TRTCVoiceRoomDef.SeatInfo> seatInfoLis
t) {
}
```

5. リスナーによるマイク・オン申請の実装 TRTCVoiceRoom#sendInvitation

```
// リスナー側の視点
// 1.リスナーが呼び出してマイク・オンを申請
String seatIndex = "1"; //マイクのindex
String userId = "123"; //ユーザーid
String inviteId = mTRTCVoiceRoom.sendInvitation("takeSeat", userId, seatIndex,
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(index, 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 content) {
if (cmd.equals("takeSeat")) {
// 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);
}
}
```

©2013-2022 Tencent Cloud. All rights reserved.

6. 管理者によるマイク・オンへの招待の実装 TRTCVoiceRoom#sendInvitation

```
// 管理者側の視点
// 1.管理者が呼び出して、視聴者が発言できるように招待をリクエスト
String seatIndex = "1"; //マイクのindex
String userId = "123"; //ユーザーid
String inviteId = mTRTCVoiceRoom.sendInvitation("pickSeat", userId, seatIndex,
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.pickSeat(index, null);
}
}
// リスナー側の視点
// 1.リスナー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 content) {
if (cmd.equals("pickSeat")) {
// 2.リスナーが管理者のリクエストに同意します
mTRTCVoiceRoom.acceptInvitation(id, null);
}
}
```

7. テキストチャットの実装TRTCVoiceRoom#sendRoomTextMsg

```
// 発信側：テキストメッセージの発信
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 userInf
o) {
Log.d(TAG,"が" + userInfo.userName + "から受信したメッセージ:" + message);
}
});
```

8. 弾幕コメントの実装 TRTCVoiceRoom#sendRoomCustomMsg

```
// 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
```

©2013-2022 Tencent Cloud. All rights reserved.

```
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.Us
erInfo userInfo) {
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！");
}
}
});
```

よくあるご質問

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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIVoiceRoom (iOS)の統合

最終更新日：：2022-09-06 14:16:36

コンポーネントの説明

TUIVoiceRoomはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIVoiceRoomコンポーネントを統合することにより、数行のコードを書くだけで、Appに「多人数ボイスチャット」などのシーンを組み込むことができます。TUIVoiceRoomはAndroidなどのプラットフォームをサポートしています。基本機能は下図のとおりです：

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになりますs。



©2013-2022 Tencent Cloud. All rights reserved.

コンポーネントの統合

ステップ1：TUIVoiceRoomコンポーネントのダウンロードとインポート

xcodeプロジェクトの `Podfile` ファイルと同一階層のディレクトリ下に `TUIVoiceRoom` フォルダを作成し、Githubリポジトリ iOSディレクトリ下のTXAppBasic、Resources、Source、TUIVoiceRoom.podspecなどのファイルを、ご自身のプロジェクトで作成した `TUIVoiceRoom` ディレクトリ下にコピーします。さらに、次のようにインポート動作を完了します：

- プロジェクトのPodfileファイルを開き、TUIVocieRoom.podspecをインポートします。次をご参照ください：

```
# pathはTXAppBasic.podspecのPodfileファイルに対する相対パスです
pod 'TXAppBasic', :path => "TUIVoiceRoom/TXAppBasic/"
# pathはTUIVoiceRoom.podspecのPodfileファイルに対する相対パスです
pod 'TUIVoiceRoom', :path => "TUIVoiceRoom/", :subspecs => ["TRTC"]
```

- 端末でPodfileのあるディレクトリ下に入り、 `pod install` を実行します。次をご参照ください：

```
pod install
```

ステップ2：権限の設定および難読化ルール

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してください。

```
<key>NSMicrophoneUsageDescription</key>
<string>VoiceRoomAppはマイクへのアクセス権限が必要です。有効にした後にレコーディングしたビデオでなければ音声は出ません</string>
```

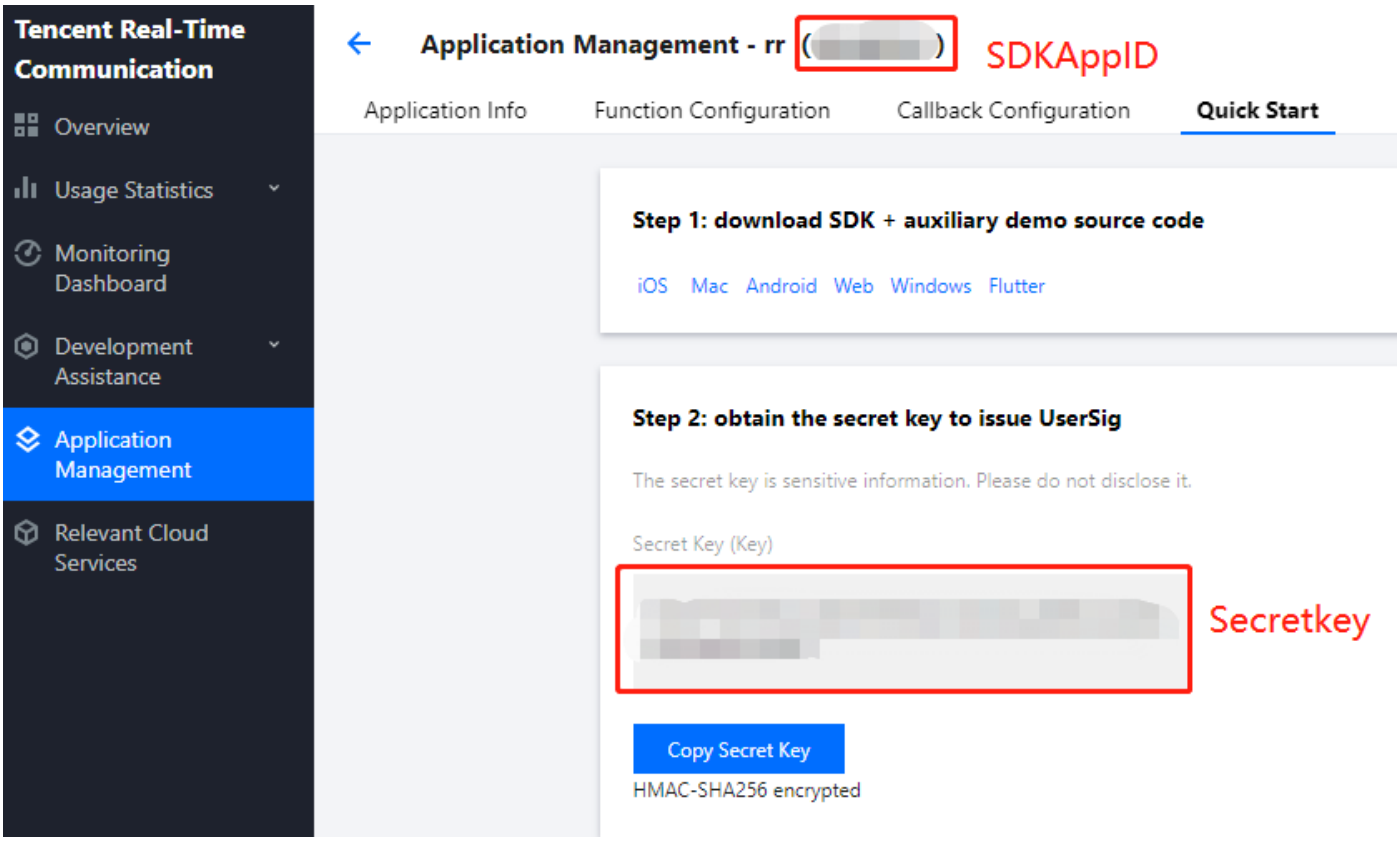
ステップ3：初期化およびログイン

```
// 初期化
let mTRTCVoiceRoom = TRTCVoiceRoom.shared()
// ログイン
mTRTCVoiceRoom.login(sdkAppID: SDKAppID, userId: userId, userSig: userSig) { cod
e, message in
if code == 0 {
//ログイン成功
}
}
```

©2013-2022 Tencent Cloud. All rights reserved.

パラメータの説明：

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション情報**をクリックすると、SDKAppID情報が次の図のように表示されます：



- **Secretkey**：**TRTC アプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます。
- **userId**：現在のユーザーID。文字列タイプであり、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**：SDKAppId、userId，Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のuserSigがオンラインで直接生成されます。UserSigの計算、使用方法をご参照ください。

ステップ4：ボイスチャットルームの実装

1. **管理者によるボイスチャットルーム作成の実装TRTCVoiceRoom#createRoom**

```
// ボイスチャットルームパラメータの初期化
let roomParam = VoiceRoomParam()
roomParam.roomName = "ルーム名"
```

```
roomParam.needRequest = false // リスナーのマイク・オンに対する管理者の同意の要否
roomParam.coverUrl = "ルームカバー図のURL"
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.seatInfoList = []
// マイク情報の初期化
for _ in 0..< param.seatCount {
let seatInfo = VoiceRoomSeatInfo()
param.seatInfoList.append(seatInfo)
}
// 管理者側でルームを作成
mTRTCVoiceRoom.createRoom(roomID: yourRoomID, roomParam: roomParam) { (code, message) in
if code == 0 {
// 作成に成功
}
}
```

2. リスナーのボイスチャットルーム入室の実装TRTCVoiceRoom#enterRoom

```
// 1.リスナーが呼び出して入室
mTRTCVoiceRoom.enterRoom(roomID: roomID) { (code, message) in
// 入室結果コールバック
if code == 0 {
// 入室に成功
}
}
```

3. リスナーの自主的なマイク・オンの実装TRTCVoiceRoom#enterRoom

```
// 1: リスナーが呼び出してマイク・オン
let seatIndex = 2; //マイクのindex
mTRTCVoiceRoom.enterSeat(seatIndex: 2) { (code, message) in
if code == 0 {
// マイク・オン成功
}
}
// 2. onSeatListChangeコールバックを受信し、マイクリストを更新します
@Override
func onSeatListChange(seatInfoList: [VoiceRoomSeatInfo]) {
// 更新したマイクリスト
}
```

©2013-2022 Tencent Cloud. All rights reserved.

4. 管理者による視聴者発言招待の実装TRTCVoiceRoom#pickSeat

```
// 1: 管理者が呼び出して、視聴者が発言できるように招待
let seatIndex = 2; //マイクのindex
let userId = "123"; //マイク・オンが必要なユーザーid
mTRTCVoiceRoom.pickSeat(seatIndex: 1, userId: "123") { (code, message) in
if code == 0 {
}
}
// 2. onSeatListChangeコールバックを受信し、マイクリストを更新します
func onSeatListChange(seatInfoList: [VoiceRoomSeatInfo]) {
// 更新したマイクリスト
}
```

5. リスナーによるマイク・オン申請の実装 TRTCVoiceRoom#sendInvitation

```
// リスナー側の視点
// 1.リスナーが呼び出してマイク・オンを申請
let seatIndex = "1"; //マイクのindex
let userId = "123"; //ユーザーid
let inviteId = mTRTCVoiceRoom.sendInvitation(cmd: "takeSeat", userId: ownerUser
Id, 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
func onInviteeAccepted(identifier: String, invitee: String) {
if identifier == selfID {
self.mTRTCVoiceRoom.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, c
ontent: String) {
if cmd == "takeSeat" {
// 2.管理者がリスナーのリクエストに同意します
self.mTRTCVoiceRoom.acceptInvitation(identifier: identifier, callback: nil)
}
}
```

6. 管理者によるマイク・オンへの招待の実装 TRTCVoiceRoom#sendInvitation

©2013-2022 Tencent Cloud. All rights reserved.

```
// 管理者側の視点
// 1.管理者はsendInvitationを呼び出して、リスナー「123」をピックして2号マイクのマイク・オンをリクエストします
let inviteId = self.mTRTCVoiceRoom.sendInvitation(cmd: "pickSeat", userId: ownerUserId, content: "2") { (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
func onInviteeAccepted(identifier: String, invitee: String) {
if identifier == selfID {
self.mTRTCVoiceRoom.pickSeat(seatIndex: ) { (code, message) in
// マイク・オン結果のコールバック
}
}
}
// リスナー側の視点
// 1.リスナーがリクエストを受信します
func onReceiveNewInvitation(identifier: String, inviter: String, cmd: String, content: String) {
if cmd == "pickSeat" {
// 2.リスナーが管理者のリクエストに同意します
self.mTRTCVoiceRoom.acceptInvitation(identifier: identifier, callback: nil)
}
}
```

7. テキストチャットの実装 TRTCVoiceRoom#sendRoomTextMsg

```
// 発信側：テキストメッセージの発信
self.mTRTCVoiceRoom.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
}
```

8. 弾幕コメントの実装 TRTCVoiceRoom#sendRoomCustomMsg

```
// 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.mTRTCVoiceRoom.sendRoomCustomMsg(cmd: "CMD_DANMU", message: "hello world", callback: nil)
self.mTRTCVoiceRoom.sendRoomCustomMsg(cmd: "CMD_LIKE", message: "", callback: n
```

©2013-2022 Tencent Cloud. All rights reserved.

```
il)
// 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: VoiceRoomUserI
nfo) {
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
}
}
```

よくあるご質問

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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIVoiceRoom API照会
TRTCVoiceRoom (iOS)

最終更新日：：2022-07-19 16:21:32

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブにします。

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

©2013-2022 Tencent Cloud. All rights reserved.

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。
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ります（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ります（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）。

©2013-2022 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。
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

©2013-2022 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。

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

©2013-2022 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ります/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を受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

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を受信。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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
*
* - 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
```

©2013-2022 Tencent Cloud. All rights reserved.

```
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:));
```

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 に送信します。

```
/**
* 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
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:));
```

©2013-2022 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
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	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、使用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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
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:(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です。ご自身でアサインし、一元管理してください。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してください。
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

©2013-2022 Tencent Cloud. All rights reserved.

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	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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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、この関数は無視できます。

```
- (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<NSNumber>	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

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userIdList	NSArray<NSString>	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。

```
- (void)enterSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callbac
k 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：

©2013-2022 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。)

```
- (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
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ります（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:));
```

©2013-2022 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、`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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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：
> 任意のマイクをミュート/ミュート解除します。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:(ActionCallbac
k _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/ミュート解除にします。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。

©2013-2022 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。

```
- (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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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

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(setAudioCaptureVolu
me(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

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

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

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。

```
- (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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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	コマンドワードです。開発者がカスタマイズするものであり、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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

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

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。

©2013-2022 Tencent Cloud. All rights reserved.

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

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

©2013-2022 Tencent Cloud. All rights reserved.

ユーザーのマイクがミュートになっているかどうかのコールバックです。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すると、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	ユーザーリスト。
totalVolume	NSInteger	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です。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

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
seatInfoList	NSArray<VoiceRoomSeatInfo>	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。
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：

©2013-2022 Tencent Cloud. All rights reserved.

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:));
```

パラ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:));
```

©2013-2022 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	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を受信します。

```
- (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を受信します。

```
- (void)onRecvRoomCustomMsg:(NSString *)command
message:(NSString *)message
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	コマンドワードです。開発者がカスタマイズするものであり、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。

©2013-2022 Tencent Cloud. All rights reserved.

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	開発者がカスタマイズする業務指定のコマンドワードです。
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:));
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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。

©2013-2022 Tencent Cloud. All rights reserved.

TRTCVoiceRoom (Android)

最終更新日：：2022-07-19 16:21:32

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブにします。

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

©2013-2022 Tencent Cloud. All rights reserved.

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。
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）。

©2013-2022 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。
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

©2013-2022 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。

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

©2013-2022 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ります/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を受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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してください。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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	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。文字列タイプであり、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、使用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。

©2013-2022 Tencent Cloud. All rights reserved.

setSelfProfile

個人情報の修正。

```
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です。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	ルーム情報です。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してください。
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：

©2013-2022 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。

©2013-2022 Tencent Cloud. All rights reserved.

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。
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、この関数は無視できます。

```
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<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。

getUserInfoList

指定されたuserIdのユーザー情報を取得します。

©2013-2022 Tencent Cloud. All rights reserved.

```
public abstract void getUserInfoList(List<String> userIdList, TRTCVoiceRoomCallback.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<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。
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)。

©2013-2022 Tencent Cloud. All rights reserved.

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。)

```
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ります（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);
```

©2013-2022 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、`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。

©2013-2022 Tencent Cloud. All rights reserved.

```
public abstract void kickSeat(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	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, TRTCVoiceRoomCallbac
k.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：該当するマイクをミュートします；false：該当するマイクをミュート解除します。
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/ミュート解除します。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。

©2013-2022 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, 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：該当するマイクをクローズします； false：該当するマイクをクローズ解除します。
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);
```

©2013-2022 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

©2013-2022 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：

©2013-2022 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

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。

```
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.Action
Callback 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, TRTCVoiceRoomC
allback.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

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
cmd	String	コマンドワードです。開発者がカスタマイズするものであり、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.ActionCall
back callback);
```

パラメータは下表に示すとおりです：

©2013-2022 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

©2013-2022 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。

©2013-2022 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(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

©2013-2022 Tencent Cloud. All rights reserved.

音量レベルリマインダを有効にすると、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	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です。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<SeatInfo>	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）。

©2013-2022 Tencent Cloud. All rights reserved.

```
void onAnchorLeaveSeat(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	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。

©2013-2022 Tencent Cloud. All rights reserved.

```
void onAudienceEnter(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	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を受信します。

```
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を受信します。

```
void onRecvRoomCustomMsg(String cmd, String message, TRTCVoiceRoomDef.UserInfo us
erInfo);
```

©2013-2022 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
cmd	String	コマンドワードです。開発者がカスタマイズするものであり、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	開発者がカスタマイズする業務指定のコマンドワードです。
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。

©2013-2022 Tencent Cloud. All rights reserved.

onInviteeRejected

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>		

©2013-2022 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
TUILiveRoom (Android)の統合

最終更新日：：2022-09-06 14:28:21

コンポーネントの説明

TUILiveRoomはオープンソースのビデオライブストリーミングUIコンポーネントであり、プロジェクトにTUILiveRoomコンポーネントを統合することにより、数行のコードを書くだけで、Appに「ビデオインタラクティブストリーミング」のシーンを組み込むことができます。TUILiveRoomにはAndroid、iOS、ミニプログラムなどのプラットフォーム用のソースコードが含まれます。基本機能は下図のとおりです：

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

©2013-2022 Tencent Cloud. All rights reserved.



コンポーネントの統合

ステップ1：TUILiveRoomコンポーネントのダウンロードとインポート

クリックしてGithubに進み、コードのクローン/ダウンロードを選択した後、`Android/debug`、`Android/tuiaudioeffect`、`Android/tuibarrage`、`Android/tuibeauty`、`Android/tuigift` と `Android/tuiliveroom` ディレクトリをプロジェクトにコピーし、次のようにインポート動作を完了します。

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

- `app` の `build.gradle` ファイルに `tuiliveroom` に対する依存関係を追加します：

©2013-2022 Tencent Cloud. All rights reserved.

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

- ルートディレクトリの `build.gradle` ファイルに `TRTC SDK` および `IM SDK` の依存関係を追加します：

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

ステップ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してください）：

```
<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：コンポーネントの初期化およびログイン

```
// 1.イベント監視およびログインを追加します
TUILogin.addLoginListener(new TUILoginListener() {
@Override
public void onConnecting() { // 接続中
super.onConnecting();
}
```

©2013-2022 Tencent Cloud. All rights reserved.

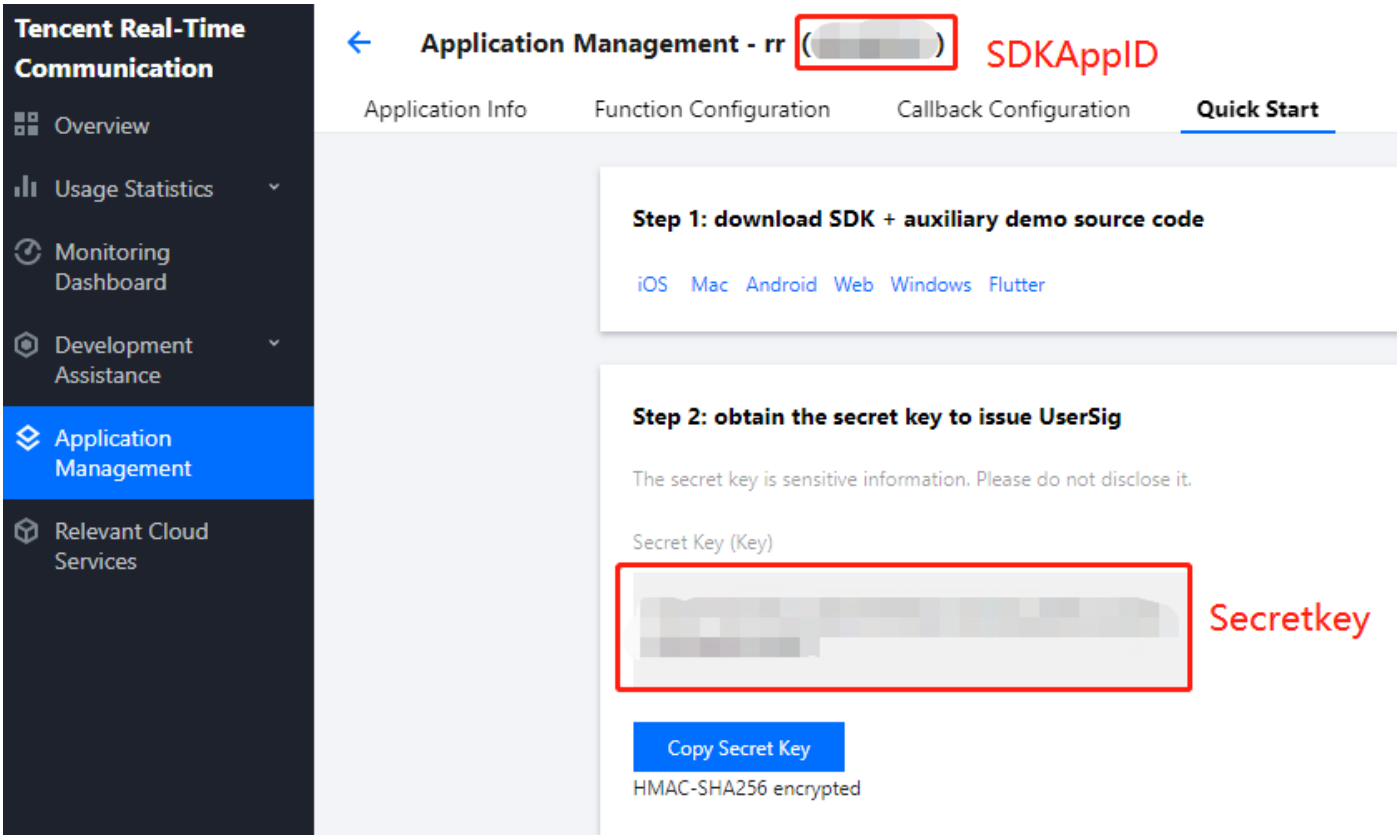
```
@Override
public void onConnectSuccess() { // 接続成功通知
super.onConnectSuccess();
}
@Override
public void onConnectFailed(int errorCode, String errorMsg) { // 接続失敗通知
super.onConnectFailed(errorCode, errorMsg);
}
@Override
public void onKickedOffline() { // ログインがキックアウトされたオフライン通知（例：アカウントが他のデバイスでログインしている）
super.onKickedOffline();
}
@Override
public void onUserSigExpired() { // userSig期限切れ通知
super.onUserSigExpired();
}
});
TUILogin.login(mContext, "Your SDKAppID", "Your userId", "Your userSig", new TUIC
allback() {
@Override
public void onSuccess() {
}
@Override
public void onError(int errorCode, String errorMsg) {
Log.d(TAG, "errorCode: " + errorCode + " errorMsg:" + errorMsg);
}
});
// 2.TUILiveRoomコンポーネントの初期化
TUILiveRoom mLiveRoom = TUILiveRoom.sharedInstance(mContext);
```

パラメータの説明：

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション**

©2013-2022 Tencent Cloud. All rights reserved.

情報をクリックすると、SDKAppID情報が次の図のように表示されます：



- **Secretkey**：**TRTC アプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます。
- **userId**：現在のユーザーID。文字列タイプであり、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**：SDKAppId、userId、Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のuserSigがオンラインで直接生成されます。UserSigの計算、使用方法をご参照ください。

ステップ4：ビデオインタラクティブストリーミングルームの実装

1. キャスター側の配信開始

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

2. 視聴者側の視聴

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

©2013-2022 Tencent Cloud. All rights reserved.

3. 視聴者とキャスターがマイク接続 TRTCLiveRoom#requestJoinAnchor

```
// 1.視聴者側がマイク接続のリクエストを送信します
// LINK_MIC_TIMEOUTはタイムアウト時間です
TRTCLiveRoom mTRTCLiveRoom=TRTCLiveRoom.sharedInstance(mContext);
mTRTCLiveRoom.requestJoinAnchor(mSelfUserId + "あなたとのマイク接続をリクエスト", LI
NK_MIC_TIMEOUT
new TRTCLiveRoomCallback.ActionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
// 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);
// 視聴者がプレビューを起動し、プッシュを開始します
mTRTCLiveRoom.startCameraPreview(true, view, null);
mTRTCLiveRoom.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
mTRTCLiveRoom.setDelegate(new TRTCLiveRoomDelegate() {
@Override
public void onRequestJoinAnchor(final TRTCLiveRoomDef.TRTCLiveUserInfo userInf
o,
String reason, final int timeout) {
// 相手のマイク接続のリクエストに同意します
mTRTCLiveRoom.responseJoinAnchor(userInfo.userId, true, "マイク接続に同意");
}
@Override
public void onAnchorEnter(final String userId) {
// 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);
// キャスターが視聴者の画面を再生します
mTRTCLiveRoom.startPlay(userId, view, null);
}
});
```

4. キャスター間のPK TRTCLiveRoom#requestRoomPK

```
// キャスターAがルーム12345を作成します
mLiveRoom.createRoom(12345, "roomA", "Your coverUrl");
// キャスターBがルーム54321を作成します
mLiveRoom.createRoom(54321, "roomB", "Your coverUrl");
```

©2013-2022 Tencent Cloud. All rights reserved.

```
// キャスターA:
TRTCLiveRoom mTRTCLiveRoom=TRTCLiveRoom.sharedInstance(mContext);
// 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
mTRTCLiveRoom.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
}else{
// ユーザー拒否
}
}
});
mTRTCLiveRoom.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
mTRTCLiveRoom.startPlay(userId, mTXCloudVideoView, null);
}
});
// キャスターB：
// 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
mTRTCLiveRoom.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
mTRTCLiveRoom.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
mTRTCLiveRoom.startPlay(userId, mTXCloudVideoView, null);
}
});
```

よくあるご質問

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。

©2013-2022 Tencent Cloud. All rights reserved.

TUILiveRoom (iOS)の統合

最終更新日：：2022-08-15 11:50:53

コンポーネントの説明

TUILiveRoomはオープンソースのビデオライブストリーミングUIコンポーネントであり、プロジェクトにTUILiveRoomコンポーネントを統合することにより、数行のコードを書くだけで、Appに「ビデオインタラクティブストリーミング」のシーンを組み込むことができます。TUILiveRoomにはAndroid、iOSなどのプラットフォーム用のソースコードが含まれます。基本機能は下図のとおりです：

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。



©2013-2022 Tencent Cloud. All rights reserved.

コンポーネントの統合

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. プロジェクトの `Podfile` ファイルと同じ階層のディレクトリ下に `TUILiveRoom` フォルダを作成します。
2. クリックして**Github/TUILiveRoom**に進み、コードのクローン/ダウンロードを選択した後、**TUILiveRoom/iOS/**ディレクトリ下の `Source` 、 `Resources` 、 `TUIBeauty` 、 `TUIAudioEffect` 、 `TUIBarrage` 、 `TUIGift` 、 `TXAppBasic` フォルダと `TUILiveRoom.podspec` ファイルを、 `ステップ1` で作成したTUILiveRoomフォルダ下にコピーします。
3. 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。

```
# :path => "TUILiveRoom.podspecを指定する相対パス"
pod 'TUILiveRoom', :path => "./TUILiveRoom/TUILiveRoom.podspec", :subspecs => [
"TRTC"]
# :path => "TXAppBasic.podspecを指定する相対パス"
pod 'TXAppBasic', :path => "./TUILiveRoom/TXAppBasic/"
# :path => "TUIBeauty.podspecを指定する相対パス"
pod 'TUIBeauty', :path => "./TUILiveRoom/TUIBeauty/"
# :path => "TUIAudioEffect.podspecを指定する相対パス"
pod 'TUIAudioEffect', :path => "./TUILiveRoom/TUIAudioEffect/"
# :path => "TUIBarrage.podspecを指定する相対パス"
pod 'TUIBarrage', :path => "./TUILiveRoom/TUIBarrage/"
# :path => "TUIGift.podspecを指定する相対パス"
pod 'TUIGift', :path => "./TUILiveRoom/TUIGift/"
```

注意：

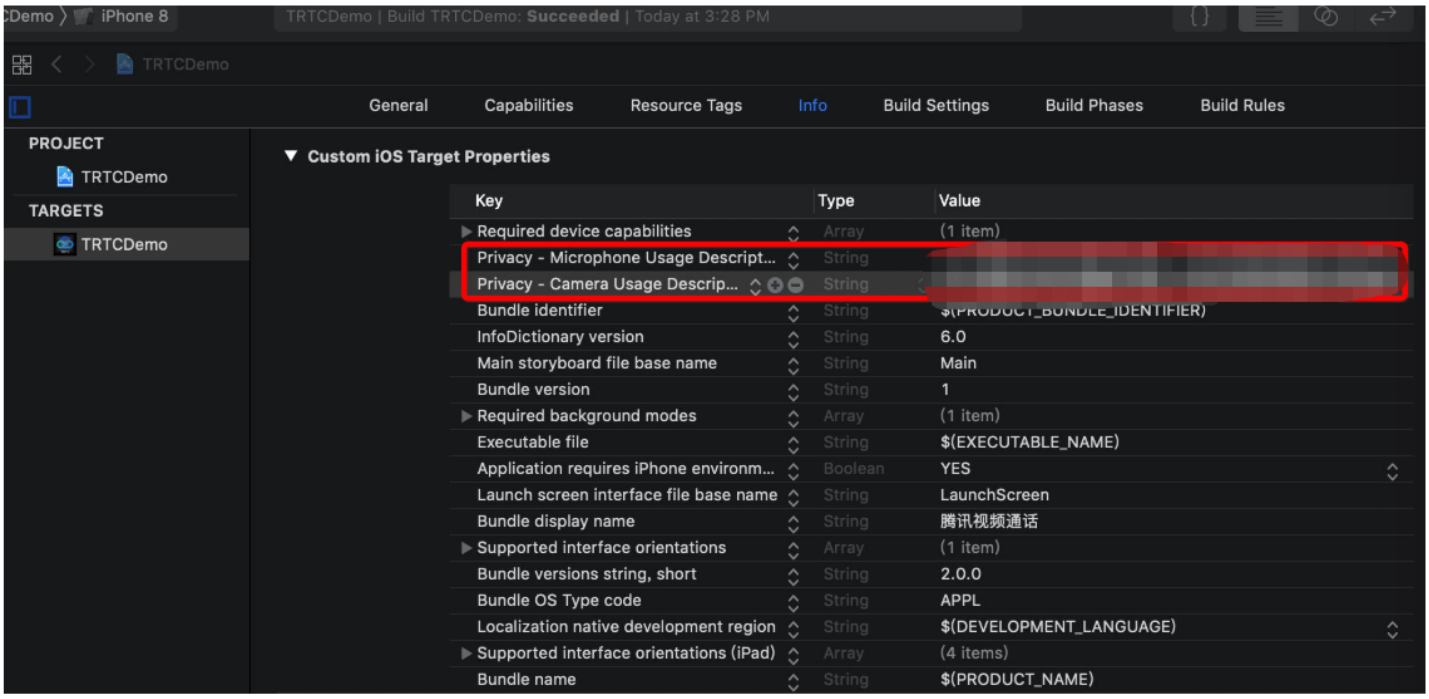
- `Source` 、 `Resources` フォルダと `TUILiveRoom.podspec` ファイルは同一のディレクトリ下にある必要があります。
- TXAppBasic.podspecはTXAppBasicフォルダ下にあります。

ステップ2：権限の設定

オーディオビデオ機能を使用するには、マイクおよびカメラの使用権限を付与する必要があります。AppのInfo.plistで以下の2項を追加します。これらはシステムが権限付与ダイアログボックスをポップアップする際に表示される、マイクとカメラのメッセージにそれぞれ対応します。

©2013-2022 Tencent Cloud. All rights reserved.

```
<key>NSCameraUsageDescription</key>
<string>RoomAppはカメラへのアクセス権限が必要です。有効にした後にレコーディングしたビデオでなければ画面は出ません</string>
<key>NSMicrophoneUsageDescription</key>
<string>RoomAppはマイクへのアクセス権限が必要です。有効にした後にレコーディングしたビデオでなければ音声は出ません</string>
```



ステップ3：コンポーネントの初期化およびログイン

```
@import TUILiveRoom;
@import TUICore;

// 1.コンポーネントのログイン
[TUILogin login:@"あなたのSDKAppID" userID:@あなたのUserID" userSig:@"あなたのUserSig" succ:^{

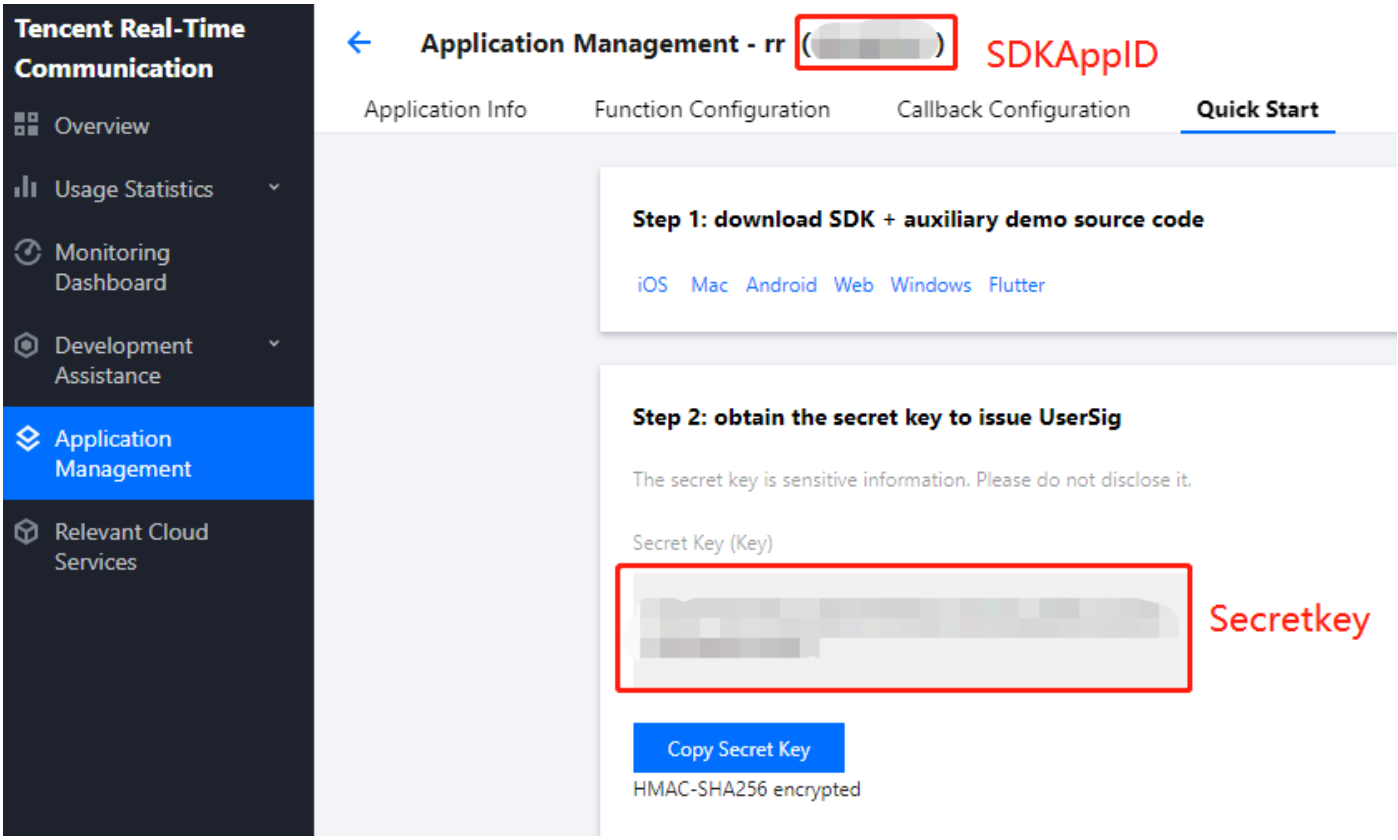
} fail:^(int code, NSString *msg) {

}];
// 2.TUILiveRoomインスタンスの初期化
TUILiveRoom *mLiveRoom = [TUILiveRoom sharedInstance];
<dx-code-holder data-codeindex="2"></dx-code-holder>
```

パラメータの説明：

©2013-2022 Tencent Cloud. All rights reserved.

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション情報**をクリックすると、SDKAppID情報が次の図のように表示されます：



- **SecretKey**：**TRTC アプリケーションキー**であり、SDKAppIDに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます。
- **UserID**：現在のユーザーのIDです。文字列形式で、長さは32バイト以内とし、特殊文字の使用はサポートしていません。英語または数字の使用をお勧めします。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ができます。
- **UserSig**：SDKAppId、UserID、SecretKeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のUserSigがオンラインで直接生成されます。また当社のTUILiveRoomデモプロジェクトを参照してご自身で計算することもできます。その他の情報については、UserSigの計算、使用方法をご参照ください。

ステップ4：ビデオインタラクティブストリーミングルームの実装

1. **キャスター側の配信開始**

   - Objective-C Objc
   - Swift Swift

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

©2013-2022 Tencent Cloud. All rights reserved.

2. 視聴者側の視聴

- Objective-C Objc
- Swift Swift

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

3. 視聴者とキャスターがマイク接続 TRTCLiveRoom#requestJoinAnchor

- Objective-C Objc
- Swift Swift

```
// 1.視聴者側がマイク接続のリクエストを送信します
[TRTCLiveRoom shareInstance].delegate = self;
// @param mSelfUserId String 現在のユーザーid
NSString *mSelfUserId = @"1314";
[[TRTCLiveRoom shareInstance] requestJoinAnchor:[NSString stringWithFormat:@"%@ マイク接続をリクエスト", mSelfUserId] timeout:30 responseCallback:^(BOOL agreed, NSString * _Nullable reason) {
if (agreed) {
// キャスターが視聴者のリクエストを受け入れます
UIView *playView = [UIView new];
[self.view addSubView:playView];
// 視聴者がプレビューを起動し、プッシュを開始します
[[TRTCLiveRoom shareInstance] startCameraPreviewWithFrontCamera:YES view:playView callback:nil];
[[TRTCLiveRoom shareInstance] startPublishWithStreamID:[NSString stringWithFormat:@"%@_stream", mSelfUserId] callback:nil];
}
}];

// 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
#pragma mark - TRTCLiveRoomDelegate
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onRequestJoinAnchor:(TRTCLiveUserInfo *)user reason:(NSString *)reason {
// 相手のマイク接続のリクエストに同意します
[[TRTCLiveRoom shareInstance] responseJoinAnchor:user.userId agree:YES reason:@"マイク接続に同意"];
}

- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onAnchorEnter:(NSString *)userID {
// 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
UIView *playView = [UIView new];
```

©2013-2022 Tencent Cloud. All rights reserved.

```
[self.view addSubview:playView];
// キャスターが視聴者の画面を再生します
[[TRTCLiveRoom shareInstance] startPlayWithUserID:userID view:playView callbac
k:nil];
}
```

4. キャスター間のPK TRTCLiveRoom#requestRoomPK

- Objective-C Objc
- Swift Swift

```
// キャスターAがルーム12345を作成します
[[TUILiveRoom sharedInstance] createRoomWithRoomId:12345 roomName:@"roomA" cov
erUrl:@"roomA coverUrl"];
// キャスターBがルーム54321を作成します
[[TUILiveRoom sharedInstance] createRoomWithRoomId:54321 roomName:@"roomB" cov
erUrl:@"roomB coverUrl"];

// キャスターA
// 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
[[TRTCLiveRoom shareInstance] requestRoomPKWithRoomID:543321 userID:@"roomB us
erId" timeout:30 responseCallback:^(BOOL agreed, NSString * _Nullable reason)
{
if (agreed) {
// ユーザーBが受け入れます
}else{
// ユーザーBが拒否します
}
}];

// キャスターB：
// 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
#pragma mark - TRTCLiveRoomDelegate
- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onRequestRoomPK:(TRTCLiveUse
rInfo *)user {
// 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
[[TRTCLiveRoom shareInstance] responseRoomPKWithUserID:user.userId agree:YES r
eason:@""];
}

- (void)trtcLiveRoom:(TRTCLiveRoom *)trtcLiveRoom onAnchorEnter:(NSString *)us
erID {
// 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
[[TRTCLiveRoom shareInstance] startPlayWithUserID:userID view:playAView callba
ck:nil];
}
```

©2013-2022 Tencent Cloud. All rights reserved.

よくあるご質問

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。

©2013-2022 Tencent Cloud. All rights reserved.

TUILiveRoom (Flutter)の統合

最終更新日：：2022-07-19 16:21:32

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

DemoのUIの再利用

ステップ1：新規アプリケーションの作成

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. `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。

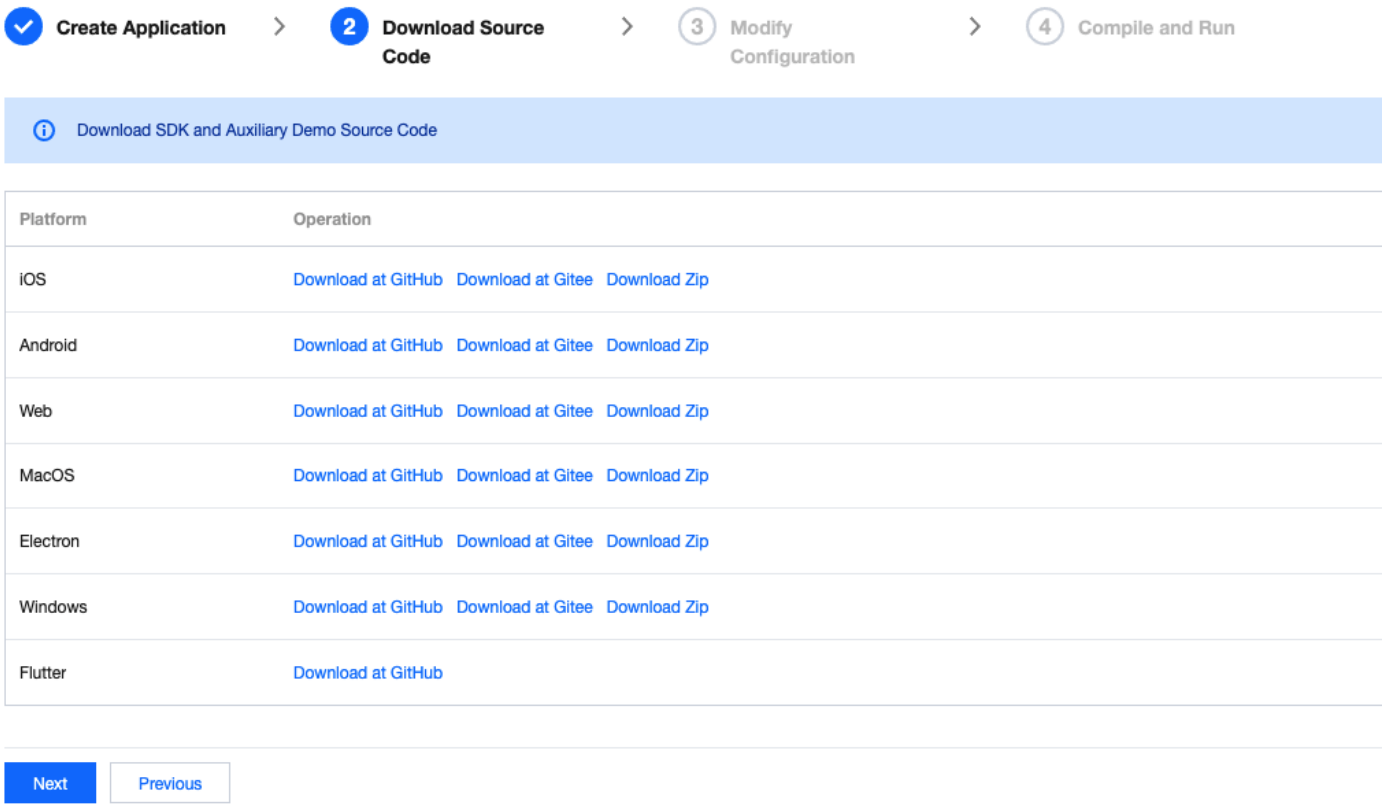
> 注意：
> 本機能はTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期してアクティブにすることができます。IMは付加価値サービスであり、課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。

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。

©2013-2022 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

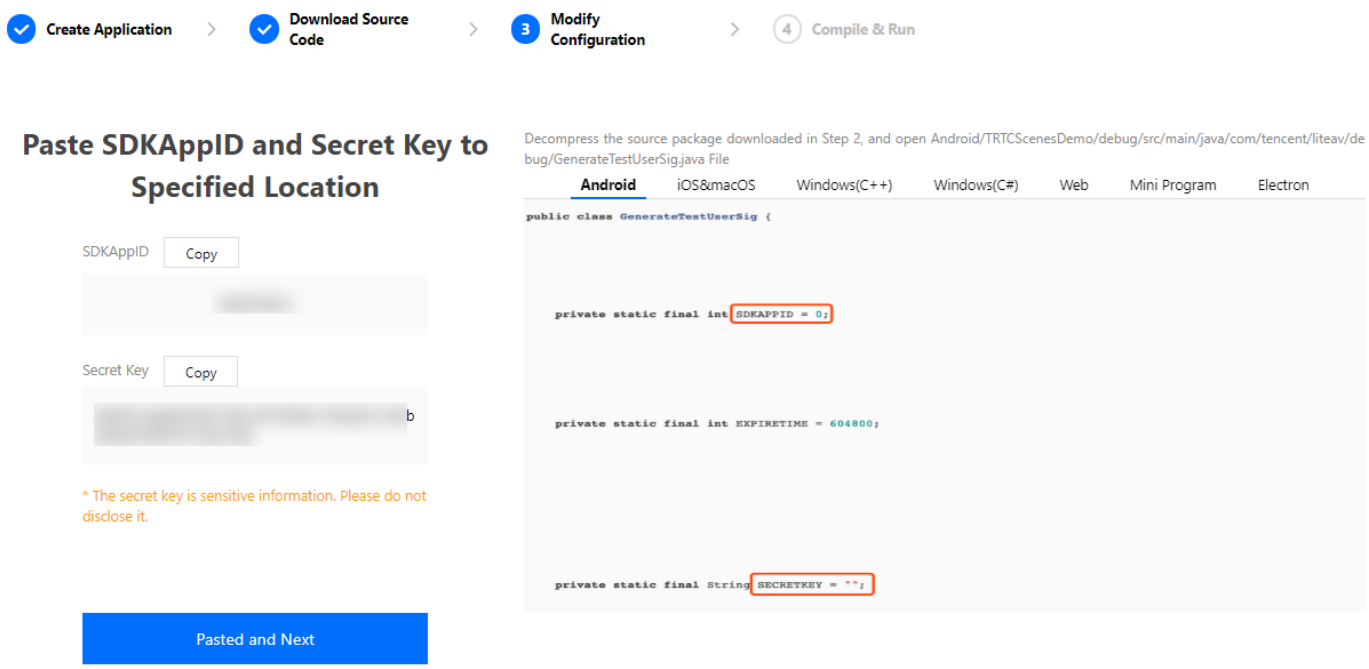
1. 設定変更画面に進み、ダウンロードしたソースコードパッケージに基づき、対応する開発環境を選択します。

2. `/example/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。

©2013-2022 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向けのインターフェースを提供します。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。詳細はサーバーでUserSigを生成するをご参照ください。

ステップ4：コンパイルと実行

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。

©2013-2022 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。

ステップ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。

ステップ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
```

©2013-2022 Tencent Cloud. All rights reserved.

ステップ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してください：

```
<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>
```

ステップ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/
```

ステップ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。文字列タイプであり、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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

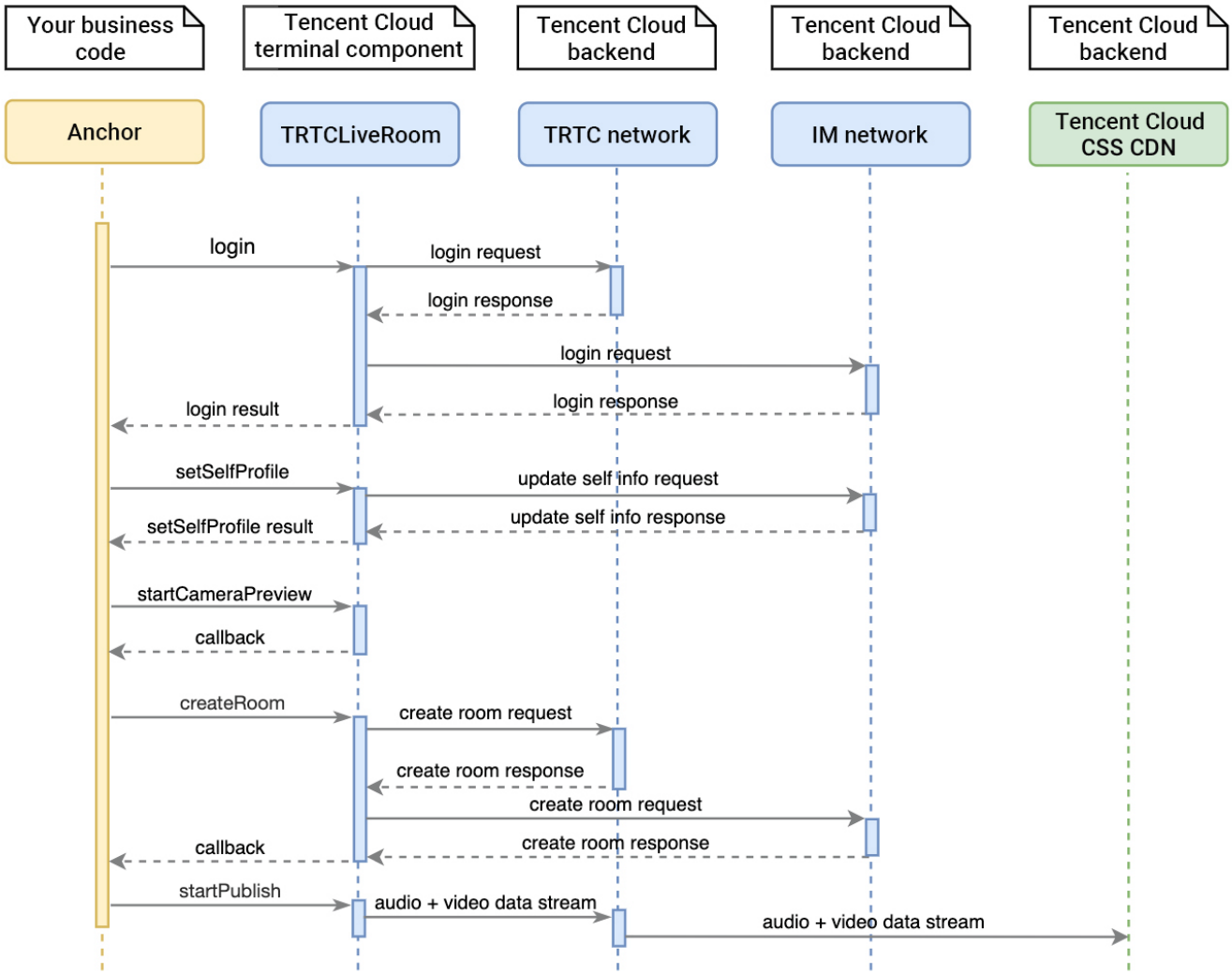
ステップ5：キャスター側での配信開始

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。

©2013-2022 Tencent Cloud. All rights reserved.



ステップ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。

©2013-2022 Tencent Cloud. All rights reserved.

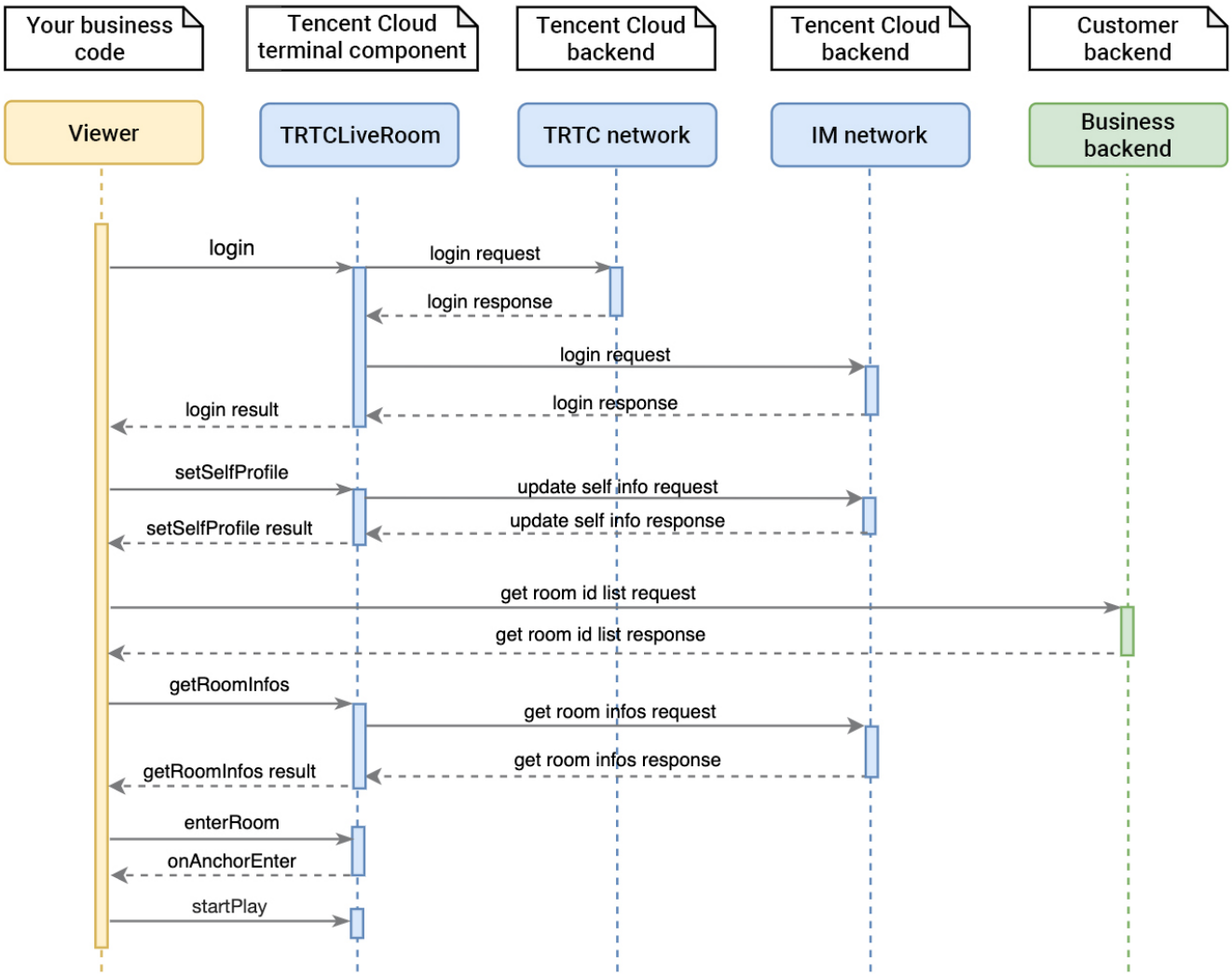
3. 視聴者側は、 `getRoomInfos` を呼び出してルームの詳細情報を取得します。この情報は、キャスター側が `createRoom` を呼び出してライブストリーミングルームを作成するときに設定する簡単な説明情報です。

> 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。

©2013-2022 Tencent Cloud. All rights reserved.



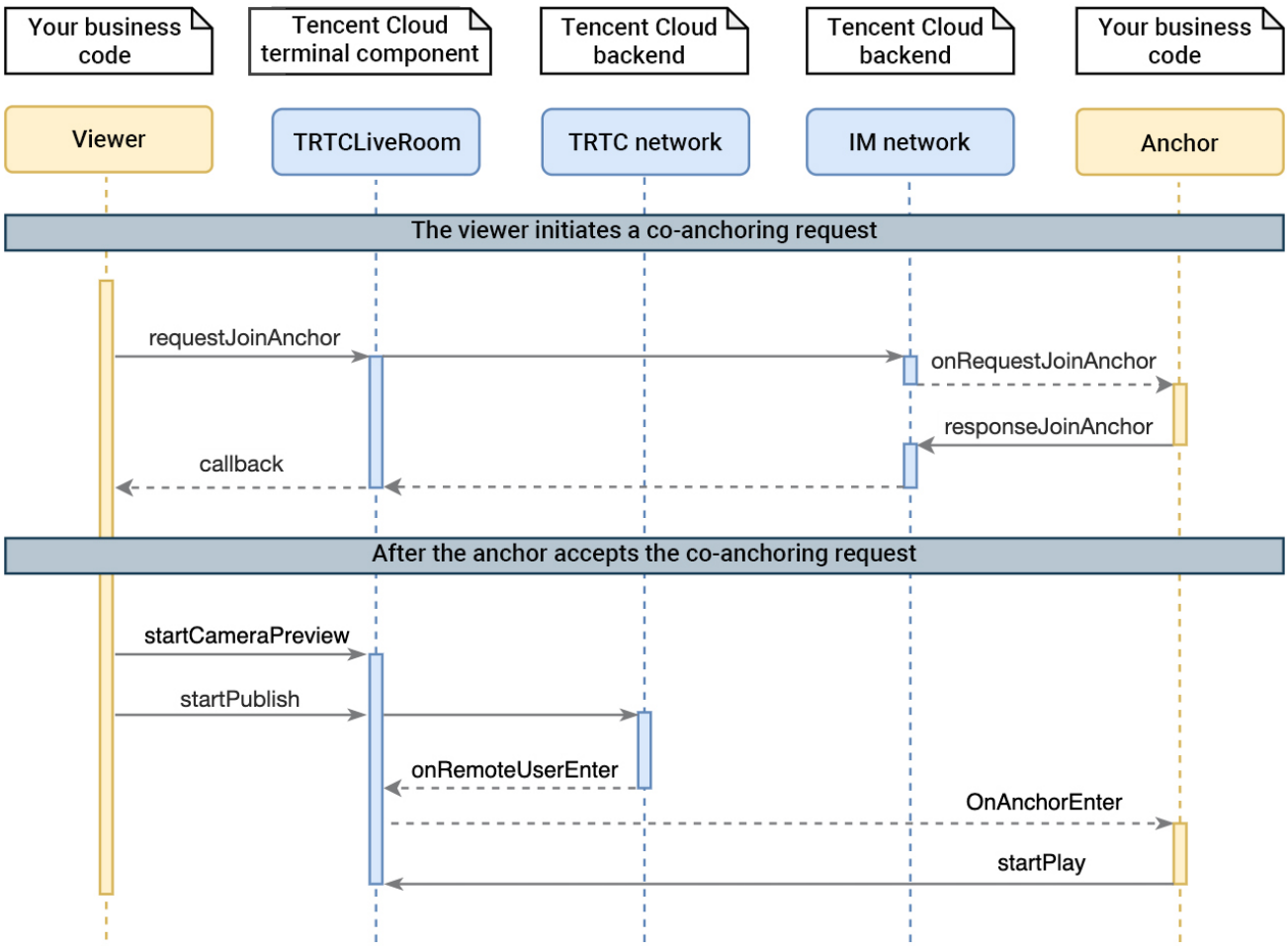
ステップ7：視聴者とキャスターとのマイク接続のオンとオフ

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

©2013-2022 Tencent Cloud. All rights reserved.

います。

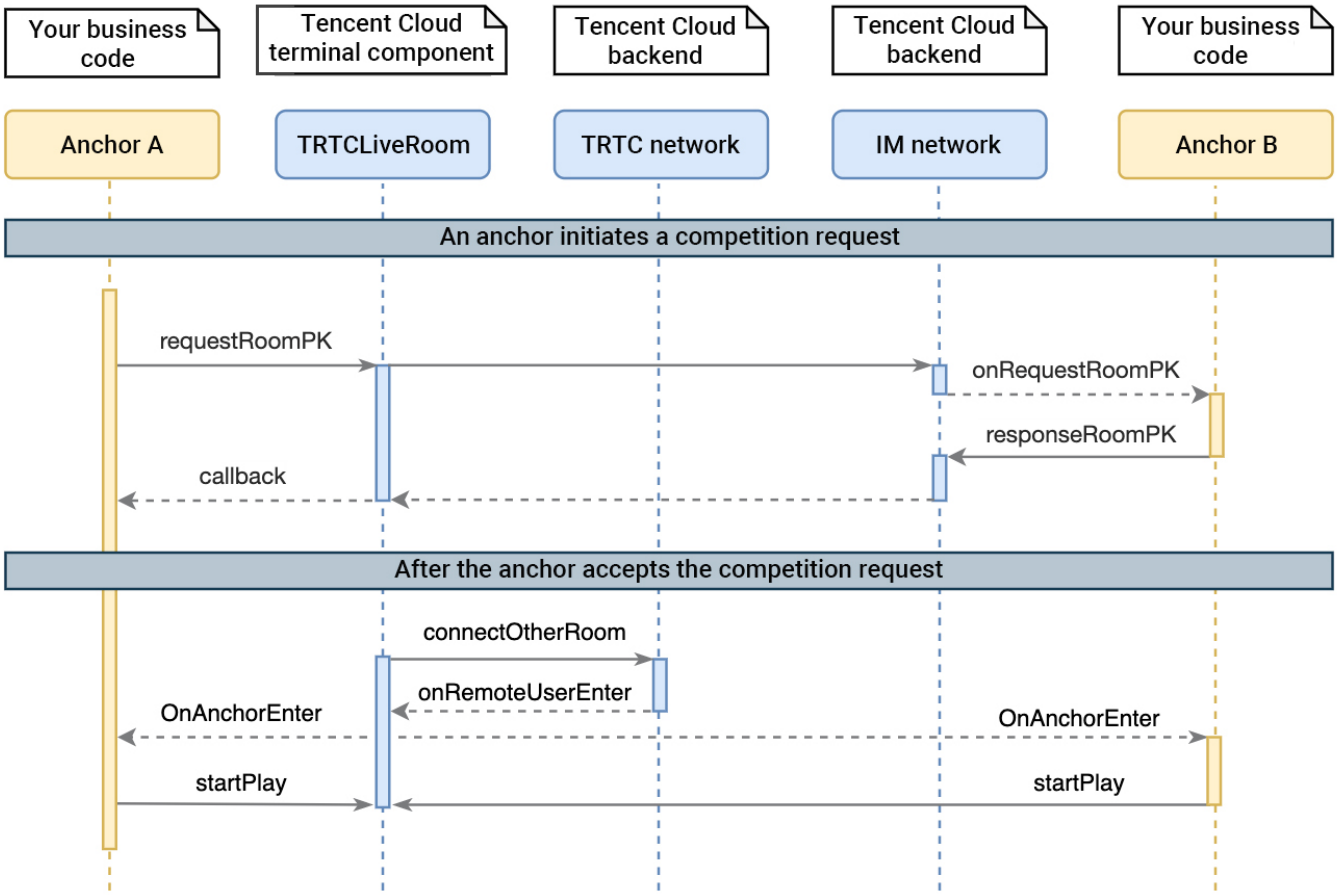
7. キャスター側が `startPlay` を呼び出すと、マイク接続の視聴者の画面を見ることができます。



ステップ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どうかが決定します。
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 。

©2013-2022 Tencent Cloud. All rights reserved.



ステップ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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIPusher&TUIPlayer (Web)の統合

最終更新日：：2022-09-06 14:23:04

コンポーネントの説明

TUIPusher & TUIPlayerは、Web端末オープンソースの、UIを備えたライブストリーミングインタラクションコンポーネントです。TUIPusher & TUIPlayerはTRTC、 IMなどの基本SDKを統合し、企業によるライブストリーミング、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ストリーミングシーン向けに、すぐにリリースできるWeb端末ライブストリーミング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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブ化すると、関連するIM SDKの体験版がデフォルトでアクティブ化されます。これは100 DAUのみをサポートします。

TUIPusher & TUIPlayerのメリット：

- ライブストリーミングシーンのニーズに合わせ、UI付きのライブストリーミングシーン向け汎用ソリューションをご提供します。ライブストリーミング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を直接統合し、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。

©2013-2022 Tencent Cloud. All rights reserved.



クイック体験

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。

TUIPusherプッシュコンポーネントの機能説明

- 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の機能説明

- 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

©2013-2022 Tencent Cloud. All rights reserved.

- 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
- 超低遅延ライブストリーミング（300msの遅延）、ライブイベントストリーミング（1000ms以下の遅延）、標準ライブストリーミング（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端末ブラウザとの互換性あり、モバイル端末ブラウザの横画面表示をサポート

注意：
一部のブラウザではWebRTCがサポートされておらず、標準ライブストリーミング回線を使用した視聴のみ可能です。他の回線の体験をご希望の場合は、ブラウザの変更をお試しください。

コンポーネントの統合

ステップ1：Tencent Cloudサービスのアクティブ化

注意：

- 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。
- 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公開しないでください。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。詳細については、サーバーでのUserSig新規作成をご参照ください。

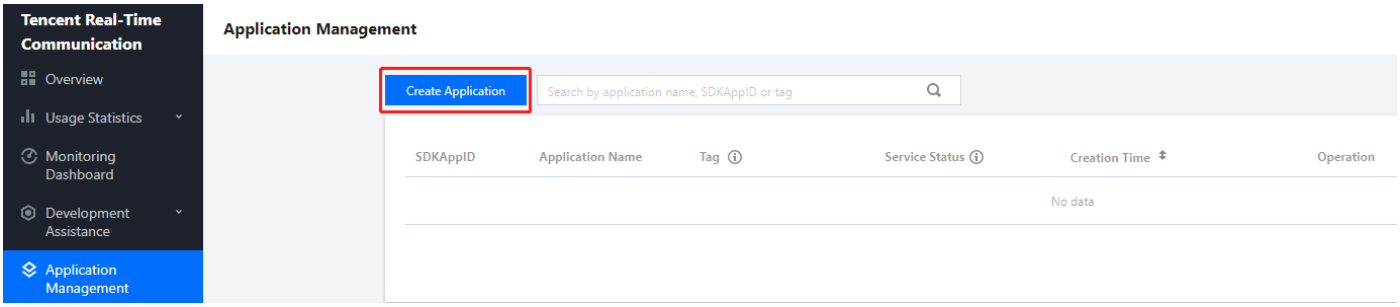
- 方法1：TRTCの場合
- 方法2：インスタントメッセージIMの場合

ステップ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。

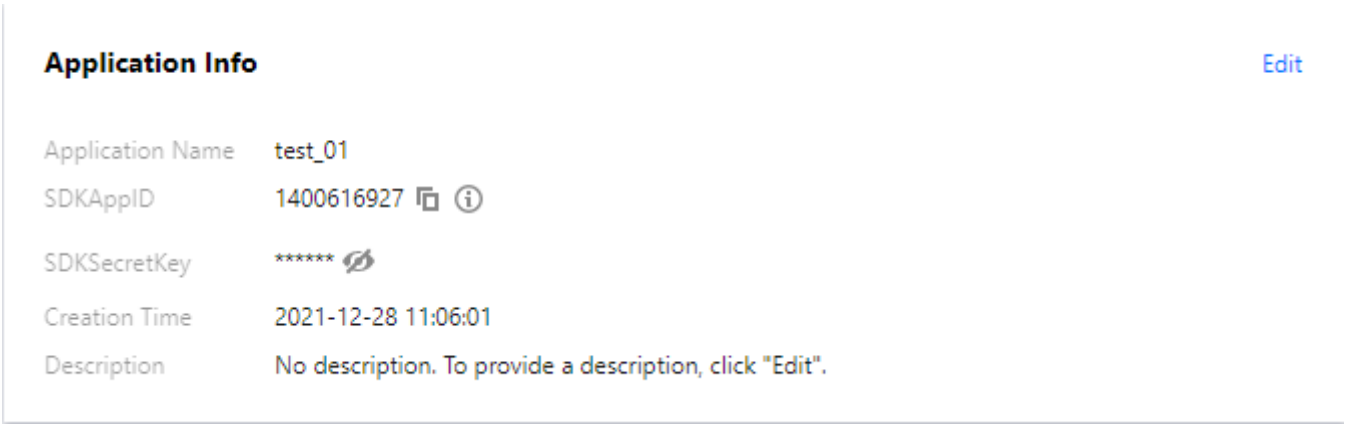
©2013-2022 Tencent Cloud. All rights reserved.

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。

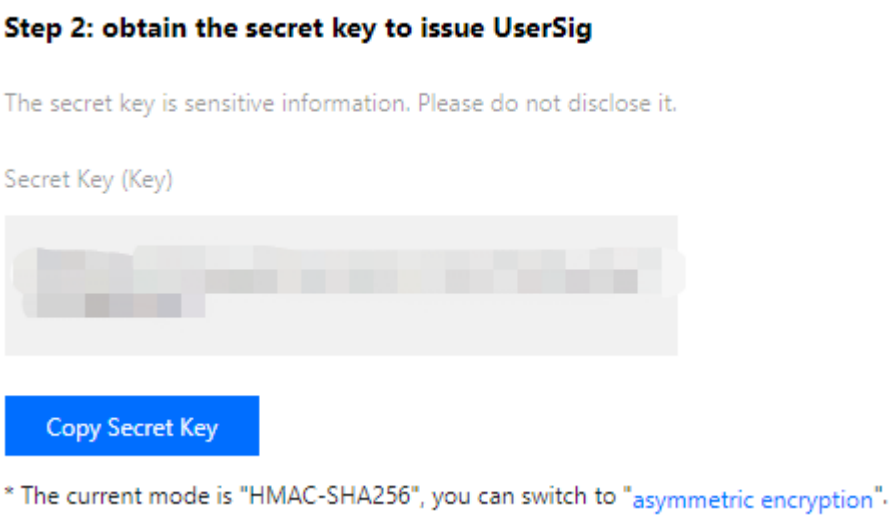


ステップ2： TRTCキー情報の取得

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。



©2013-2022 Tencent Cloud. All rights reserved.

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。

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のルーム情報、キャスター情報を一致させてください**。

注意：

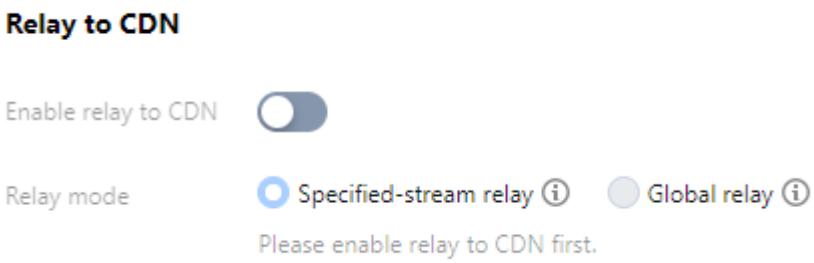
©2013-2022 Tencent Cloud. All rights reserved.

- 上記設定が完了すると、TUIPusher & TUIPlayerを使用して超低遅延ライブストリーミング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ストリーミング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公開しないでください。
- 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。

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ストリーミングは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ストリーミング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ストリーミング、ライブイベントストリーミング、標準ライブストリーミングの全機能を体験できます。

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。

©2013-2022 Tencent Cloud. All rights reserved.

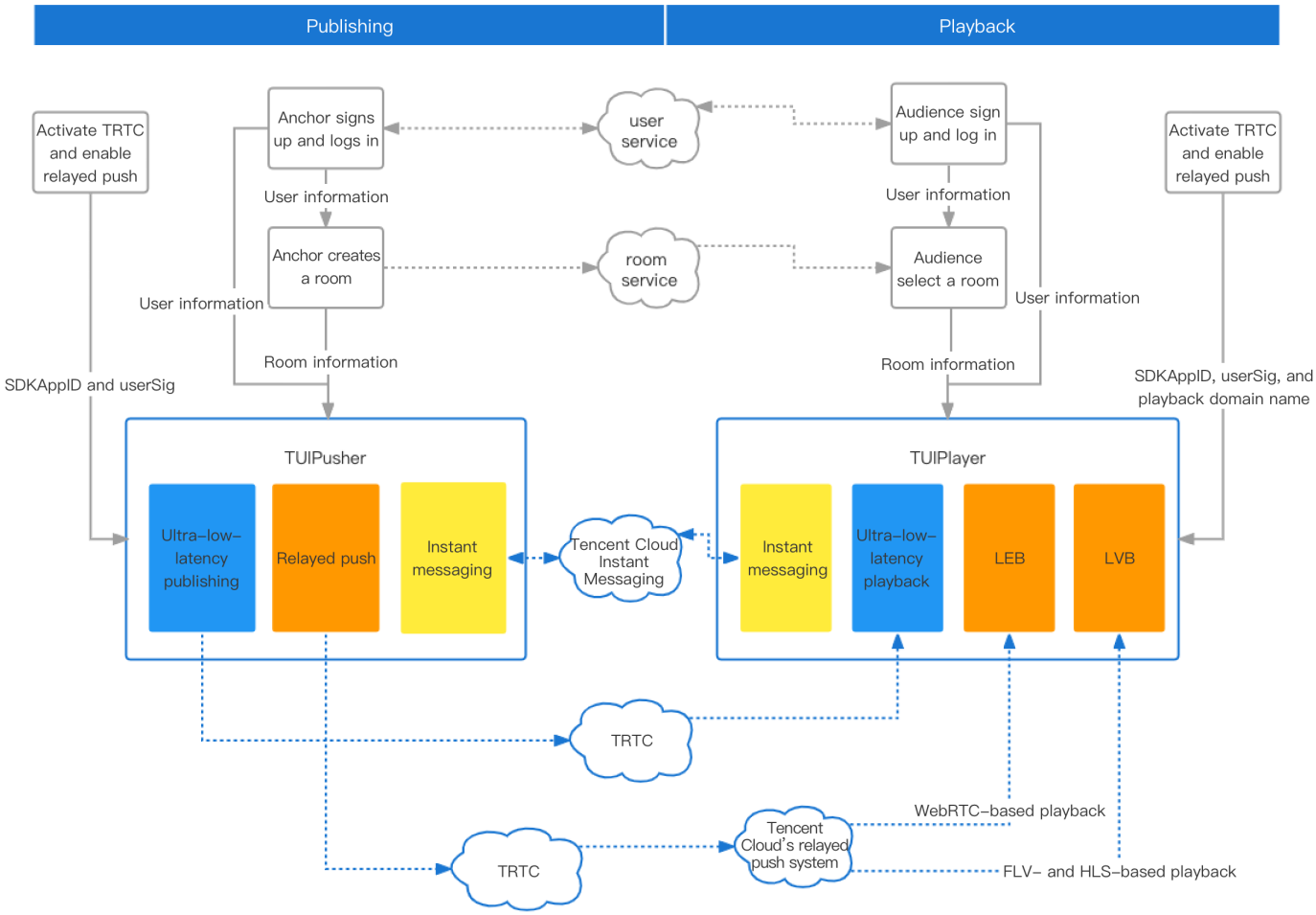
- ユーザーID、ユーザー名、ユーザープロフィール画像などを含む製品ユーザー情報を管理するためのユーザー管理システムを作成します。
- ライブストリーミングルームID、ライブストリーミングルーム名、ライブストリーミングルームのホスト情報などを含む製品のライブストリーミングルーム情報を管理するためのルーム管理システムを作成します。
- サーバーがUserSigを発行します。

> 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。

- `TUIPusher/src/pusher.vue` 、 `TUIPlayer/src/player.vue` ファイルを参照し、ユーザー情報、ライブストリーミングルーム情報、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

©2013-2022 Tencent Cloud. All rights reserved.

することができます。詳細な業務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す。

©2013-2022 Tencent Cloud. All rights reserved.

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

サポートプラットフォーム

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ページ	-	サポートあり	サポートなし

©2013-2022 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
Mac OS	デスクトップ版 WeCom Embedded Webページ	-	サポートあり	サポートなし
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ページ	-	サポートあり	サポートなし
Windows	デスクトップ版 WeCom Embedded Webページ	-	サポートあり	サポートなし
iOS	WeChat Embeddedブラウザ	-	サポートあり	サポートなし
iOS	WeCom Embedded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

©2013-2022 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
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 Embeddedブラウザ	-	サポートあり	サポートなし
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	モバイル版UCブラウザ	-	サポートあり（標準ライブストリーミング視聴のみサポート）	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。

©2013-2022 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	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	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	サポート	サポート	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	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を使用する時、ユーザはファイアーフォールの制限でオーディオビデオ通話を利用できない可能性があります。ファイアウォール制限への対応 を参照し、関連するポートとドメイン名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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

TUILiveRoom APIのクエリー
TRTCLiveRoom API (iOS)

最終更新日：：2022-07-19 16:21:32

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

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。

©2013-2022 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。
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

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

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
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

©2013-2022 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。

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	視聴者入室通知を受信します。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
onAudienceExit	視聴者退室通知を受信します。

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。
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を受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

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。

©2013-2022 Tencent Cloud. All rights reserved.

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.intl.cloud.tencent.com/trtc/app)】> アプリケーション情報でSDKAppIDを確認できます。
/// - userID: 現在のユーザーID。文字列タイプであり、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です。取得方法については、[UserSigの計算と使用方法](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です。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。文字列タイプであり、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、使用方法をご参照ください。

©2013-2022 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、費用は安価ですがディレーは高いです。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。

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
```

©2013-2022 Tencent Cloud. All rights reserved.

```
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

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です。ご自身でアサインし、一元管理してください。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 | ルーム情報であり、ルームの説明情報として利用されます（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できます。
/// - 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に呼び出します。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:));
```

©2013-2022 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	UInt32	ルームIDです。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	ルーム情報であり、ルームの説明情報として利用されます（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できます。
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。

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

©2013-2022 Tencent Cloud. All rights reserved.

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。その中には、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が含まれている場合、enterRoom() 成功後、直接startPlay(userID)を呼び出すと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)を呼び出すと、キャスターの画面を再生できます。
/// - 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に呼び出します
/// - 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を使用してください。
- (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。

©2013-2022 Tencent Cloud. All rights reserved.

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。その中には、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。
/// - Note:
/// - 視聴者がライブストリーミングルームを退出する時に呼び出します
/// - キャスターはルームを退出する際に、このインターフェースを呼び出すことができません。
- (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、この関数は無視できます。

©2013-2022 Tencent Cloud. All rights reserved.

```
/// 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で設定されたものです。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。

```
/// 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：

©2013-2022 Tencent Cloud. All rights reserved.

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:));
```

パラ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呼び出し
- 視聴者のマイク接続開始時の呼び出し

©2013-2022 Tencent Cloud. All rights reserved.

```
/// 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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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が含まれている場合、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) を呼び出すと、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:
/// - 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。

©2013-2022 Tencent Cloud. All rights reserved.

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してください。

```
/// 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:
/// - 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:));
```

©2013-2022 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
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。
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してください。

```
/// 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。
```

©2013-2022 Tencent Cloud. All rights reserved.

```
/// - 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。

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

©2013-2022 Tencent Cloud. All rights reserved.

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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

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：

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

©2013-2022 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。

音声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:));
```

パラ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：ミュートオン、false：ミュートオフ。

©2013-2022 Tencent Cloud. All rights reserved.

muteRemoteAudio

リモートの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：ミュートオン、false：ミュートオフ。

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。

```
/// 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：ミュートオン、false：ミュートオフ。

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：

©2013-2022 Tencent Cloud. All rights reserved.

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。 |

```
/* 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。

©2013-2022 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。

```
/// 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。
callback	(_ code: Int, _ message: String?) -> Void	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。

```
/// カスタムメッセージを送信します
/// - Parameters:
/// - command: コマンドワードです。開発者がカスタマイズするもので、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	コマンドワードです。開発者がカスタマイズするもので、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。

©2013-2022 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	(_ code: Int, _ message: String?) -> Void	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：

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の注意喚起情報を表示します。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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:));
```

パラ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

©2013-2022 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。
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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
- (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

視聴者入室通知を受信します。

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

©2013-2022 Tencent Cloud. All rights reserved.

```
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

視聴者退室通知を受信します。

```
- (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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
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

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してください。

```
- (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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
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を受信します。

```
- (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：

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を受信します。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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	コマンドワードです。開発者がカスタマイズするもので、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。

©2013-2022 Tencent Cloud. All rights reserved.

TRTCLiveRoom API (Android)

最終更新日：：2022-07-19 16:21:32

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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。
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（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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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。

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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。
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

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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
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、使用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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

setSelfProfile

個人情報の修正。

```
public abstract void setSelfProfile(String userName, String avatarURL, 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
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、一元管理してください。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	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できます。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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 ca
llback);
```

パラ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。その中には、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。

©2013-2022 Tencent Cloud. All rights reserved.

- 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。

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、この関数は無視できます。

```
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

©2013-2022 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。

```
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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

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呼び出し
- 視聴者のマイク接続開始時の呼び出し

```
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

©2013-2022 Tencent Cloud. All rights reserved.

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, TRTCLiveRoom
Callback.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。

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してください。

```
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。

©2013-2022 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。

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

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してください。

```
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。
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

©2013-2022 Tencent Cloud. All rights reserved.

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：

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：

©2013-2022 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	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。

音声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。

©2013-2022 Tencent Cloud. All rights reserved.

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

リモートの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。
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を取得します。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。

```
public abstract void sendRoomTextMsg(String message, 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
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

©2013-2022 Tencent Cloud. All rights reserved.

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.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	コマンドワードです。開発者がカスタマイズするものであり、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。

```
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：

©2013-2022 Tencent Cloud. All rights reserved.

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の注意喚起情報を表示します。

```
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

©2013-2022 Tencent Cloud. All rights reserved.

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
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	新しく入室したキャスターの ID。

©2013-2022 Tencent Cloud. All rights reserved.

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

視聴者入室通知を受信します。

```
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	TRTCLiveUserInfo	入室した視聴者の情報。

onAudienceExit

視聴者退室通知を受信します。

```
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

©2013-2022 Tencent Cloud. All rights reserved.

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();
```

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。

©2013-2022 Tencent Cloud. All rights reserved.

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を受信します。

```
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を受信します。

```
void onRecvRoomCustomMsg(String cmd, String message, TRTCLiveRoomDef.TRTCLiveUser
Info 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	コマンドワードです。開発者がカスタマイズするものであり、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

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。

```
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);
```

©2013-2022 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
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：

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

©2013-2022 Tencent Cloud. All rights reserved.

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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

TRTCLiveRoom API (Flutter)

最終更新日：：2022-07-19 16:21:32

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

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廃棄します。
registerListener	イベントコールバックを設定します。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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

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。

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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。

ルームイベントのコールバック

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	視聴者入室通知を受信します。
onAudienceExit	視聴者退室通知を受信します。

©2013-2022 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

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。
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を受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
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、使用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	プロフィール画像のアドレス。

ルーム関連インターフェース関数

©2013-2022 Tencent Cloud. All rights reserved.

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：

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：

©2013-2022 Tencent Cloud. All rights reserved.

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。その中には、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。
- 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、この関数は無視できます。

```
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<String>	ルームナンバーリスト。

©2013-2022 Tencent Cloud. All rights reserved.

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

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();
```

©2013-2022 Tencent Cloud. All rights reserved.

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);
```

パラ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を指定してください。

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：**

©2013-2022 Tencent Cloud. All rights reserved.

- 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してください。

```
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。

©2013-2022 Tencent Cloud. All rights reserved.

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してください。

```
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：

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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();
```

音声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);
```

©2013-2022 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	true：ミュート起動；false：ミュート停止。

muteRemoteAudio

リモートの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。
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を取得します。

```
getAudioEffectManager();
```

©2013-2022 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

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

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
cmd	String	コマンドワードです。開発者がカスタマイズするものであり、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。

> 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の注意喚起情報を表示します。

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	警告メッセージ。

onKickedOffline

©2013-2022 Tencent Cloud. All rights reserved.

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
result	int	result > 0 の場合は入室に費やした時間(ms)、result < 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

©2013-2022 Tencent Cloud. All rights reserved.

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
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

視聴者入室通知を受信します。

```
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

©2013-2022 Tencent Cloud. All rights reserved.

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。
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

©2013-2022 Tencent Cloud. All rights reserved.

てマイク接続から退出してください。

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してください。

パラ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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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
message	String	テキストメッセージ。

onRecvRoomCustomMsg

カスタムメッセージを受信します。

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
command	String	コマンドワードです。開発者がカスタマイズするものであり、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。

©2013-2022 Tencent Cloud. All rights reserved.

オンラインカラオケ
Quick Integration (TUIKaraoke)
TUIKaraoke (iOS)の統合

最終更新日：：2022-09-06 14:54:30

コンポーネントの説明

TUIKaraokeはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIKaraokeコンポーネントを統合することにより、数行のコードを書くだけで、アプリケーションにオンラインカラオケシーンを組み込むことができ、カラオケ、マイク管理、ギフトの送付と受領、テキストチャットなどのTRTCのKTVシーンでの関連機能を体験できるようになります。TUIKaraokeはAndroidプラットフォーム用のソースコードもサポートしています。基本機能は下図のとおりです：

説明：
TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

©2013-2022 Tencent Cloud. All rights reserved.



コンポーネントの統合

ステップ1：TUIKaraokeコンポーネントのダウンロードとインポート

クリックしてGithubに進み、コードのクローン/ダウンロードを選択した後、iOS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にコピーし、次のようにインポート動作を完了します。

- `Podfile` ファイルにインポートコマンドを追加します。次をご参照ください：

```
pod 'TUIKaraoke', :path => "./", :subspecs => ["TRTC"]
pod 'TXLiteAVSDK_TRTC'
pod 'TXAppBasic', :path => "TXAppBasic/"
```

- 端末を開き、 `Podfile` ファイルのあるディレクトリ下に進み、インストールコマンドを実行します。次をご参照ください：

```
pod install
```

©2013-2022 Tencent Cloud. All rights reserved.

ステップ2：権限の設定

プロジェクトのinfo.plistファイルの中でAppの権限を設定します。SDKには以下の権限が必要です（iOSシステムではマイクを動的に申請してください）：

```
<key>NSMicrophoneUsageDescription</key>
<string>Karaokeにはマイクへのアクセス権限が必要です</string>
```

ステップ3：初期化およびログイン

インターフェースに関する説明については、TUIKaraokeをご参照ください。

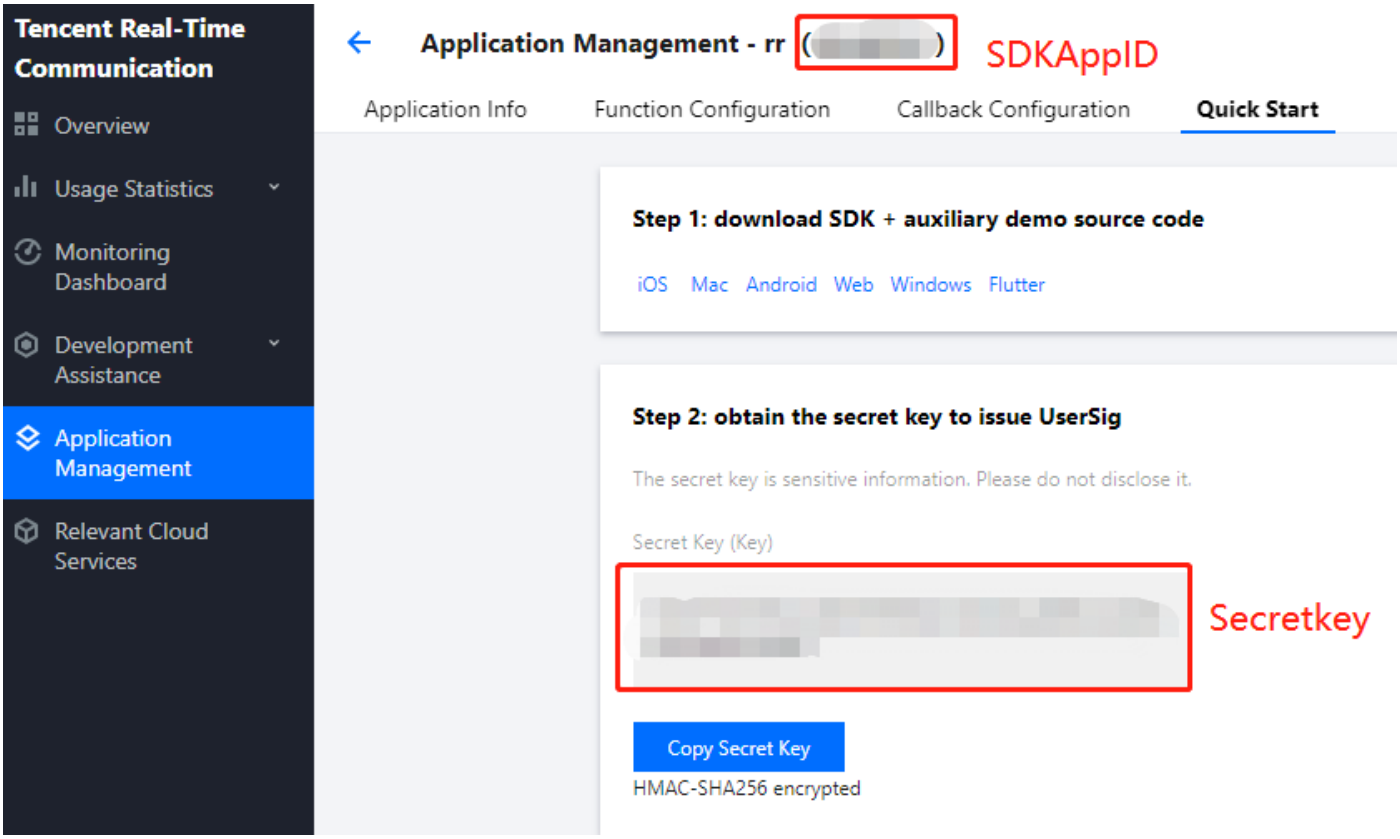
```
// 1.初期化
let karaokeRoom = TRTCKaraokeRoom.shared()
karaokeRoom.setDelegate(delegate: self)
// 2.ログイン
karaokeRoom.login(SDKAppID: Int32(SDKAppID), UserId: UserId, UserSig: ProfileMana
ger.shared.curUserSig()) { code, message in
if code == 0 {
//ログイン成功
}
}
```

パラメータの説明：

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション**

©2013-2022 Tencent Cloud. All rights reserved.

情報をクリックすると、SDKAppID情報が次の図のように表示されます：



- **Secretkey**：**TRTCアプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます：
- **userId**：現在のユーザーのIDです。文字列タイプであり、長さは32バイト以内とし、特殊文字の使用はサポートしていません。英語または数字の使用をお勧めします。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ができます。
- **userSig**：SDKAppId、userId、Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のUserSigがオンラインで直接生成されます。その他の情報については、UserSigの計算、使用方法をご参照ください。

ステップ4：オンラインKTVシーンの実装

1. **キャスターがルームを作成TUIKaraoke.createRoom**

```
int roomId = "ルームID";
let param = RoomParam.init()
param.roomName = "ルーム名";
param.needRequest = false; // 管理者によるマイク・オン確認の要否
param.seatCount = 8; //ルームの座席数。計8席あります
param.coverUrl = "ルームカバー図のURL";
karaokeRoom.createRoom(roomID: Int32(roomInfo.roomID), roomParam: param) { [wea
```

©2013-2022 Tencent Cloud. All rights reserved.

```
k self] (code, message) in
guard let `self` = self else { return }
if code == 0 {
//作成に成功
}
}
```

2. リスナーが入室TUIKaraoke.enterRoom

```
karaokeRoom.enterRoom(roomID: roomInfo.roomID) { [weak self] (code, message) in
guard let `self` = self else { return }
if code == 0 {
//入室に成功
}
}
```

3. リスナーが自主的にマイク・オンTUIKaraoke.enterSeat

```
// 1.リスナーが呼び出してマイク・オン
int seatIndex = 1;
karaokeRoom.enterSeat(seatIndex: seatIndex) { [weak self] (code, message) in
guard let `self` = self else { return }
if code == 0 {
//マイク・オン成功
}
}
// 2.onSeatListChangeコールバックを受信し、マイクリストを更新します
func onSeatListChange(seatInfoList: [SeatInfo]) {
}
```

説明：
マイク管理に関連するその他の操作については、TUIKaraokeインターフェースドキュメントをご参照の上、必要に応じて実装するか、またはTUIKaraokeデモプロジェクトをご参照ください。

4. 音楽再生とKTV体験シーンの実装

ご自身の業務に応じて音楽IDおよびURLリンクを取得し、楽曲を再生できます。詳細についてはTUIKaraoke音楽再生インターフェースをご参照ください。

```
//音楽の再生
karaokeRoom.startPlayMusic(musicID: musicID, originalUrl: muscicLocalPath, acco
```

©2013-2022 Tencent Cloud. All rights reserved.

```
mpanyUrl: accompanyLocalPath);
//音楽の停止
karaokeRoom.stopPlayMusic();
```

上記の手順が完了すると、KTVの基本機能を実装できます。業務上、チャット、ギフト送付などの機能も必要な場合、次の機能を統合することができます。

ステップ5：テキストチャット機能（オプション）

各キャスターまたはリスナー間のテキストチャット機能を実装したい場合は、次のメソッドによってチャットメッセージを送受信することができます。
インターフェースに関する説明については、TRTCKaraokeRoom.sendRoomTextMsgをご参照ください。

```
// 発信側：テキストメッセージの発信
karaokeRoom.sendRoomTextMsg(message: message) { [weak self] (code, message) in
if code == 0 {
//送信に成功
}
}
// 受信側：テキストメッセージのモニタリング
karaokeRoom.setDelegate(delegate: self)
func onRecvRoomTextMsg(message: String, userInfo: UserInfo) {
debugPrint("" + userInfo.userName + "から受信したメッセージ：" + message)
}
```

ステップ6：ギフト送付機能（オプション）

ギフト送付および受領機能を実装したい場合は、次のメソッドによってギフトを送付または受領し、表示することができます。

```
// 送信側：カスタマイズした「IMCMD_GIFT」によってギフトメッセージを区別
karaokeRoom.sendRoomCustomMsg(cmd: kSendGiftCmd, message: message) { code, msg in
if (code == 0) {
//送信に成功
}
}
// 受信側：ギフトメッセージのモニタリング
karaokeRoom.setDelegate(delegate: self)
func onRecvRoomCustomMsg(cmd: String, message: String, userInfo: UserInfo) {
if cmd == kSendGiftCmd {
debugPrint("" + userInfo.userName + "から受領したギフト：" + message)
}
}
```

©2013-2022 Tencent Cloud. All rights reserved.

よくあるご質問

TUIKaraokeコンポーネントはボイスチェンジ、キー調整、リバーブなどのオーディオエフェクト機能をサポートしていますか？

サポートしています。具体的にはTUIKaraokeデモプロジェクトをご参照ください。

? 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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIKaraoke (Android)の統合

最終更新日：：2022-09-06 15:08:33

コンポーネントの説明

TUIKaraokeはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIKaraokeコンポーネントを統合することにより、数行のコードを書くだけで、アプリケーションにオンラインカラオケシーンを組み込むことができ、カラオケ、マイク管理、ギフトの送付と受領、テキストチャットなどのTRTCのKTVシーンでの関連機能を体験できるようになります。TUIKaraokeはiOSプラットフォーム用のソースコードもサポートしています。基本機能は下図のとおりです。



コンポーネントの統合

ステップ1：TUIKaraokeコンポーネントのダウンロードとインポート

クリックしてGithubに進み、コードのクローン/ダウンロードを選択した後、Androidディレクトリ下のSource、Debugディレクトリをプロジェクトにコピーし、次のようにインポート動作を完了します。

©2013-2022 Tencent Cloud. All rights reserved.

- `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'
include ':Debug'
```

- appのbuild.gradleファイルにTUIKaraokeに対する依存関係を追加します。

```
api project(':Source')
```

- ルートディレクトリの `build.gradle` ファイルに `TRTC SDK` および `IM SDK` の依存関係を追加します。

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

ステップ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システムではマイク、ストレージ読み取りの権限などを動的に申請する必要があります）。

```
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" /> // ユースケース：フローティングウィンドウ機能にはこの権限が必要です。
<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.RECORD_AUDIO" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
<uses-permission android:name="android.permission.BLUETOOTH" /> // ユースケース：Bluetoothイヤホンを使用する場合はこの権限が必要です。
```

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.** { *; }
```

ステップ3：初期化およびログイン

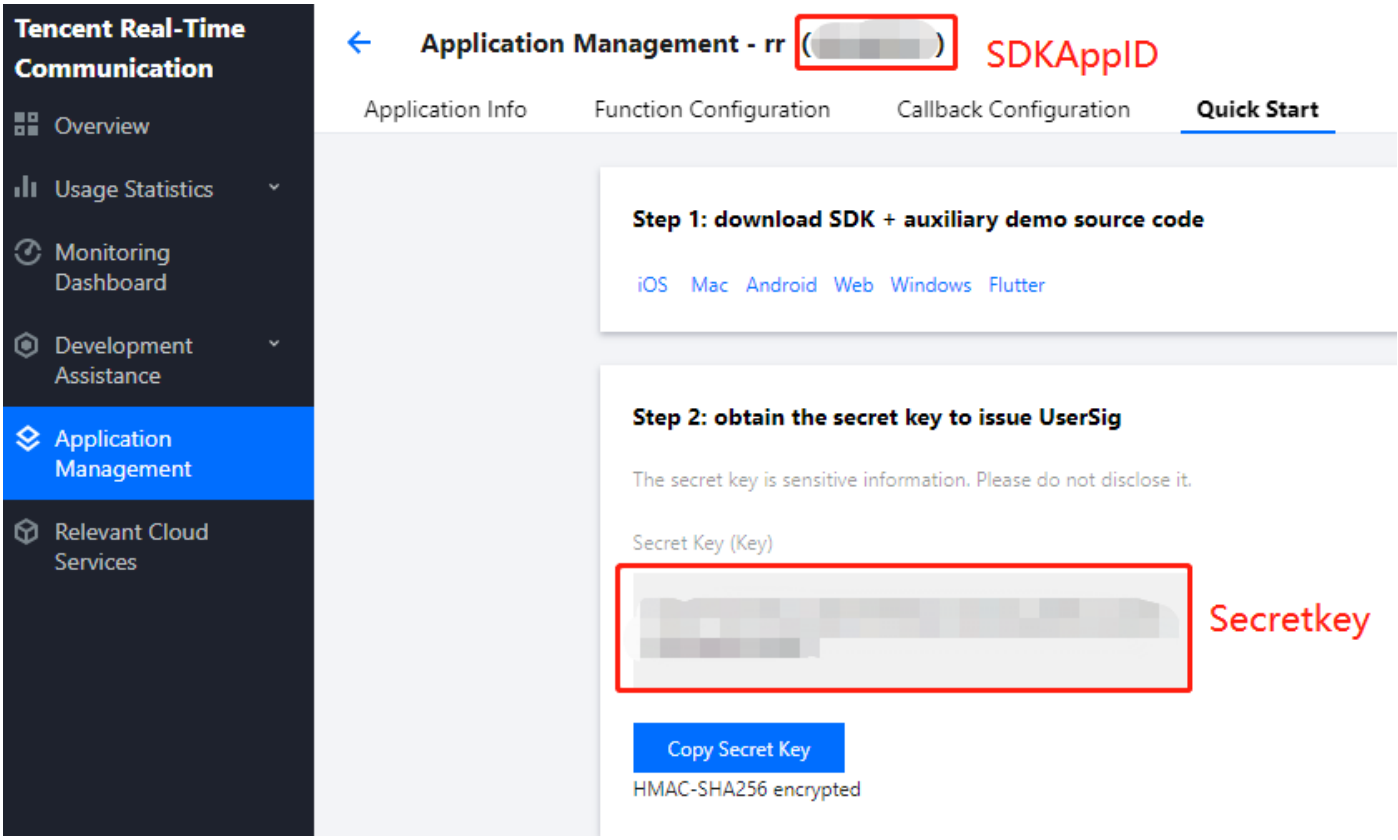
インターフェースに関する詳細については、TUIKaraokeをご参照ください。

©2013-2022 Tencent Cloud. All rights reserved.

```
// 1.初期化
TRTCKaraokeRoom mTRTCKaraokeRoom = TRTCKaraokeRoom.sharedInstance(this);
mTRTCKaraokeRoom.setDelegate(this);
// 2.ログイン
mTRTCKaraokeRoom.login(SDKAppID, UserID, UserSig, new TRTCKaraokeRoomCallback.Act
ionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//ログイン成功
}
}
});
```

パラメータの説明：

- **SDKAppID**：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション情報**をクリックすると、SDKAppID情報が次の図のように表示されます。



- **Secretkey**：**TRTCアプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます。

©2013-2022 Tencent Cloud. All rights reserved.

- **userId**：現在のユーザーのIDです。文字列形式で、長さは32バイト以内とし、特殊文字の使用はサポートしていません。英語または数字の使用をお勧めします。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ができます。
- **userSig**：SDKAppId、userId、Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のUserSigがオンラインで直接生成されます。UserSigの計算、使用方法をご参照ください。

ステップ4：オンラインKTVシーンの実装

1. キャスターがルームを作成TUIKaraoke.createRoom

```
int roomId = "ルームID";
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
roomParam.seatCount = 8; //ルームの座席数。計8席あります
roomParam.coverUrl = "ルームカバー図のURL";
mTRTCKaraokeRoom.createRoom(roomId, roomParam, new TRTCKaraokeRoomCallback.Acti
onCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//作成に成功
}
}
});
```

2. リスナーが入室TUIKaraoke.enterRoom

```
mTRTCKaraokeRoom.enterRoom(roomId, new TRTCKaraokeRoomCallback.ActionCallback()
{
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//入室に成功
}
}
});
```

3. リスナーが自主的にマイク・オン TUIKaraoke.enterSeat

©2013-2022 Tencent Cloud. All rights reserved.

```
// 1.リスナーが呼び出してマイク・オン
int seatIndex = 1;
mTRTCKaraokeRoom.enterSeat(seatIndex, new TRTCKaraokeRoomCallback.ActionCallbac
k() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//マイク・オン成功
}
}
});
// 2.onSeatListChangeコールバックを受信し、マイクリストを更新します
@Override
public void onSeatListChange(final List<TRTCKaraokeRoomDef.SeatInfo> seatInfoLi
st) {
}
```

説明：
マイク管理に関連するその他の操作については、TUIKaraokeインターフェースドキュメントをご参照の上、必要に応じて実装するか、または当社のTUIKaraokeデモプロジェクトをご参照ください。

4. **音楽再生とKTV体験シーンの実装**

ご自身の業務に応じて音楽IDおよびURLリンクを取得し、楽曲を再生できます。インターフェースの詳細についてはTUIKaraoke音楽再生インターフェースをご参照ください。

```
//音楽の再生
mTRTCKaraokeRoom.startPlayMusic(musicID,url);
//音楽の停止
mTRTCKaraokeRoom.stopPlayMusic();
```

上記の手順が完了すると、KTVの基本機能を実装できます。業務上、テキストチャット、ギフト送付などの機能も必要な場合、次の機能を統合することができます。

ステップ5：テキストチャット機能（オプション）

各キャスターまたはリスナー間のテキストチャット機能を実装したい場合は、次のメソッドによってチャットメッセージを送受信することができます。
インターフェースに関する詳細については、TRTCKaraokeRoom.sendRoomTextMsgをご参照ください。

©2013-2022 Tencent Cloud. All rights reserved.

```
// 発信側：テキストメッセージの発信
mTRTCKaraokeRoom.sendRoomTextMsg("Hello Word!", new TRTCKaraokeRoomCallback.Actio
nCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//送信に成功
}
}
});
// 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 userInf
o) {
Log.d(TAG,"が" + userInfo.userName + "から受信したメッセージ:" + message);
}
});
```

ステップ6：ギフト送付機能（オプション）

ギフト送付および受領機能を実装したい場合は、次のメソッドによってギフトを送付または受領し、表示することができます。

```
// 送信側：カスタマイズした「CMD_GIFT」によってギフトメッセージを区別
mTRTCKaraokeRoom.sendRoomCustomMsg("CMD_GIFT",date, new TRTCKaraokeRoomCallback.A
ctionCallback() {
@Override
public void onCallback(int code, String msg) {
if (code == 0) {
//送信に成功
}
}
});
// 受信側：ギフト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.Us
erInfo userInfo) {
if ("CMD_GIFT".equals(cmd)) {
// ギフトメッセージの受信
Log.d(TAG, "" + userInfo.userName + "からのギフトを受信:" + message);
}
}
});
```

©2013-2022 Tencent Cloud. All rights reserved.

よくあるご質問

TUIKaraokeコンポーネントはボイスチェンジ、キー調整、リバーブなどのオーディオエフェクト機能をサポートしていますか。

サポートしています。

説明：
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。

©2013-2022 Tencent Cloud. All rights reserved.

TUIKaraoke APIのクエリー
TRTCKaraoke(iOS)

最終更新日：：2022-11-09 11:23:18

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブにします。

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。

- 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

©2013-2022 Tencent Cloud. All rights reserved.

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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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ります（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ります（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。
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

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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ります/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を受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

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を受信。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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

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。

©2013-2022 Tencent Cloud. All rights reserved.

```
/**
* 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());
```

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です。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 に送信します。

```
/**
* 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キューです。TRTCKaraokeRoomの各種ステータス通知コールバックは、指定したqueueに送信されます。
```

©2013-2022 Tencent Cloud. All rights reserved.

```
*/
- (void)setDelegateQueue:(dispatch_queue_t)queue NS_SWIFT_NAME(setDelegateQueue(q
ueue:));
```

パラ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送信されます。

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:u
serSig: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。文字列タイプであり、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、使用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：

©2013-2022 Tencent Cloud. All rights reserved.

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	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です。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	ルーム情報です。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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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最新のKaraokeルームリストを取得します。その中には、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。

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、この関数は無視できます。

```
- (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

©2013-2022 Tencent Cloud. All rights reserved.

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。

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
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。
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:(NSString *)backingUrl NS_SWIFT_NAME(startPlayMusic(musicID:originalUrl:accompanyUrl:));
```

パラメータは下表に示すとおりです：

©2013-2022 Tencent Cloud. All rights reserved.

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：
> 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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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。

```
- (void)enterSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callbac
k 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ります（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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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：

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：

©2013-2022 Tencent Cloud. All rights reserved.

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）。

説明：
任意のマイクをミュート/ミュート解除します。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:(ActionCallbac
k _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：該当するマイクをミュートします；false：該当するマイクをミュート解除します。
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/ミュート解除にします。

©2013-2022 Tencent Cloud. All rights reserved.

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：該当するマイクをクローズします； false：該当するマイクをクローズ解除します。
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;
```

©2013-2022 Tencent Cloud. All rights reserved.

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：

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：

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
- (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：ミュートをオンにします；false：ミュートをオフにします。

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：ミュートをオンにします；false：ミュートをオフにします。

©2013-2022 Tencent Cloud. All rights reserved.

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

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。

```
- (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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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	コマンドワードです。開発者がカスタマイズするものであり、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。
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。

©2013-2022 Tencent Cloud. All rights reserved.

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:));
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

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:));
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

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
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

©2013-2022 Tencent Cloud. All rights reserved.

ユーザーのマイクがミュートになっているかどうかのコールバックです。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すると、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	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です。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

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
seatInfoList	List<SeatInfo>	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。

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：

©2013-2022 Tencent Cloud. All rights reserved.

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。
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:));
```

©2013-2022 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	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を受信します。

```
- (void)onRecvRoomTextMsg:(NSString *)message
userInfo:(KaraokeUserInfo *)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	UserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージを受信します。

```
- (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	コマンドワードです。開発者がカスタマイズするもので、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。

©2013-2022 Tencent Cloud. All rights reserved.

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
id	String	招待ID。
inviter	String	招待者のユーザーID。
cmd	String	開発者がカスタマイズする業務指定のコマンドワードです。
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
- (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。

```
- (void)onInvitationCancelled:(NSString *)identifier
invitee:(NSString *)invitee NS_SWIFT_NAME(onInvitationCancelled(identifier:invite
e:));
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

TRTCKaraoke(Android)

最終更新日：：2022-07-19 16:21:32

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。

> 説明：
> TUIKitシリーズコンポーネントは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サービスの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。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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。
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

©2013-2022 Tencent Cloud. All rights reserved.

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）。
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。
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

©2013-2022 Tencent Cloud. All rights reserved.

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
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。

©2013-2022 Tencent Cloud. All rights reserved.

API	説明
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

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ります/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を受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

シグナリングイベントのコールバック

API	説明

©2013-2022 Tencent Cloud. All rights reserved.

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

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してください。

©2013-2022 Tencent Cloud. All rights reserved.

```
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。

```
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】>アプリケーション情報の中でSDKAppIDを確認できます。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
userId	String	現在のユーザーID。文字列タイプであり、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、使用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(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

©2013-2022 Tencent Cloud. All rights reserved.

ルームの作成（管理者が呼び出し）：

```
public abstract void createRoom(int roomId, TRTCKaraokeRoomDef.RoomParam roomPara
m, 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
roomI	int	ルームIDは、ご自身でアサインし、一元管理してください。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	ルーム情報です。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してください。
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上で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。

©2013-2022 Tencent Cloud. All rights reserved.

enterRoom

入室（リスナーが呼び出し）。

```
public abstract void enterRoom(int roomId, 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
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

©2013-2022 Tencent Cloud. All rights reserved.

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、この関数は無視できます。

```
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<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。

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<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。
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：

©2013-2022 Tencent Cloud. All rights reserved.

- 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);
```

パラ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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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：
`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。

©2013-2022 Tencent Cloud. All rights reserved.

leaveSeat

ユーザーが視聴者になります（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(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

©2013-2022 Tencent Cloud. All rights reserved.

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, 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：
> 任意のマイクをミュート/ミュート解除します。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。

©2013-2022 Tencent Cloud. All rights reserved.

パラメータ	タイプ	意味
isMute	boolean	true：該当するマイクをミュートします；false：該当するマイクをミュート解除します。
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/ミュート解除にします。

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。

```
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：該当するマイクをクローズします； false：該当するマイクをクローズ解除します。
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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

```
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：

©2013-2022 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。

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
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

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。

```
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.Acti
onCallback 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。

©2013-2022 Tencent Cloud. All rights reserved.

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
cmd	String	コマンドワードです。開発者がカスタマイズするものであり、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。

©2013-2022 Tencent Cloud. All rights reserved.

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：

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。

©2013-2022 Tencent Cloud. All rights reserved.

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, 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。

©2013-2022 Tencent Cloud. All rights reserved.

```
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);
```

©2013-2022 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。
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すると、各メンバーの音量を通知します。

```
void onUserVolumeUpdate(List<TRTCCloudDef.TRTCVolumeInfo> userVolumes, int totalV
olume);
```

パラ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です。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<SeatInfo>	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：

©2013-2022 Tencent Cloud. All rights reserved.

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, 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。

©2013-2022 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(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を受信します。

```
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

©2013-2022 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。
userInfo	UserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージを受信します。

```
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：

パラメータ	タイプ	意味
command	String	コマンドワードです。開発者がカスタマイズするもので、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	開発者がカスタマイズする業務指定のコマンドワードです。
content	String	業務指定のコンテンツ。

©2013-2022 Tencent Cloud. All rights reserved.

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

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

©2013-2022 Tencent Cloud. All rights reserved.

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

```
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。

©2013-2022 Tencent Cloud. All rights reserved.