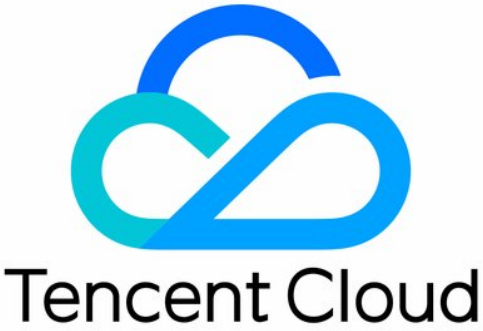
Tencent Real-Time Communication

Live Streaming (Including UI)

製品ドキュメント



©2013-2022 Tencent Cloud. All rights reserved.

Copyright Notice

©2013-2024 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：

Live Streaming (Including UI)
- TUILiveRoom (Android)の統合
- TUILiveRoom (iOS)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)

©2013-2022 Tencent Cloud. All rights reserved.

Live Streaming (Including UI) 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.

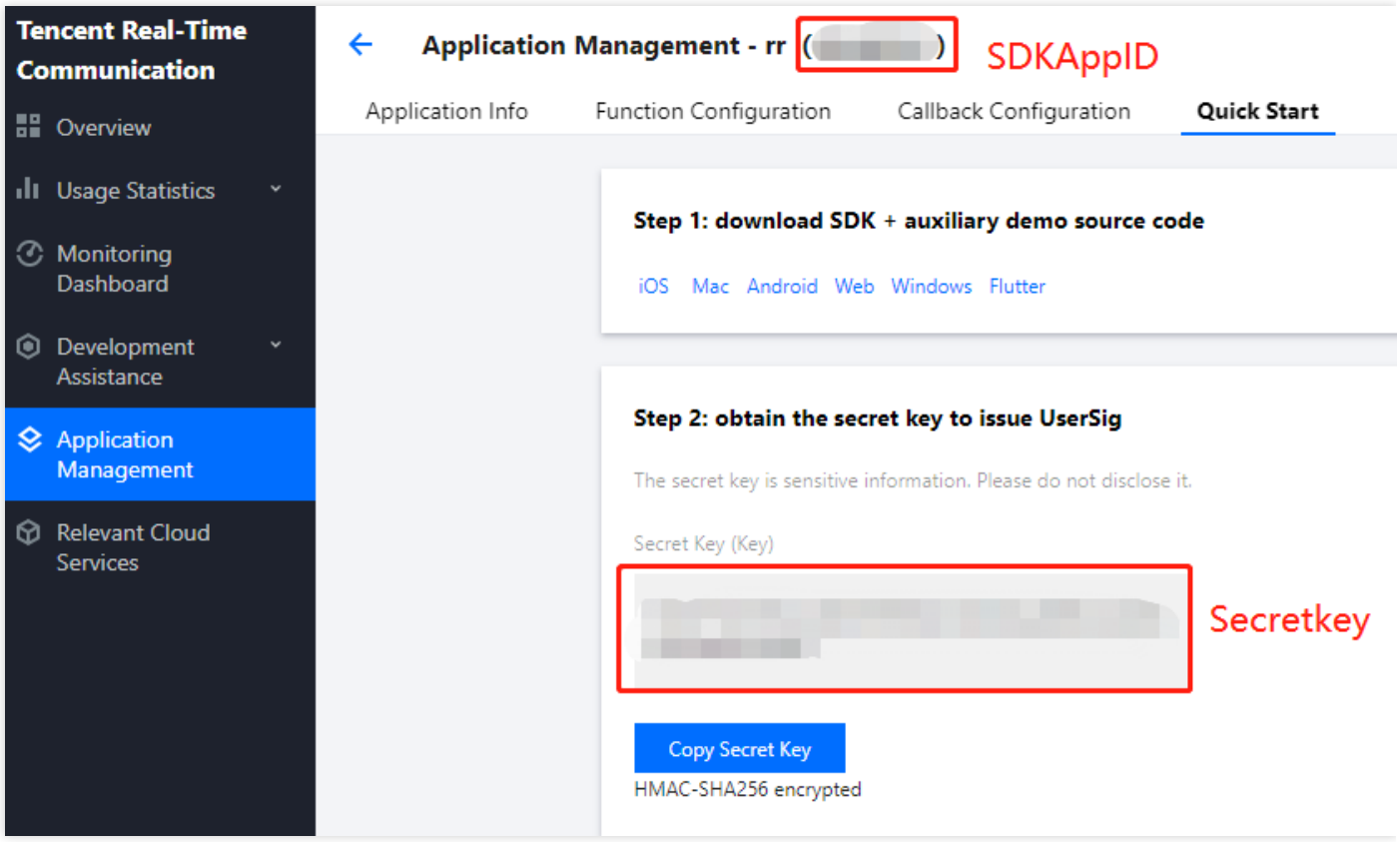
```
@Override
public void onConnectSuccess() { // 接続成功通知
super.onConnectSuccess();
}
@Override
public void onConnectFailed(int errorCode, String errorMsg) { // 接続失敗通知
super.onConnectFailed(errorCode, errorMsg);
}
@Override
public void onKickedOffline() { // ログインがキックアウトされたオフライン通知（例：アカウントが他のデバイスでログインしている）
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：

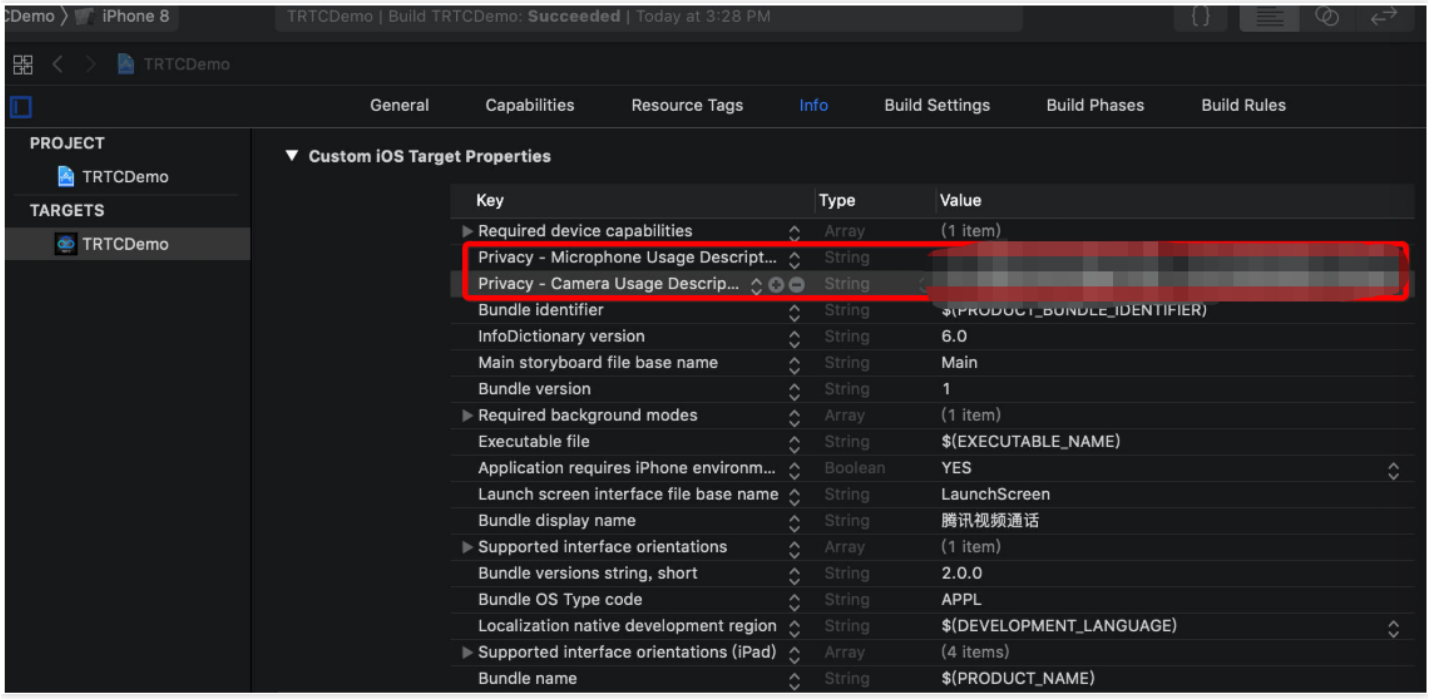
- `Source` 、 `Resources` フォルダと `TUILiveRoom.podspec` ファイルは同一のディレクトリ下にある必要があります。
- TXAppBasic.podspecはTXAppBasicフォルダ下にあります。

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>
```



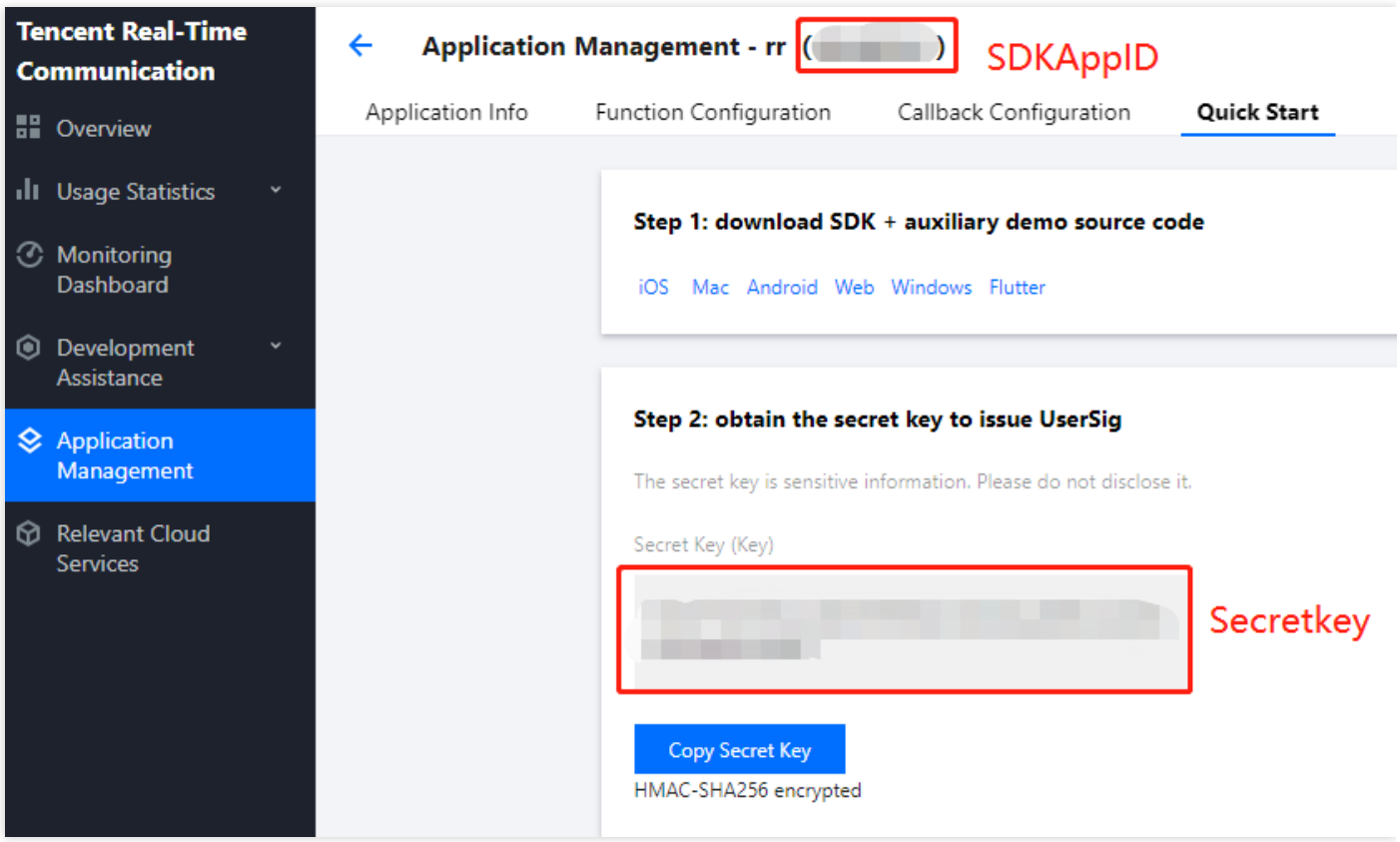
ステップ3：コンポーネントの初期化およびログイン

```
@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:playV
iew callback:nil];
[[TRTCLiveRoom shareInstance] startPublishWithStreamID:[NSString stringWithFor
mat:@"%@_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:(TRTCLiv
eUserInfo *)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 *)us
erID {
// 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.

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

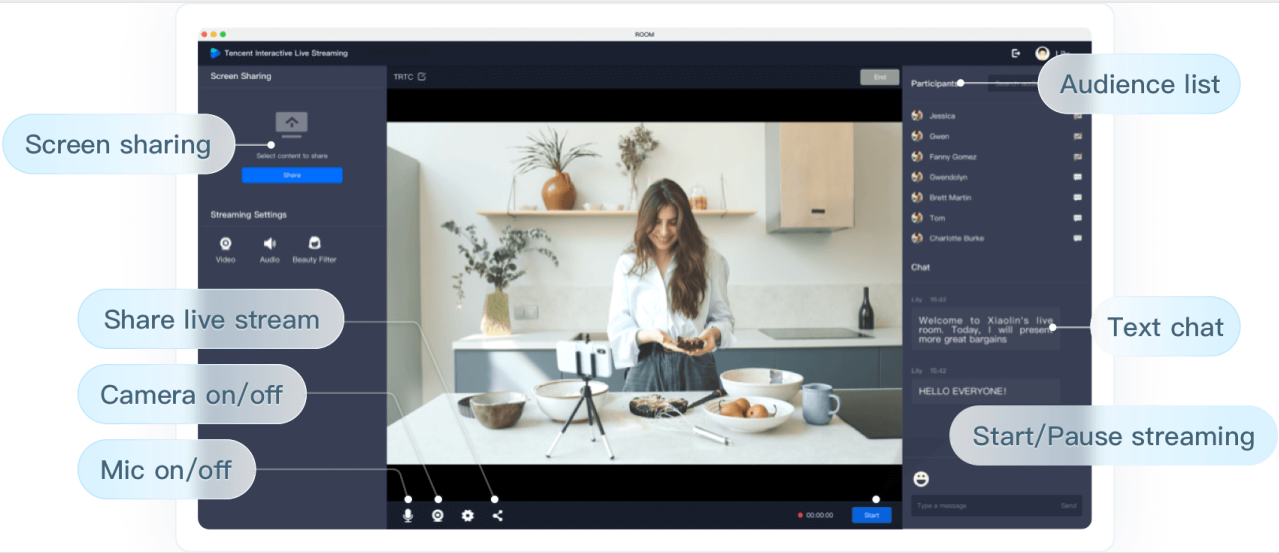
TUIPusher & TUIPlayerは、Web端末オープンソースの、UIを備えたライブストリーミングインタラクションコンポーネントです。TUIPusher & TUIPlayerはTRTC、 IMなどの基本SDKを統合し、企業によるライブストリーミング、eコマースマーケティング、業界研修、リモート教育などの様々なライブストリーミングシーン向けに、すぐにリリースできるWeb端末ライブストリーミングプッシュプルストリーミングツールによるソリューションをご提供します。

> 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。

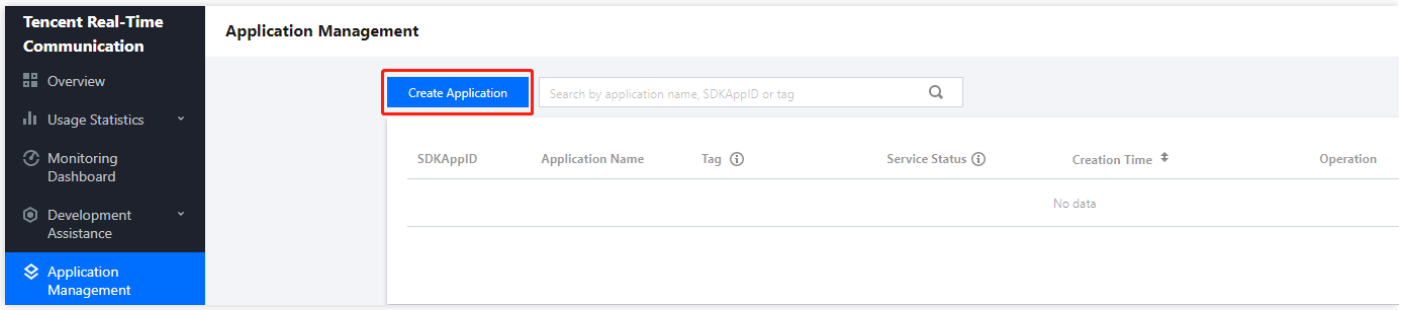
- 方法1：TRTCの場合
- 方法2：インスタントメッセージIMの場合

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。

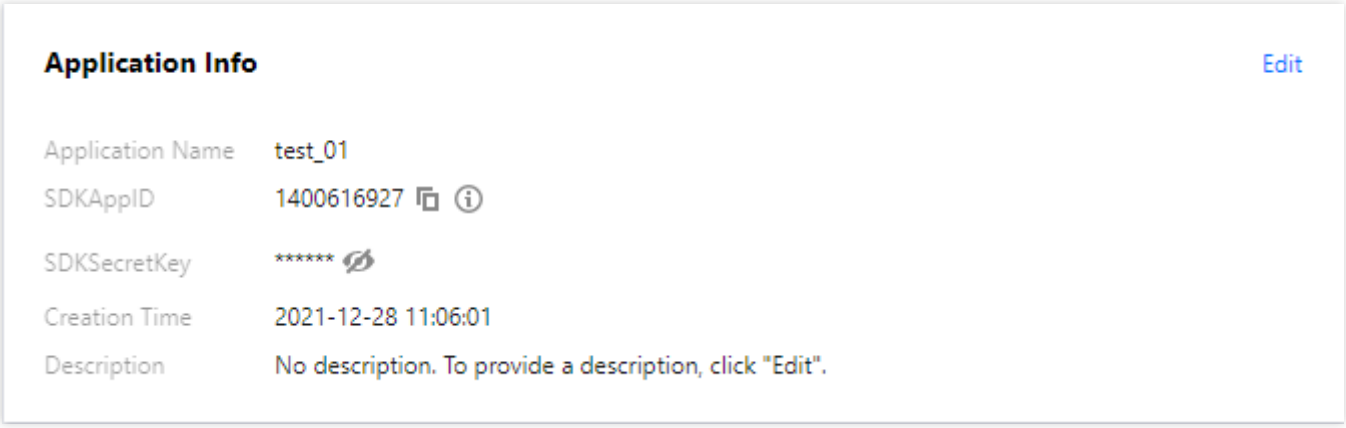
©2013-2022 Tencent Cloud. All rights reserved.

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。

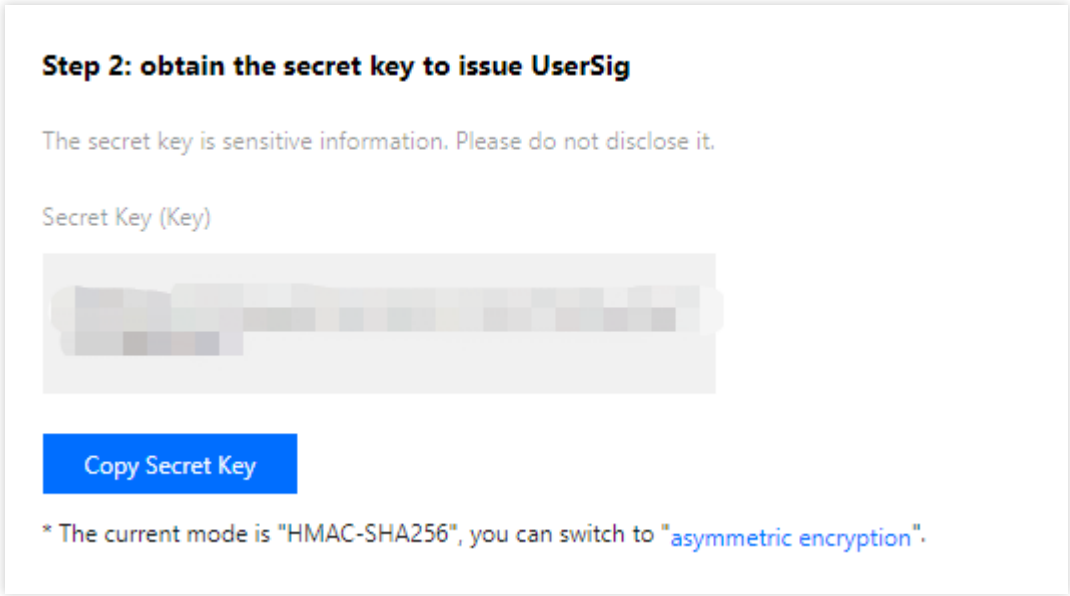


ステップ2： TRTCキー情報の取得

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：

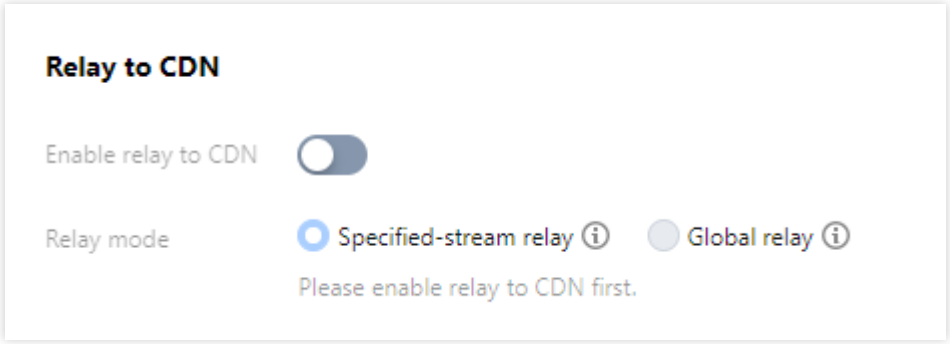
©2013-2022 Tencent Cloud. All rights reserved.

- 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://www.tencentcloud.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.

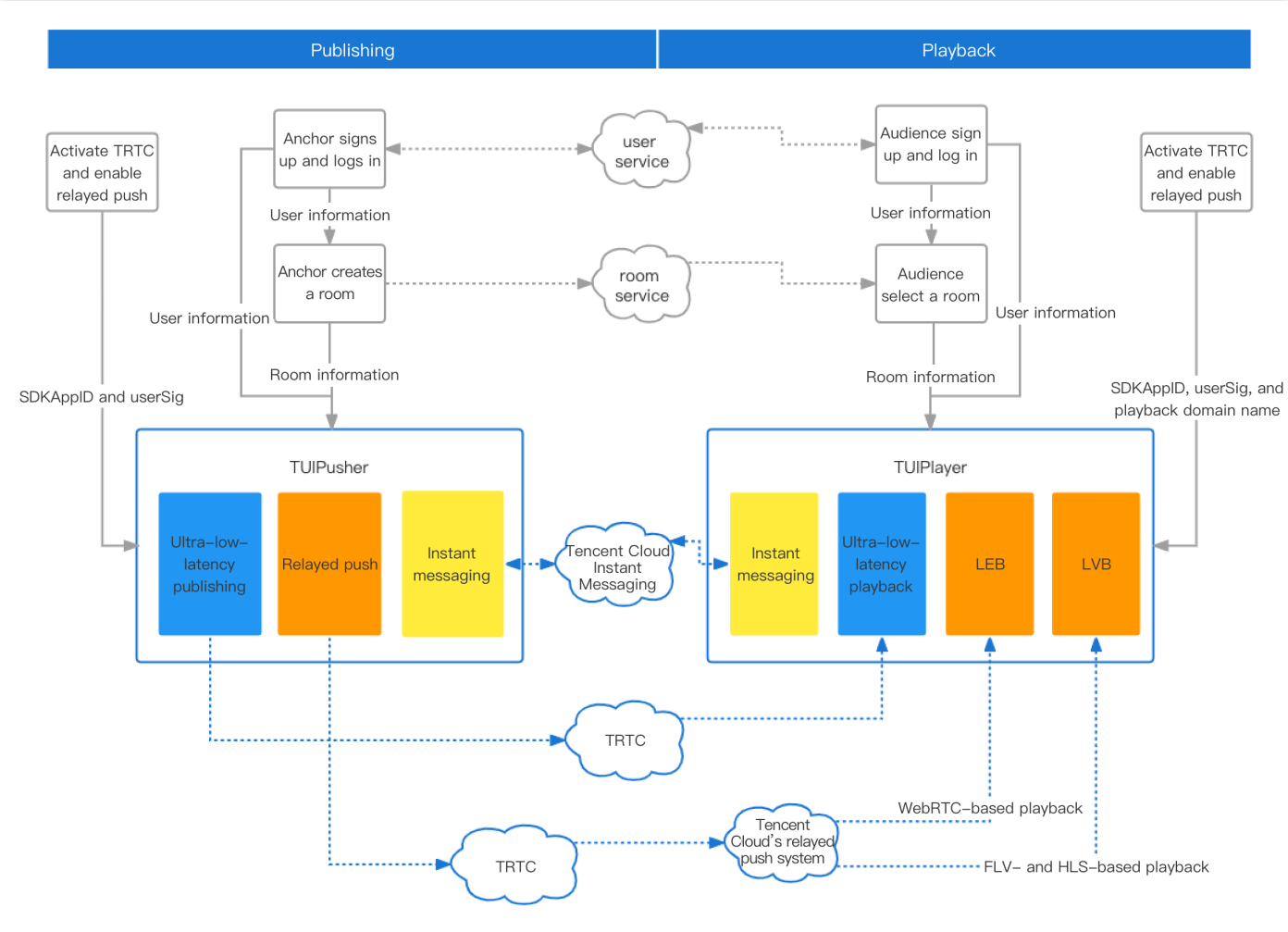
- ユーザーID、ユーザー名、ユーザープロフィール画像などを含む製品ユーザー情報を管理するためのユーザー管理システムを作成します。
- ライブストリーミングルームID、ライブストリーミングルーム名、ライブストリーミングルームのホスト情報などを含む製品のライブストリーミングルーム情報を管理するためのルーム管理システムを作成します。
- サーバーがUserSigを発行します。

> 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.tencentcloud.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://www.tencentcloud.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。

explain

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。
setDelegateHandler	イベントのコールバックが配置されているスレッドを設定します。
login	ログイン。
logout	ログアウト。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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。

©2013-2022 Tencent Cloud. All rights reserved.

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。

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

©2013-2022 Tencent Cloud. All rights reserved.

	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.

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

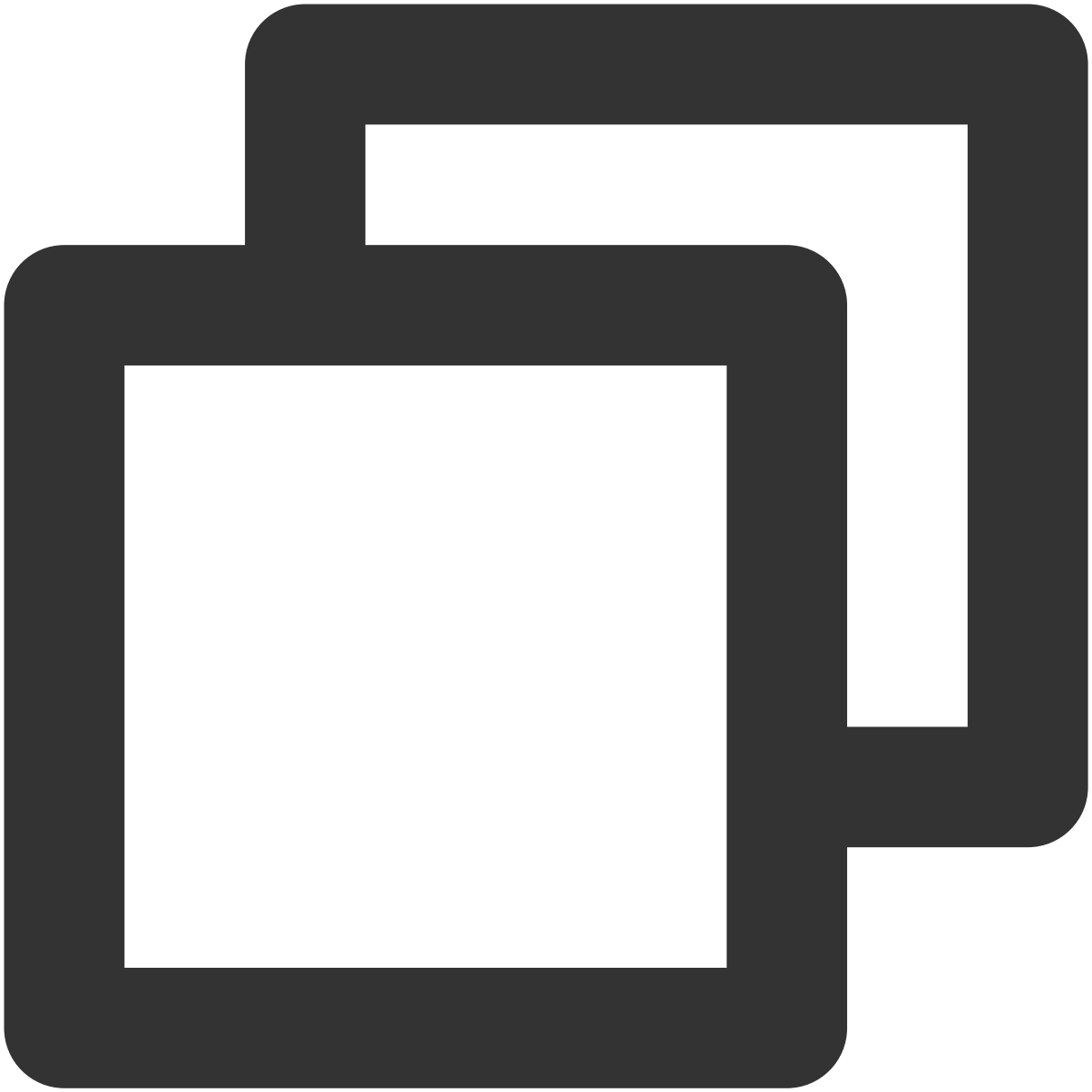
API	説明
onRecvRoomTextMsg	テキストメッセージの受信。
onRecvRoomCustomMsg	カスタムメッセージの受信。

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。

©2013-2022 Tencent Cloud. All rights reserved.



```
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

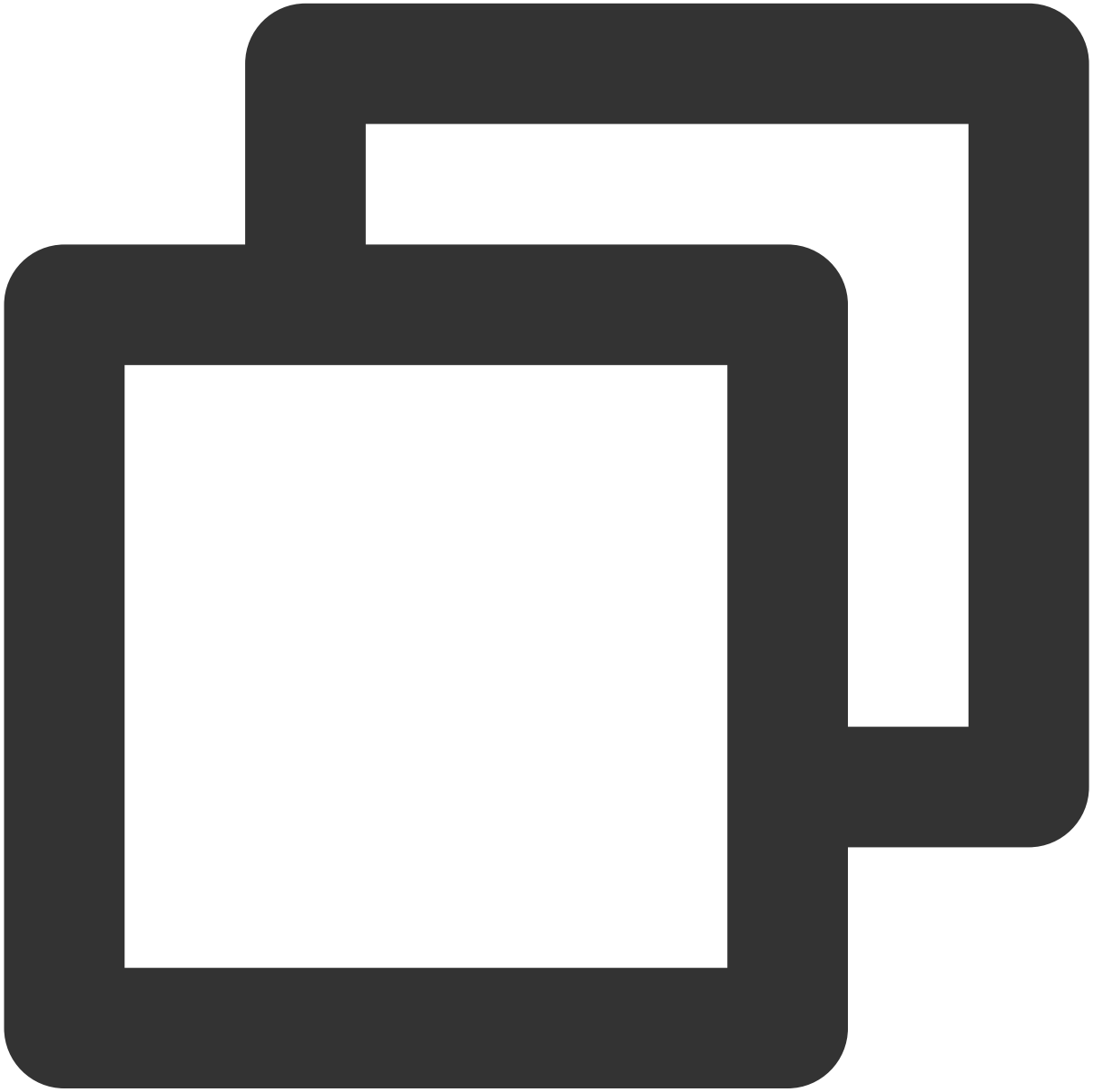
©2013-2022 Tencent Cloud. All rights reserved.

destroySharedInstance

TRTCLiveRoom シングルトンオブジェクトを破棄します。

explain

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。

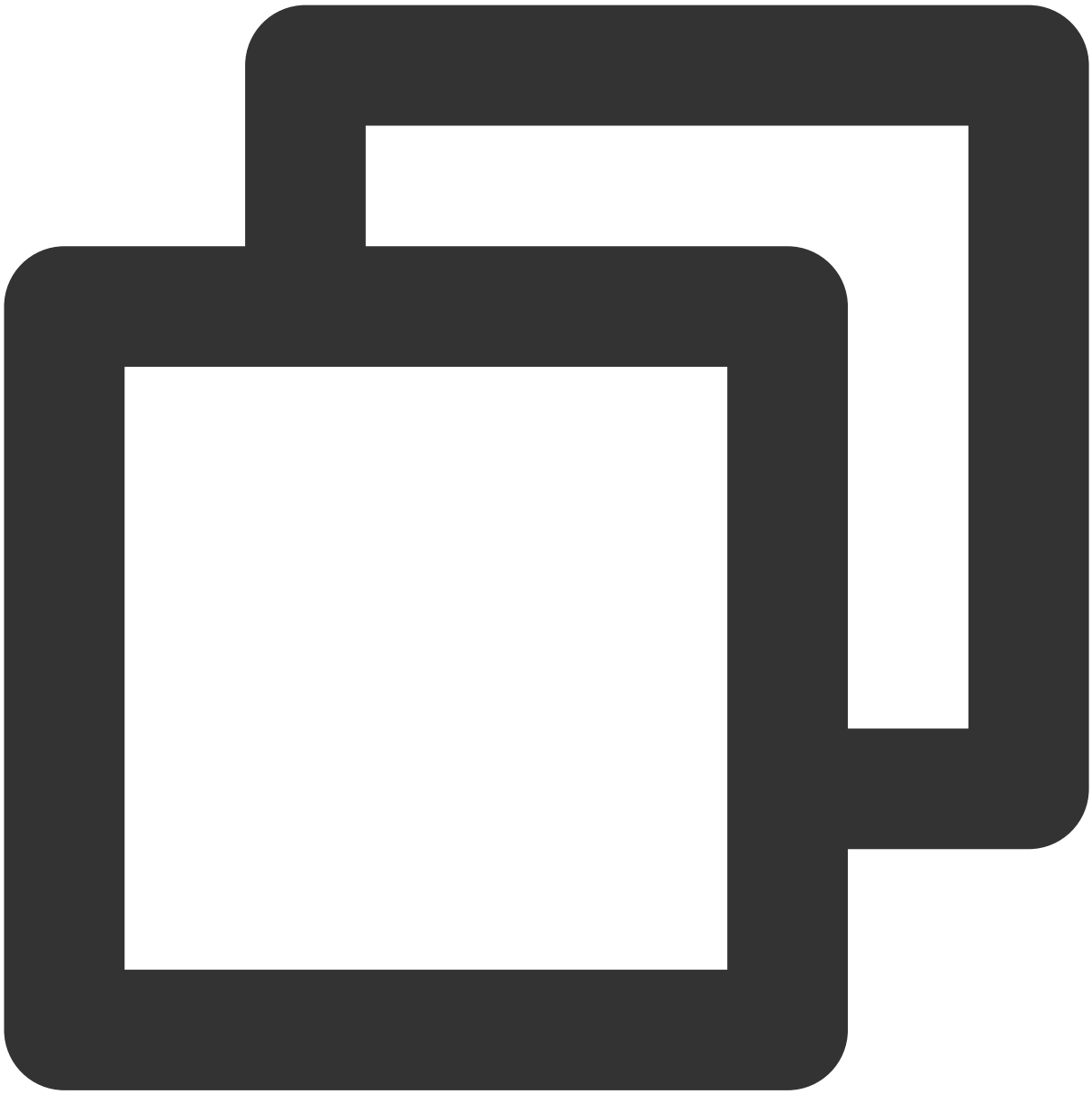


```
public static void destroySharedInstance();
```

setDelegate

©2013-2022 Tencent Cloud. All rights reserved.

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);
```

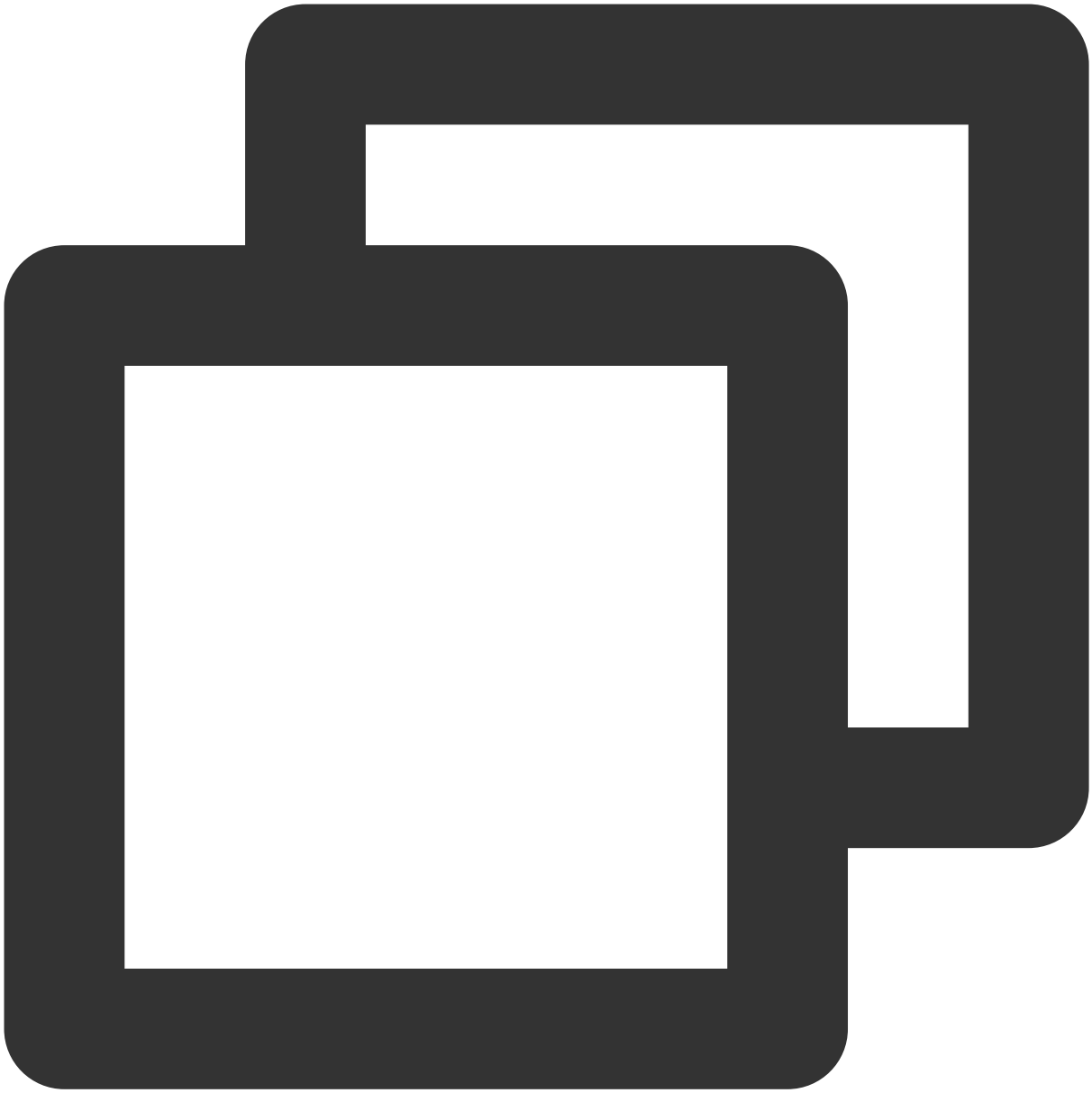
explain

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。

©2013-2022 Tencent Cloud. All rights reserved.



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

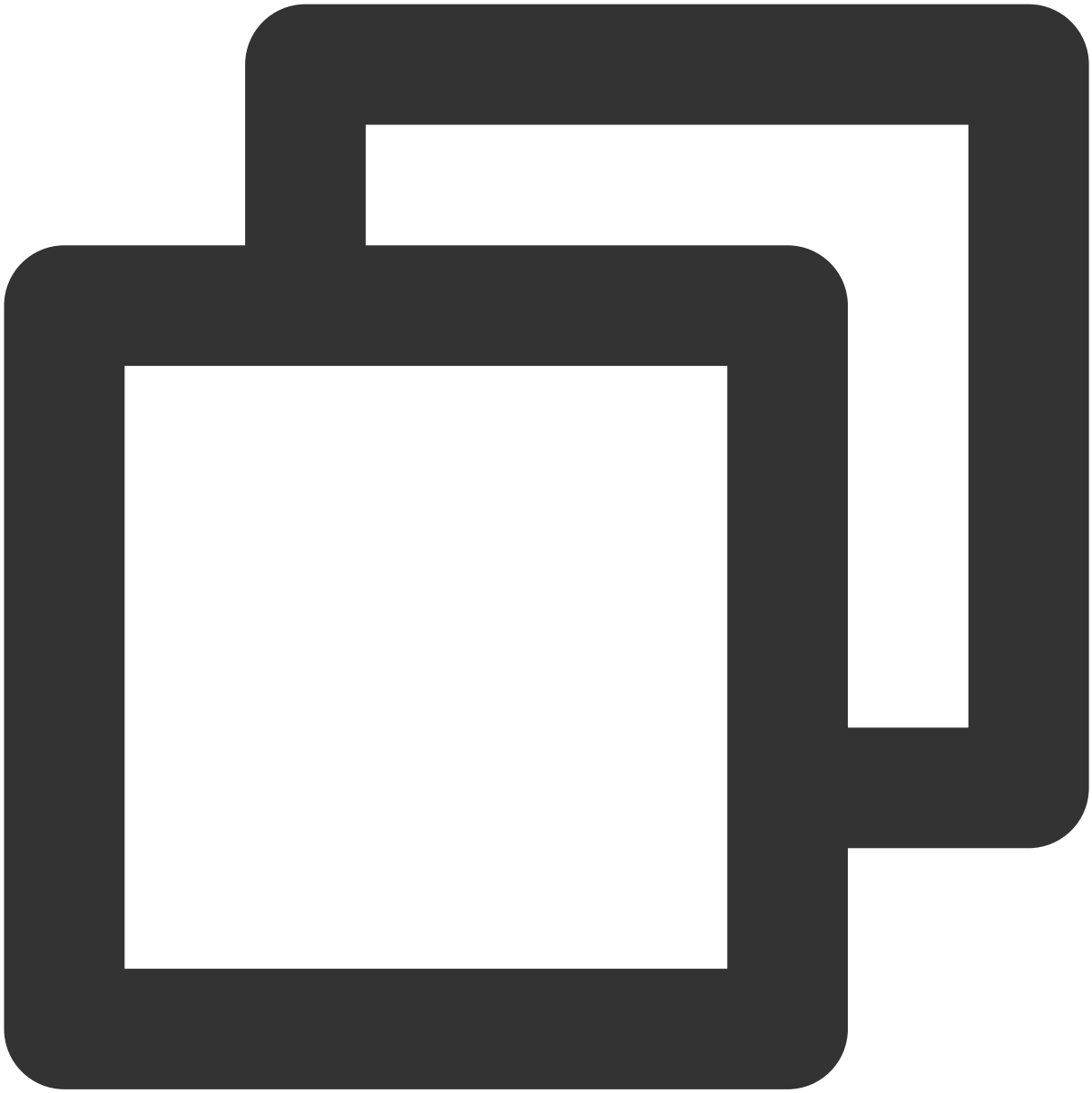
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
handler	Handler	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。

©2013-2022 Tencent Cloud. All rights reserved.

login

ログイン。



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

パラメータは下表に示すとおりです：

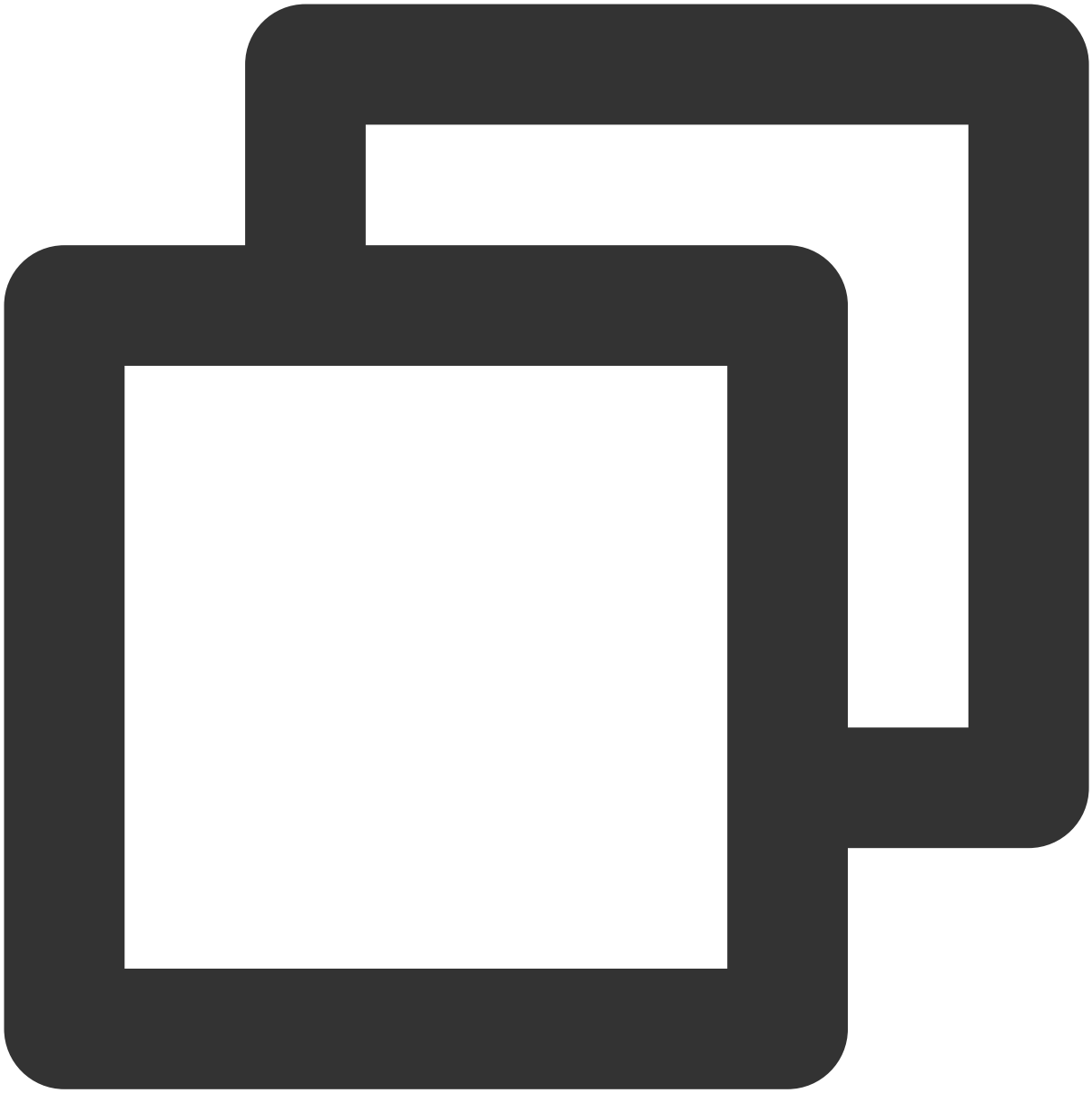
©2013-2022 Tencent Cloud. All rights reserved.

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。

©2013-2022 Tencent Cloud. All rights reserved.



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

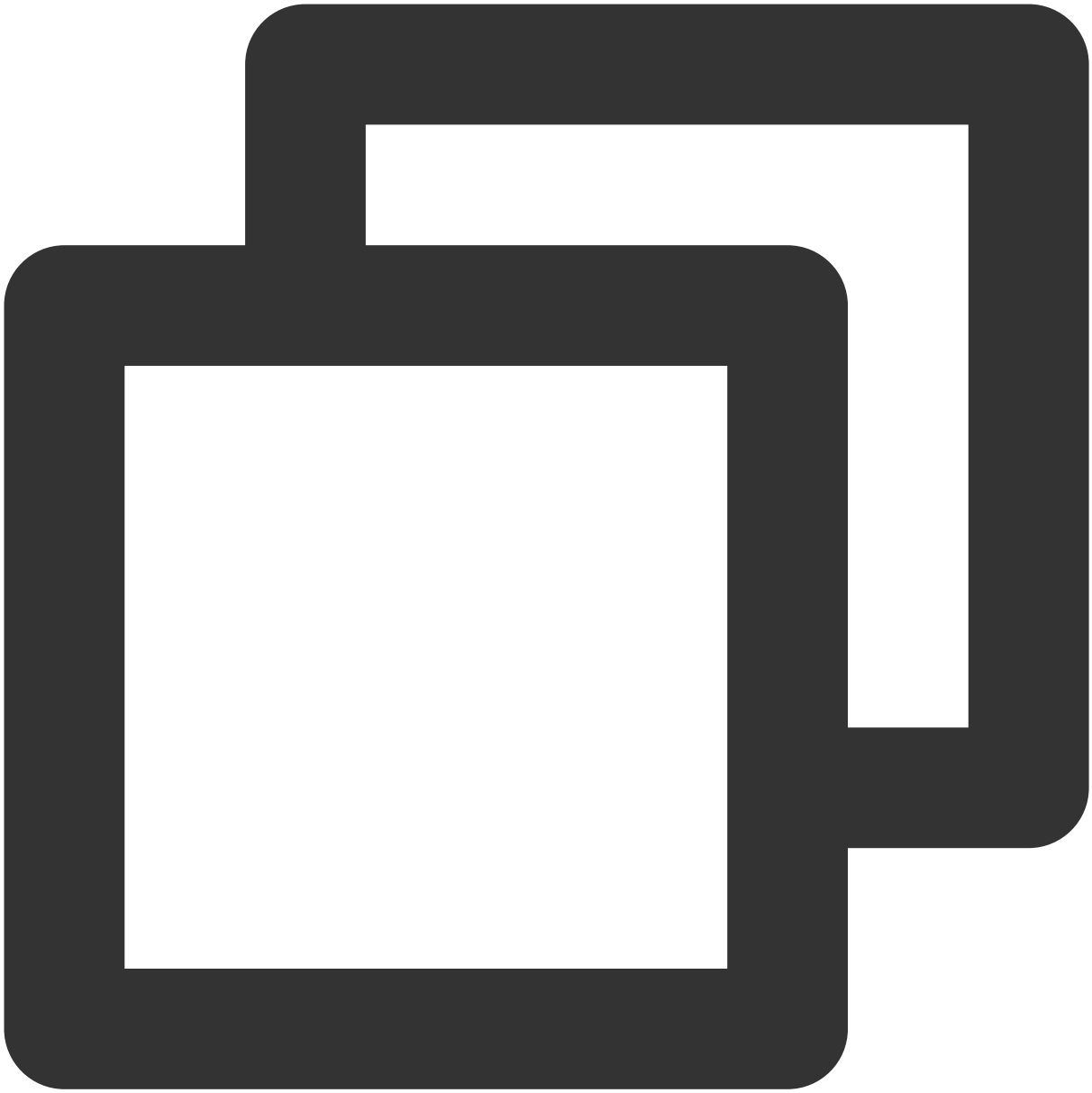
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
callback	ActionCallback	ログアウトのコールバック。成功時にcodeは0になります。

setSelfProfile

個人情報の修正。

©2013-2022 Tencent Cloud. All rights reserved.



```
public abstract void setSelfProfile(String userName, String avatarURL, TRTCLiveRoom
```

パラメータは下表に示すとおりです：

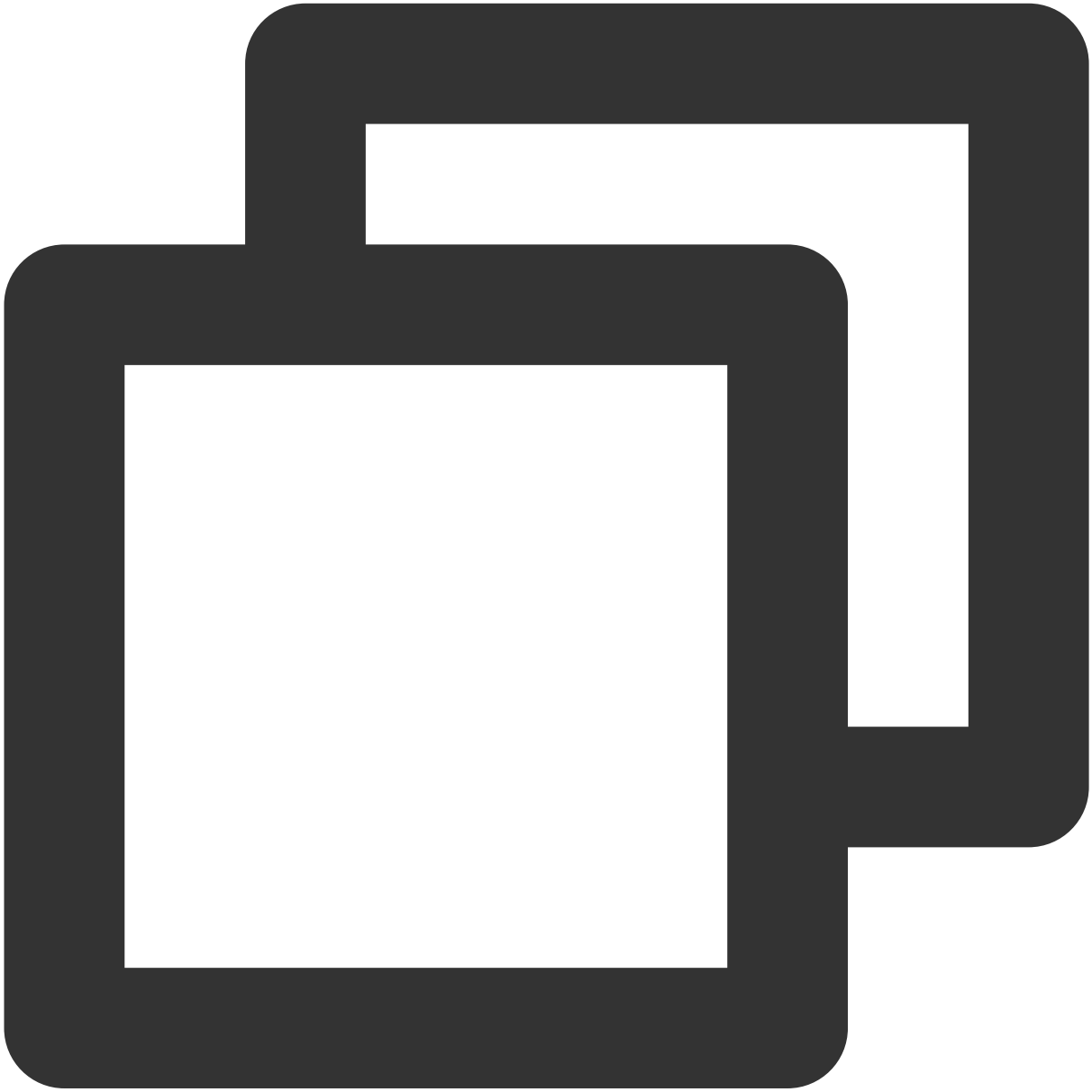
パラメータ	タイプ	意味
userName	String	ニックネーム。
avatarURL	String	プロフィール画像のアドレス。
callback	ActionCallback	個人情報設定のコールバック。成功時にcodeは0になります。

©2013-2022 Tencent Cloud. All rights reserved.

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 roo
```

パラ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.

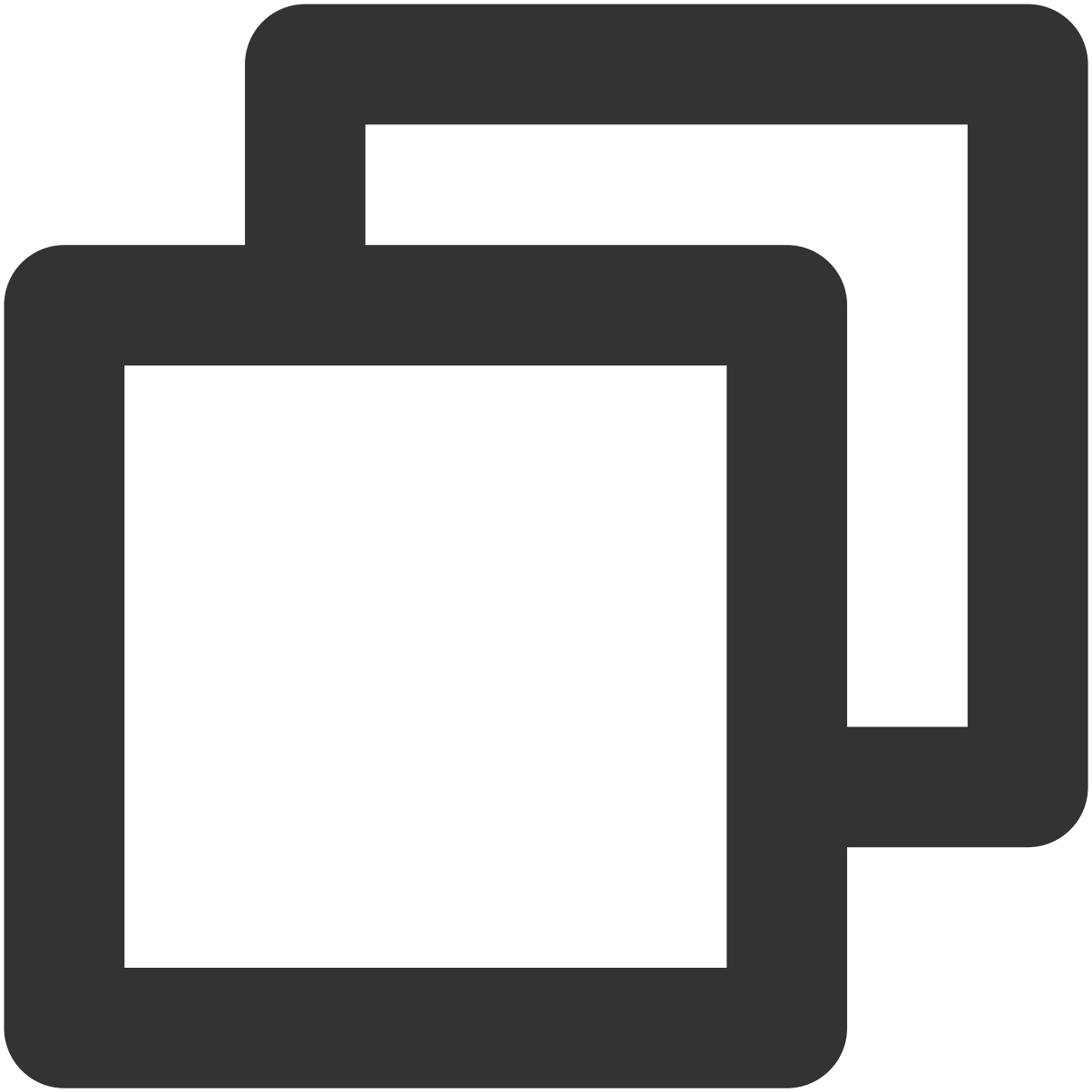
roomId	int	ルームIDは、ご自身でアサインし、一元管理してください。複数の roomID を、1つのライブストリーミングルームリストにまとめることができます。Tencent Cloudでは現在、ライブストリーミングルームリストの管理サービスを行っていませんので、ご自身でライブストリーミングルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報は、ルーム名、マイク情報、カバー情報などのルーム説明の情報に使用します。ルームリストおよびルーム情報をご自身のサーバーで管理する場合は、このパラメータは無視できます。
callback	ActionCallback	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。

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。

©2013-2022 Tencent Cloud. All rights reserved.



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

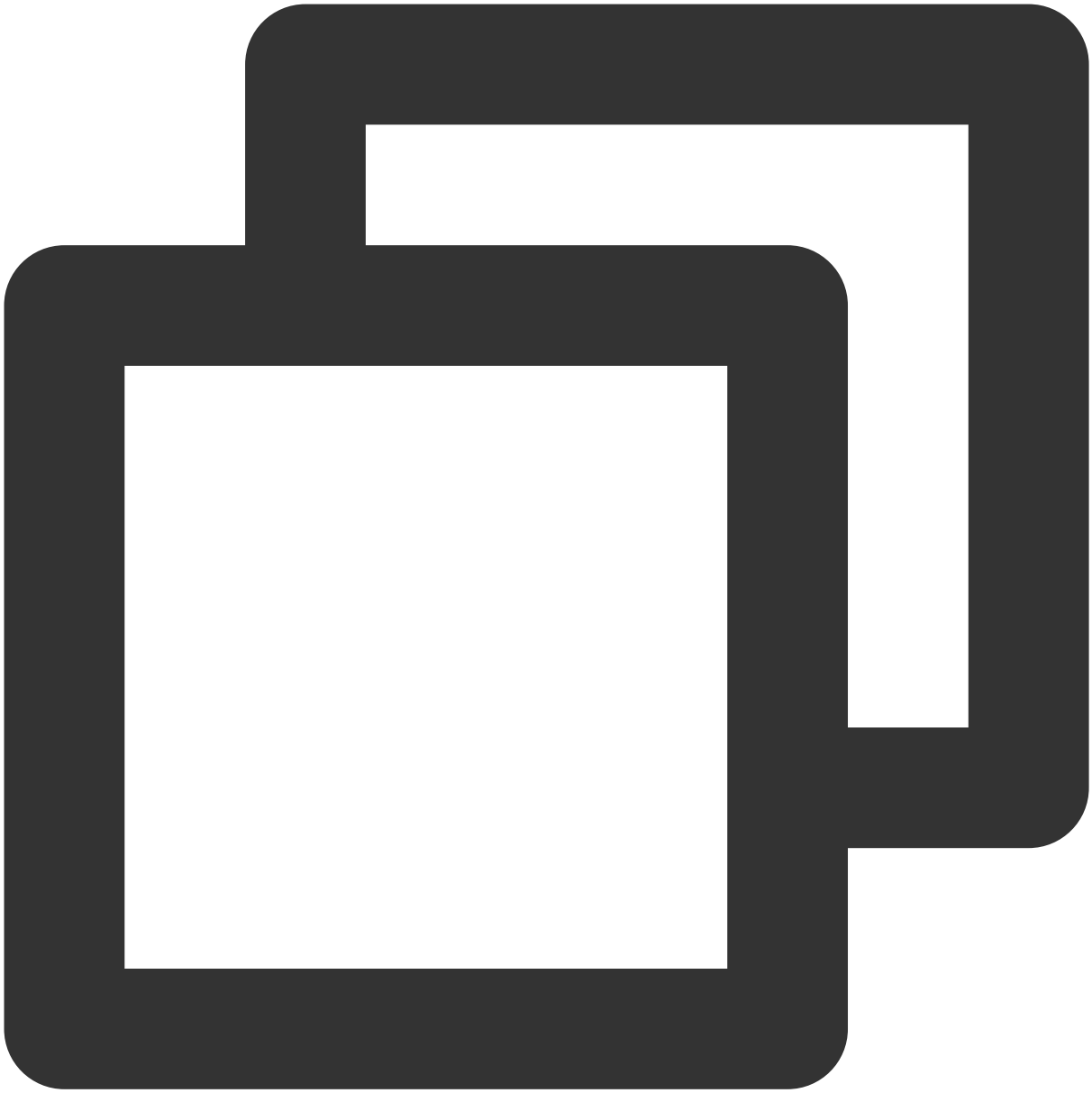
パラメータは下表に示すとおりです：

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）

©2013-2022 Tencent Cloud. All rights reserved.



```
public abstract void enterRoom(int roomId, TRTCLiveRoomCallback.ActionCallback call
```

パラ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。

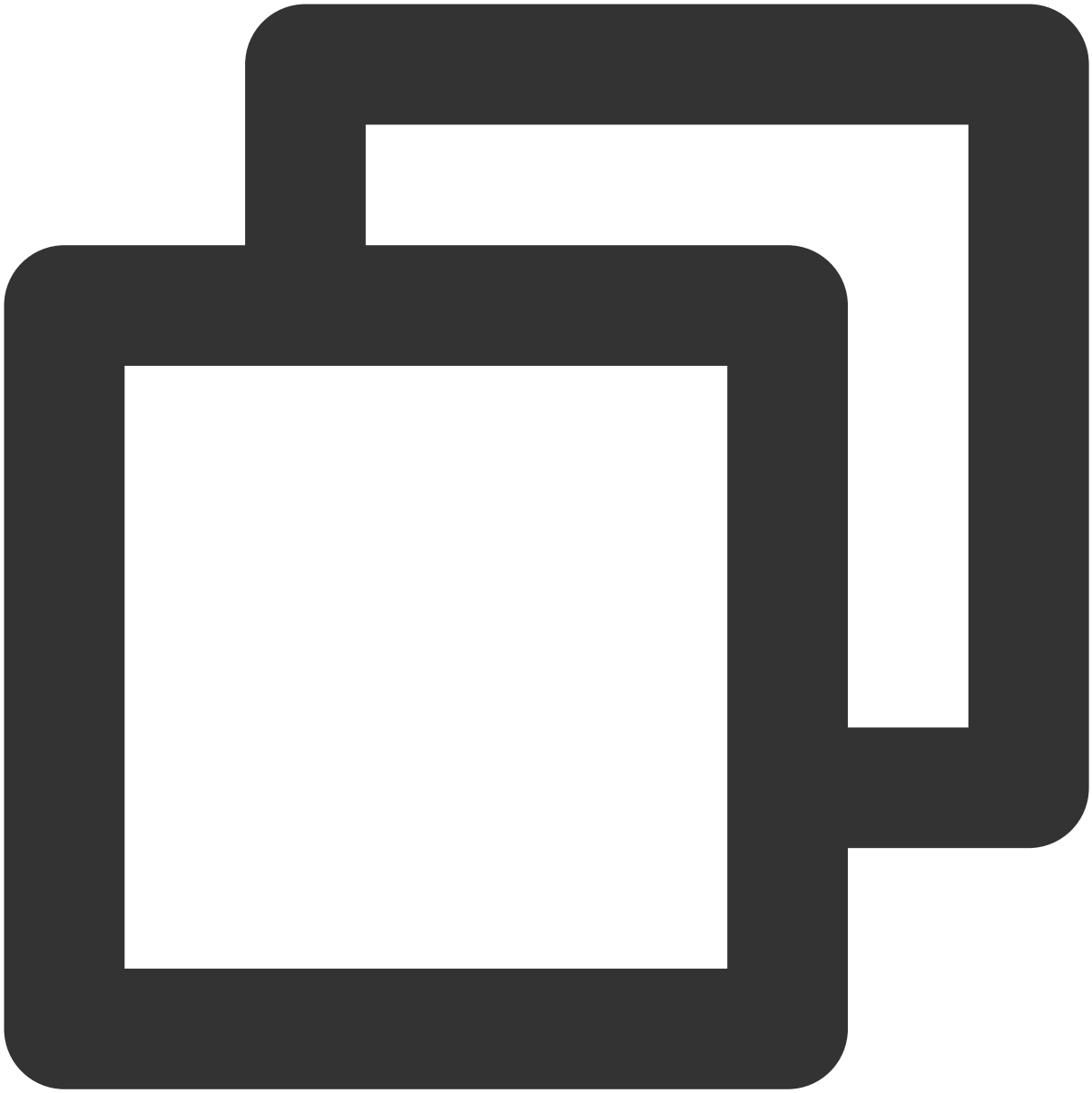
©2013-2022 Tencent Cloud. All rights reserved.

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`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。

©2013-2022 Tencent Cloud. All rights reserved.



```
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。

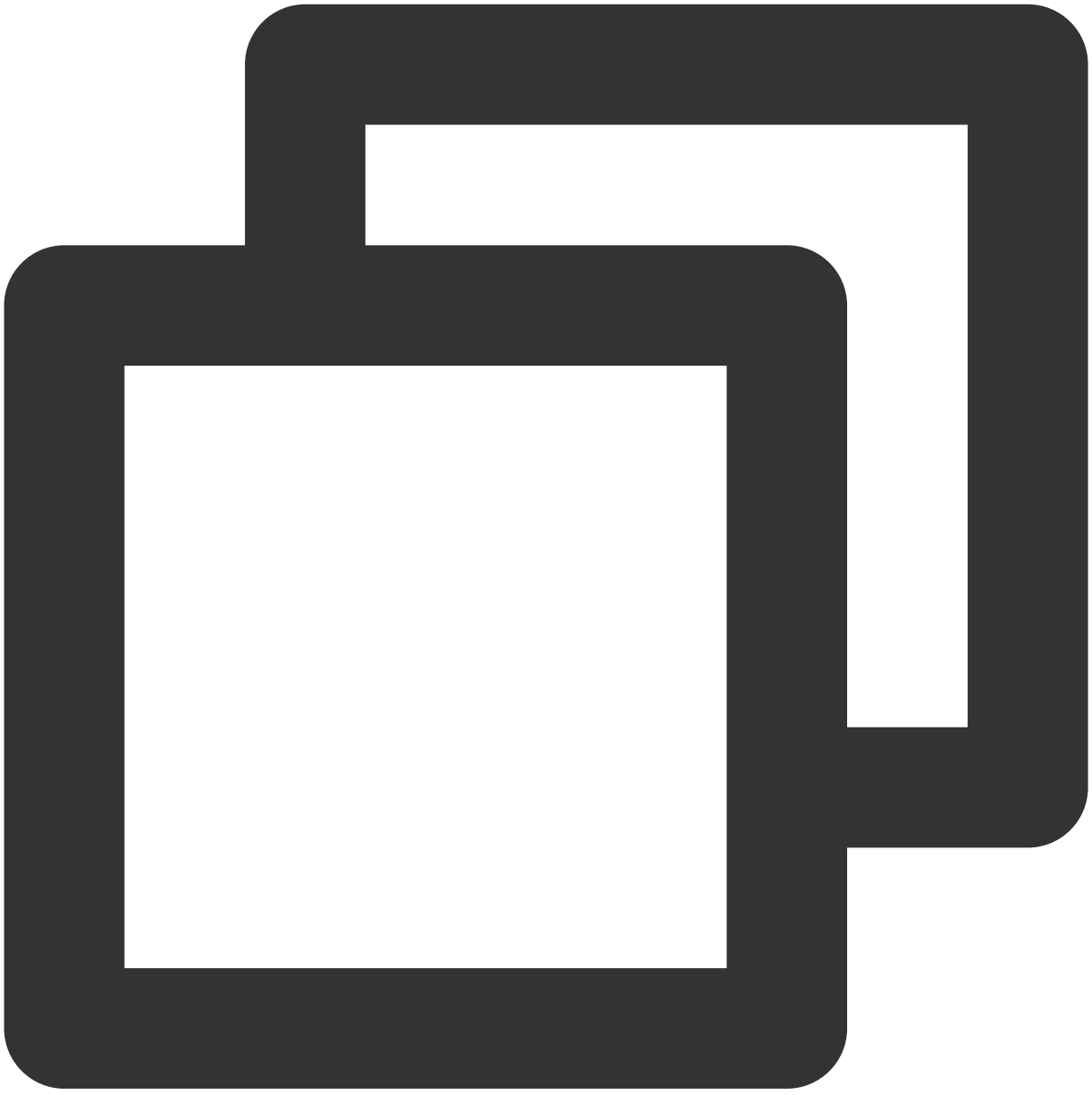
getRoomInfos

©2013-2022 Tencent Cloud. All rights reserved.

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。

explain

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.Ro
```

パラメータは下表に示すとおりです：

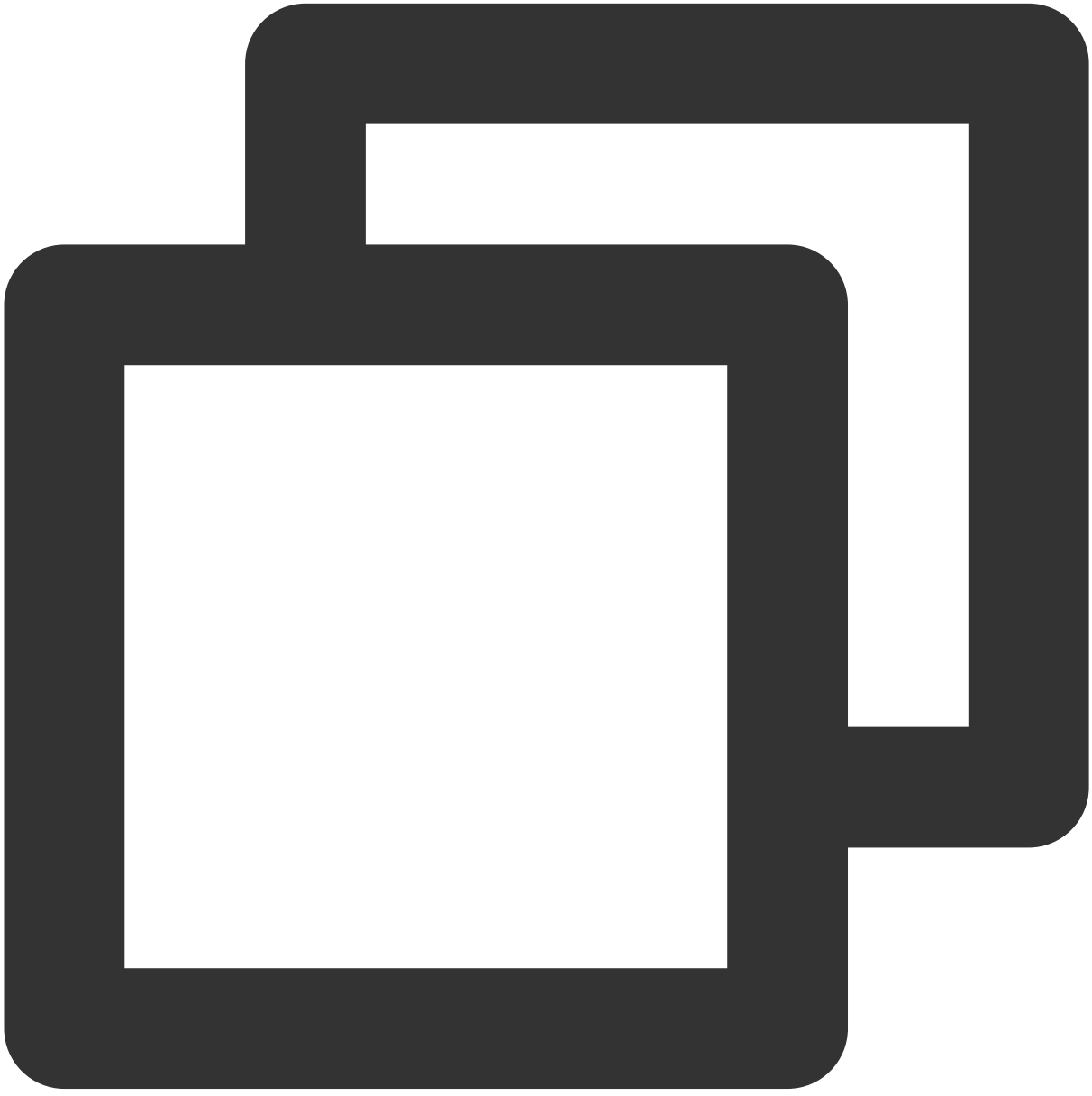
パラメータ	タイプ	意味

©2013-2022 Tencent Cloud. All rights reserved.

roomIdList	List<Integer>	ルームナンバーリスト。
callback	RoomInfoCallback	ルーム詳細情報のコールバック。

getAnchorList

ルーム内の全キャスターのリストを取得します。 `enterRoom()` 成功後に呼び出しが有効となります。



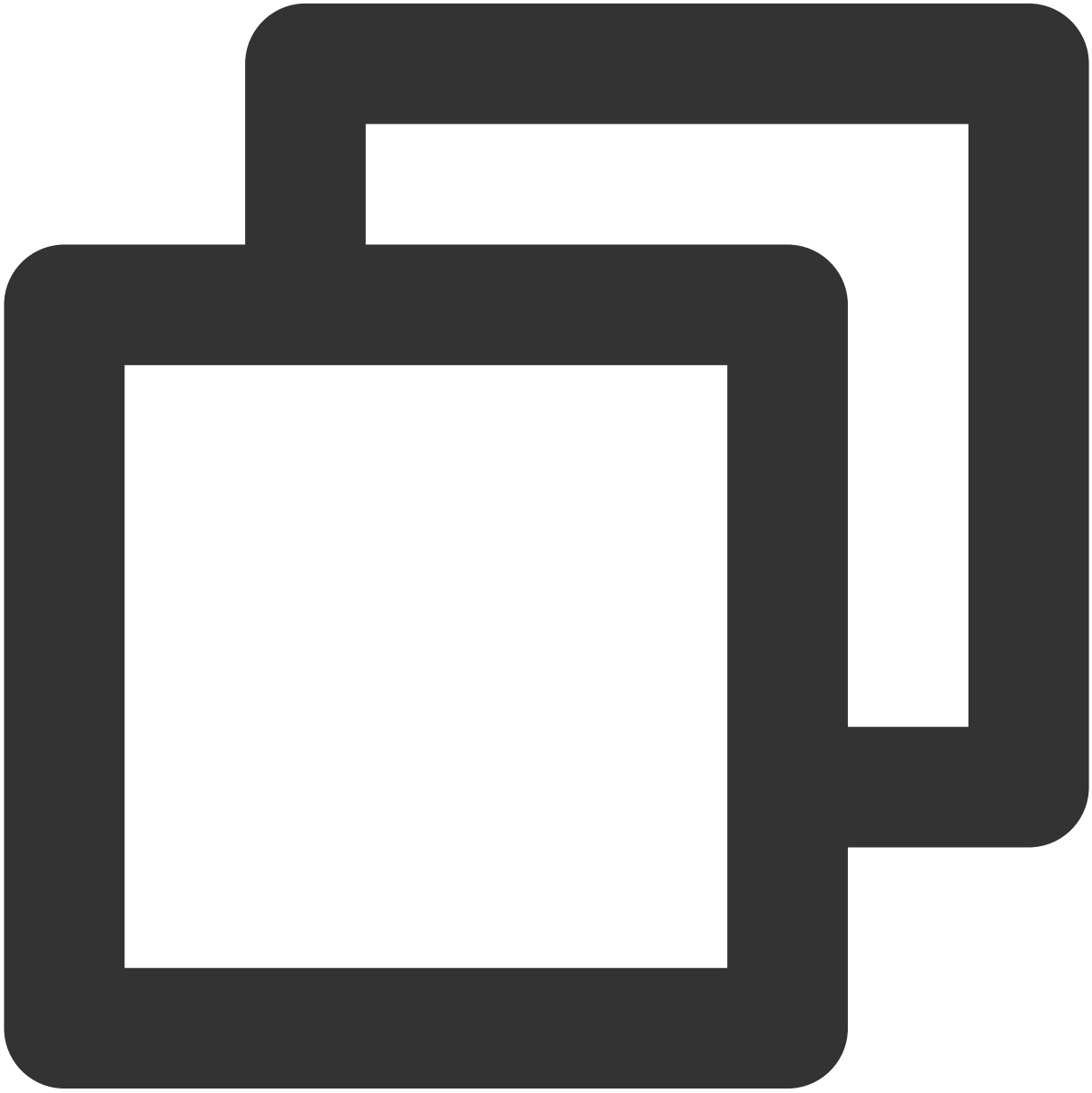
```
public abstract void getAnchorList(TRTCLiveRoomCallback.UserListCallback callback);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
callback	UserListCallback	ユーザーの詳細情報のコールバック。

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
```

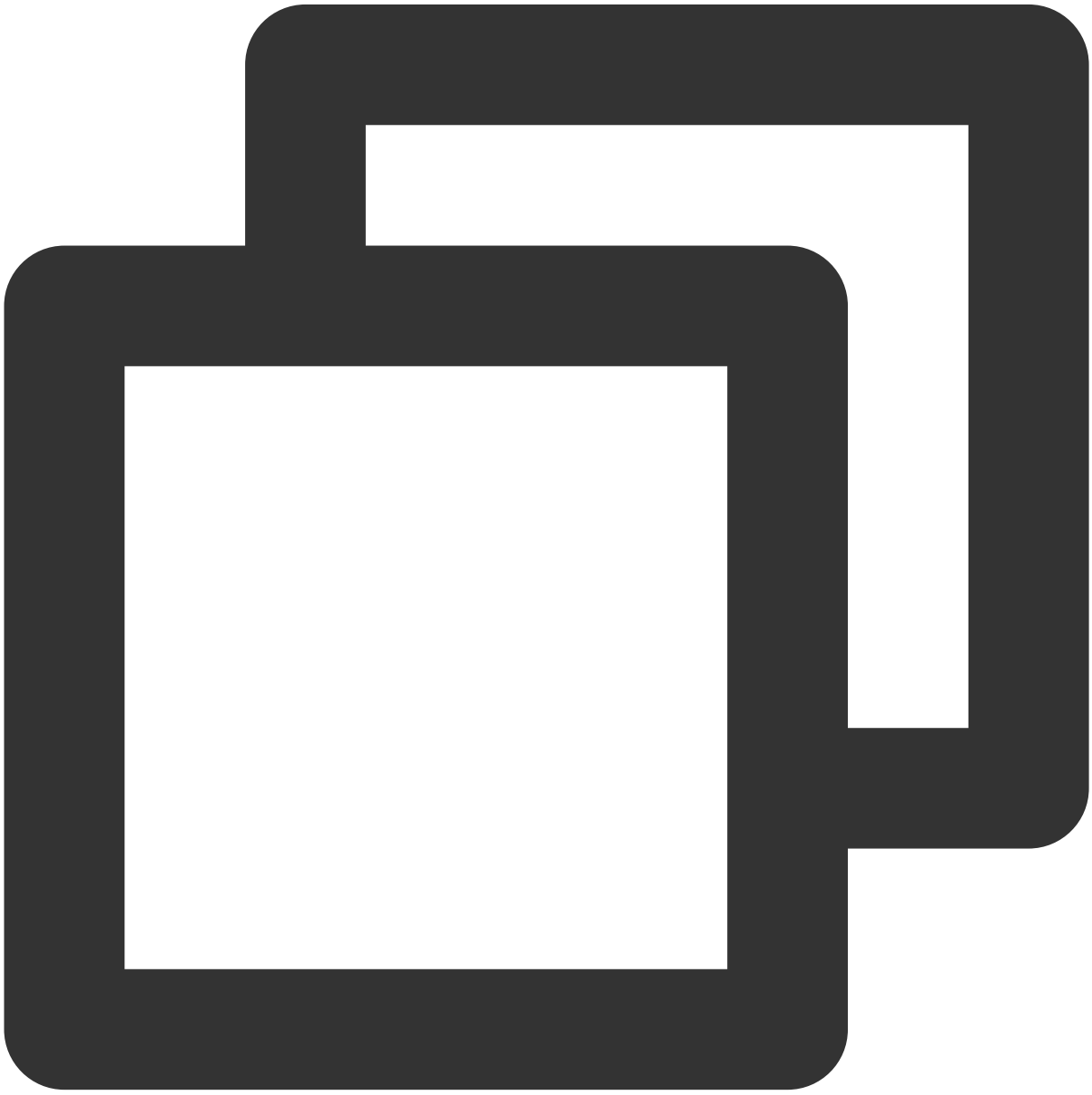
パラメータは下表に示すとおりです：

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。



©2013-2022 Tencent Cloud. All rights reserved.

```
public abstract void startCameraPreview(boolean isFront, TXCloudVideoView view, TRT
```

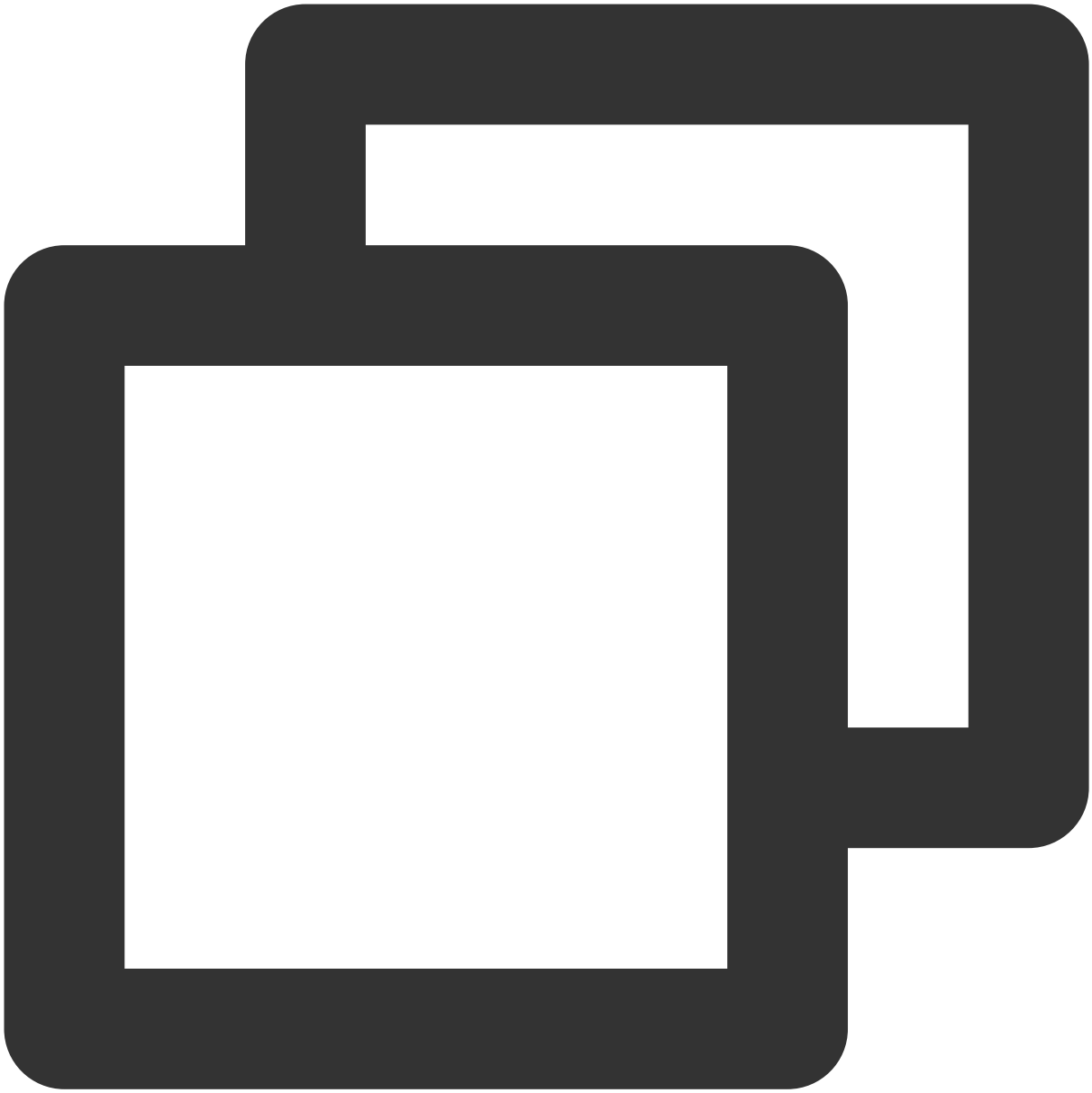
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
isFront	boolean	true：フロントカメラ、false：リアカメラ。
view	TXCloudVideoView	ビデオ画像をロードするウィジェット。
callback	ActionCallback	操作コールバック。

stopCameraPreview

ローカルのビデオキャプチャおよびプレビューを停止します。

©2013-2022 Tencent Cloud. All rights reserved.

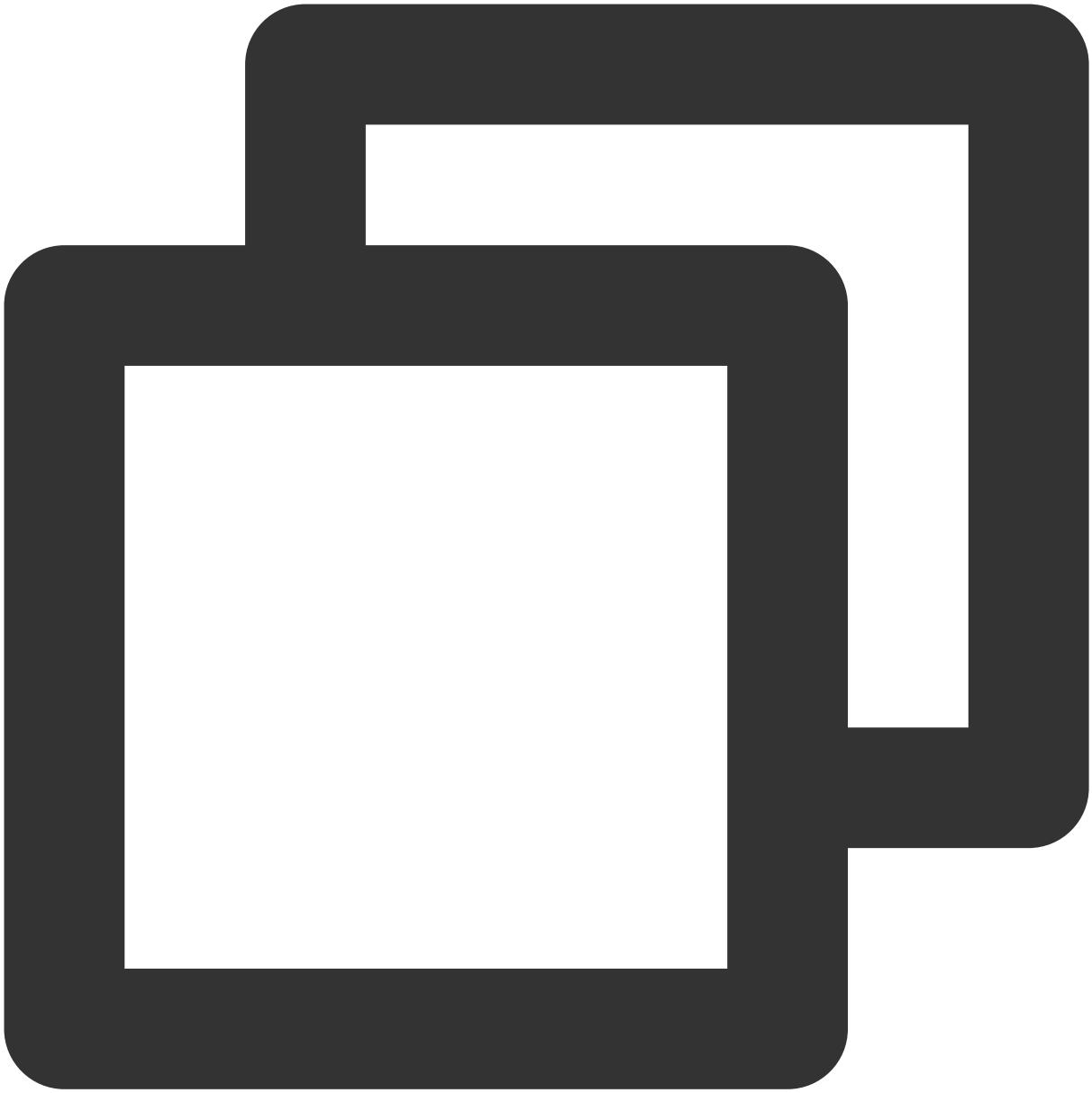


```
public abstract void stopCameraPreview();
```

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

©2013-2022 Tencent Cloud. All rights reserved.



```
public abstract void startPublish(String streamId, TRTCLiveRoomCallback.ActionCallb
```

パラメータは下表に示すとおりです：

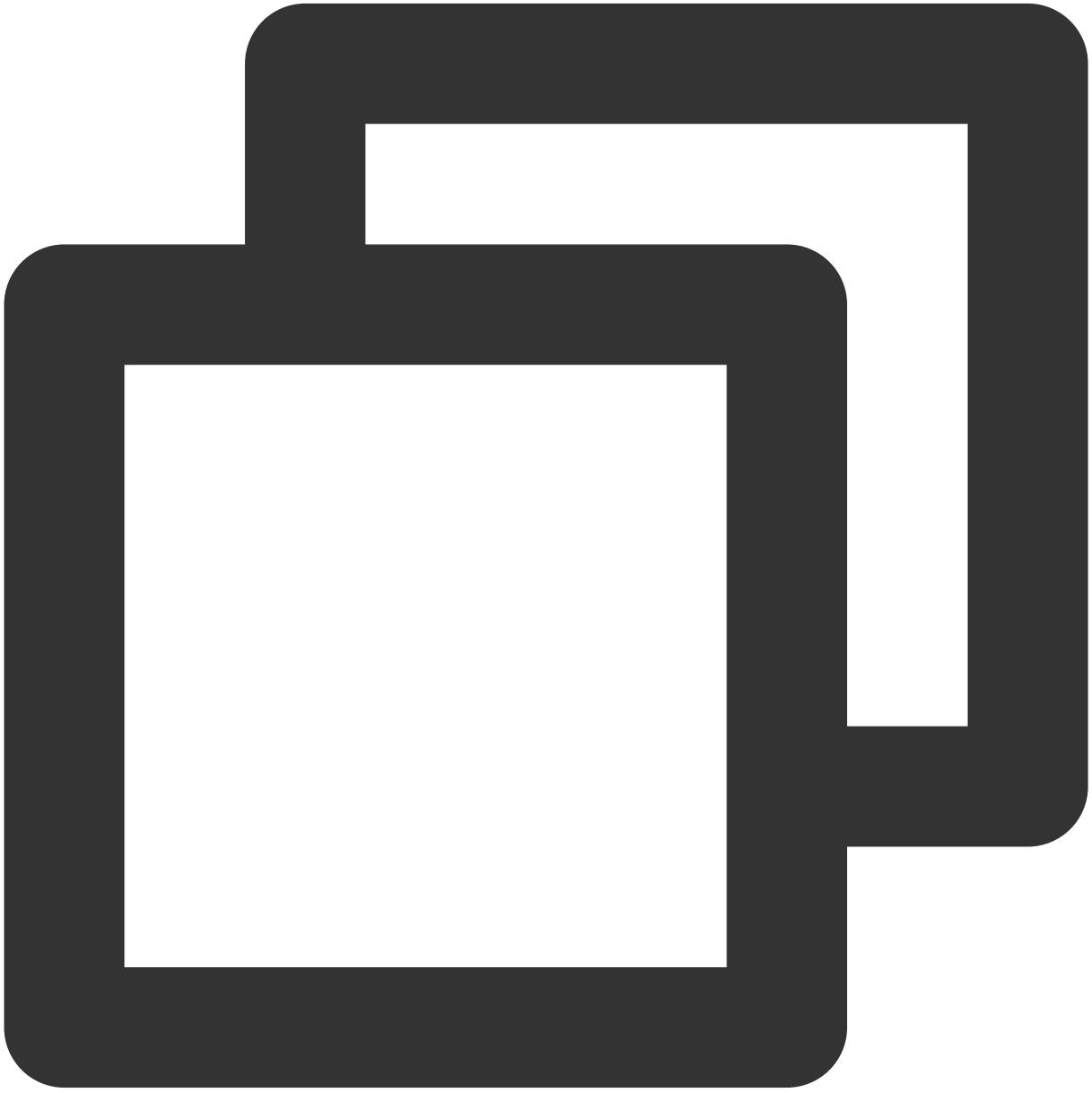
パラメータ	タイプ	意味
streamId	String	ライブCDNのstreamIdをバインドするために利用します。視聴者にライブCDN経由で視聴させたい場合は、現在のキャスターのライブストリーミングstreamIdを指定する必要があります。

©2013-2022 Tencent Cloud. All rights reserved.

callback	ActionCallback	操作コールバック。

stopPublish

ライブストリーミング（プッシュ）を停止します。次のシーンに適用します：
キャスターがライブストリーミングを終了する時の呼び出し
視聴者がマイク接続を終了する時の呼び出し



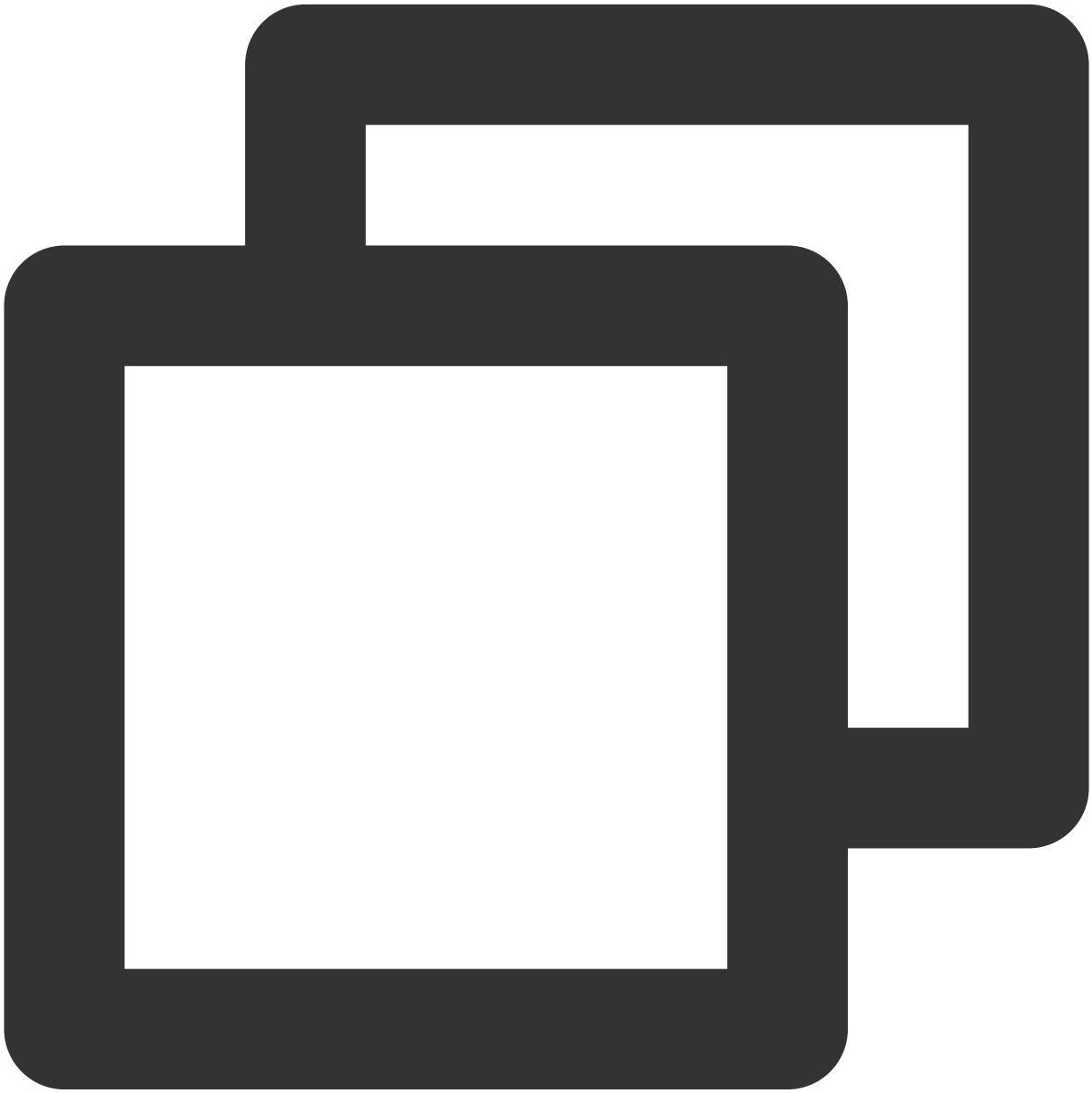
```
public abstract void stopPublish(TRTCLiveRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

startPlay

リモートのビデオ画面を再生します。普通の視聴とマイク接続のシーンで呼び出すことができます。



```
public abstract void startPlay(String userId, TXCloudVideoView view, TRTCLiveRoomCa
```

パラメ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。

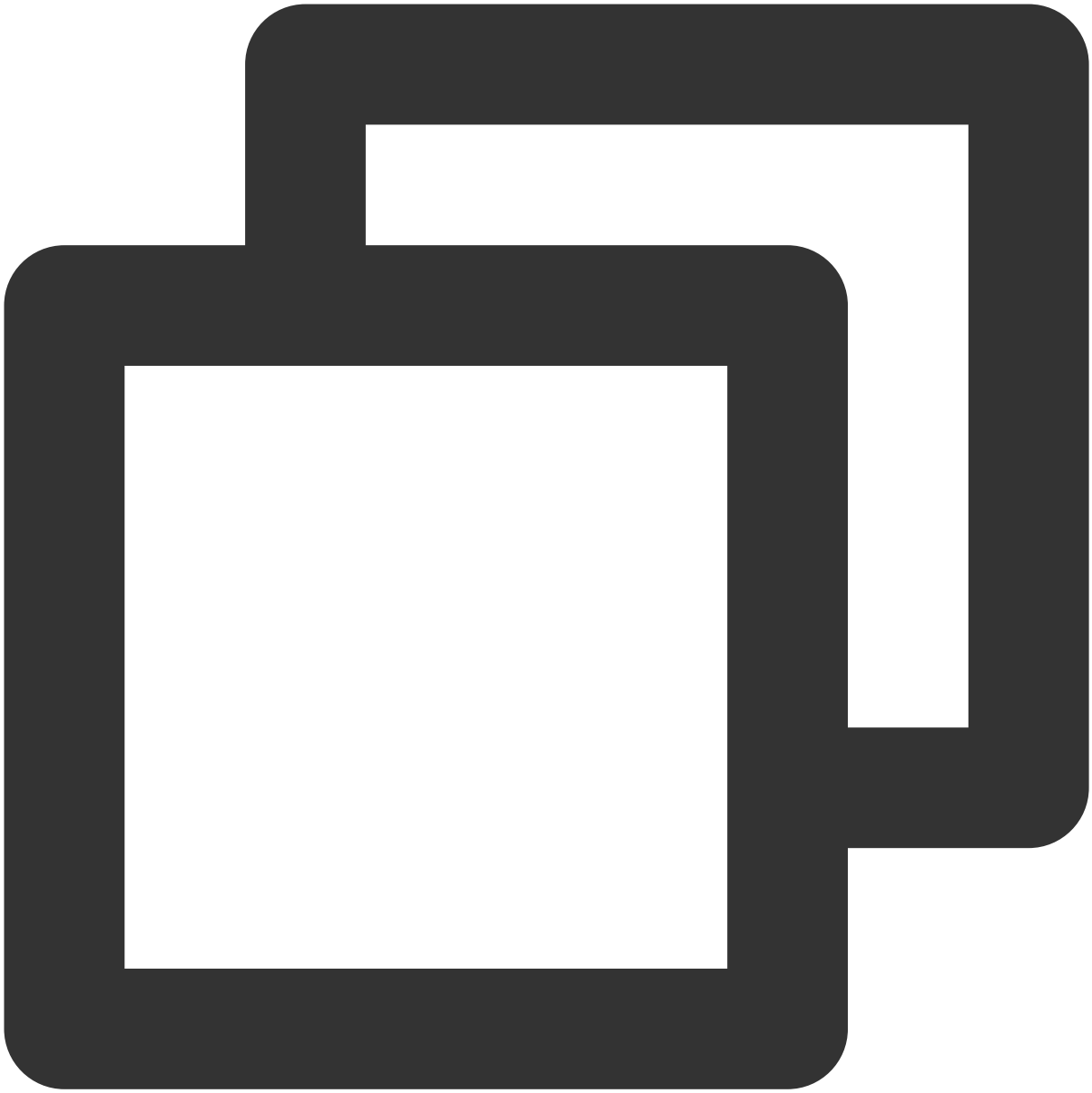
ライブストリーミングのマイク接続シーン

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。

©2013-2022 Tencent Cloud. All rights reserved.



```
public abstract void stopPlay(String userId, TRTCLiveRoomCallback.ActionCallback ca
```

パラメータは下表に示すとおりです：

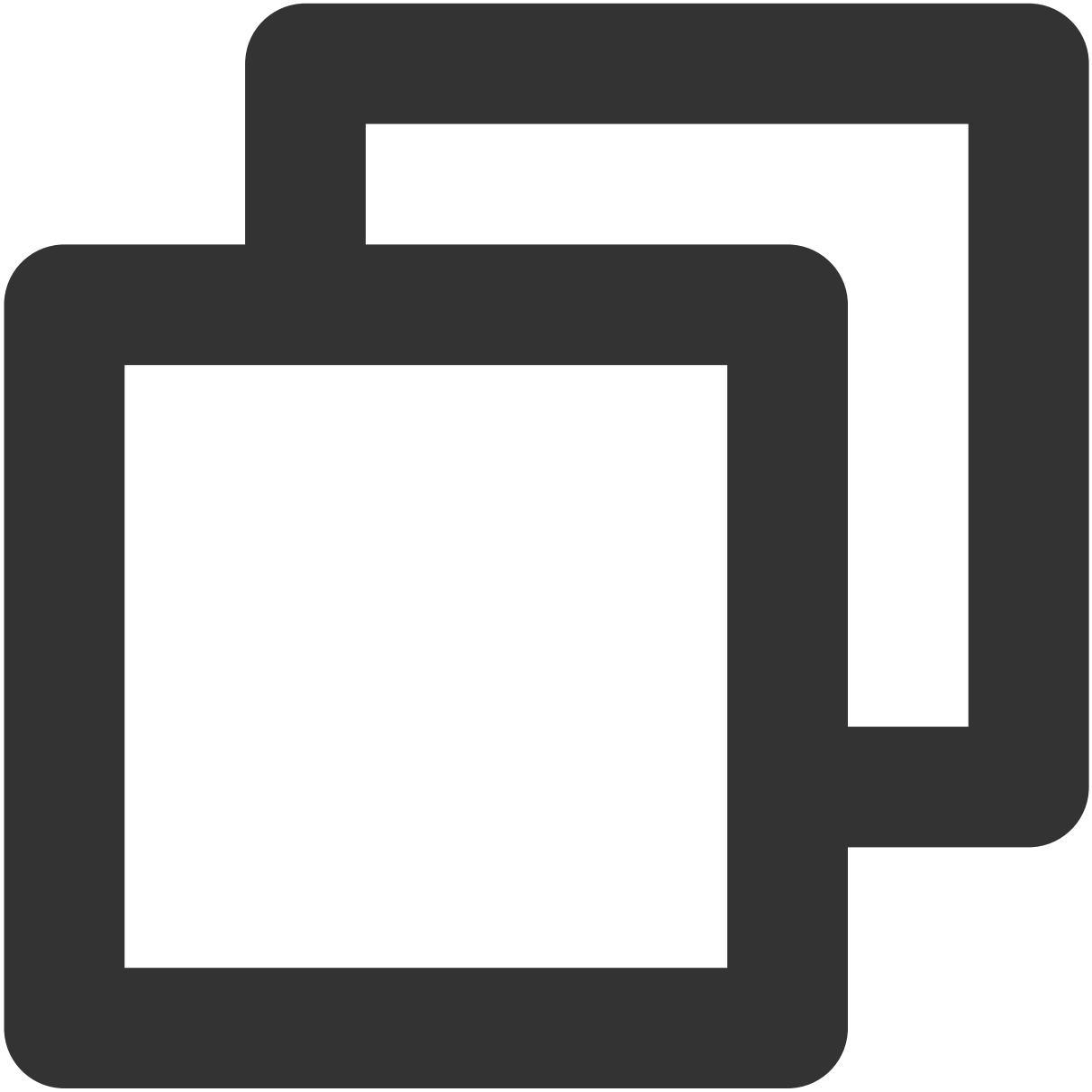
パラメータ	タイプ	意味
userId	String	相手側のユーザー情報。
callback	ActionCallback	操作コールバック。

©2013-2022 Tencent Cloud. All rights reserved.

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, TRTCLiveRoomCall
```

パラ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.

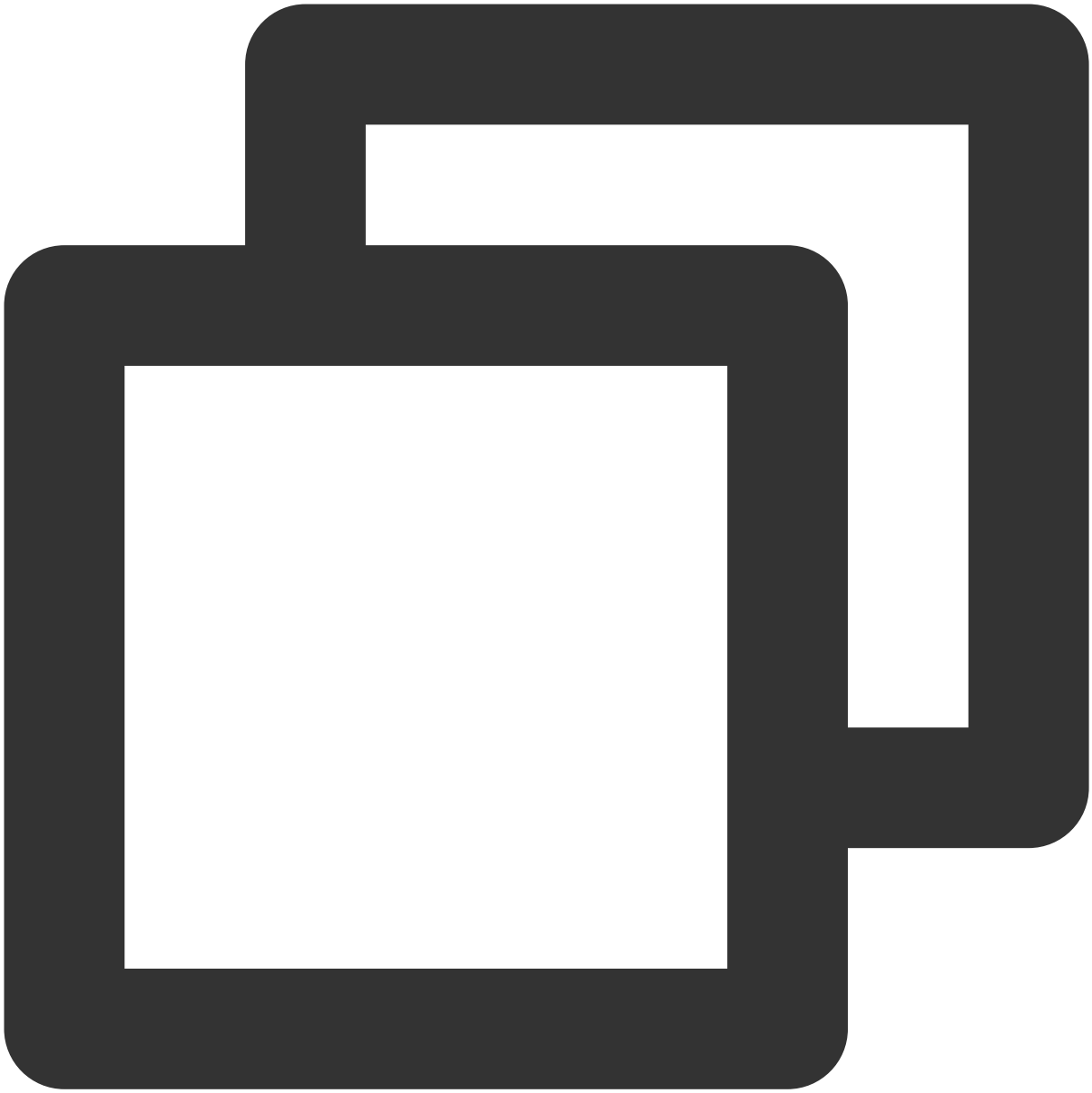
reason	String	マイク接続の理由。
timeout	int	タイムアウトの時間。
callback	ActionCallback	キャスターの応答のコールバック。

キャスターと視聴者のマイク接続フローは次のとおりです：
1. 【視聴者】 `requestJoinAnchor()` を呼び出し、キャスターにマイク接続リクエストを送信します。
2. 【キャスター】 `TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバック通知を受信します。
3. 【キャスター】 `responseJoinAnchor()` を呼び出して、視聴者からのマイク接続リクエストを受け入れるか決定します。
4. 【視聴者】responseCallbackのコールバック通知を受信します。この通知にキャスターの処理結果が含まれています。
5. 【視聴者】リクエストが同意されると、 `startCameraPreview()` が呼び出され、ローカルカメラが起動します。
6. 【視聴者】その後 `startPublish()` を呼び出し、正式にプッシュストリームの状態に入ります。
7. 【キャスター】一度視聴者がマイク接続に入ると、キャスターは、 `TRTCLiveRoomDelegate` の `onAnchorEnter()` の通知を受信します。
8. 【キャスター】キャスターが `startPlay()` を呼び出すと、マイク接続の視聴者のビデオ画面を見ることができるようになります。
9. 【視聴者】ライブストリーミングルームの中で他の視聴者がすでにキャスターとマイク接続を行っている場合、新しく参加したマイク接続の視聴者は `onAnchorEnter()` の通知を受け取ります。この時に `startPlay()` を呼び出せば他のマイク接続者のビデオ画面を再生することができます。

responseJoinAnchor

キャスターがマイク接続のリクエストを処理します。キャスターは、 `TRTCLiveRoomDelegate` の `onRequestJoinAnchor()` のコールバックを受け取った後、このインターフェースを呼び出して、視聴者のマイク接続リクエストを処理してください。

©2013-2022 Tencent Cloud. All rights reserved.



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

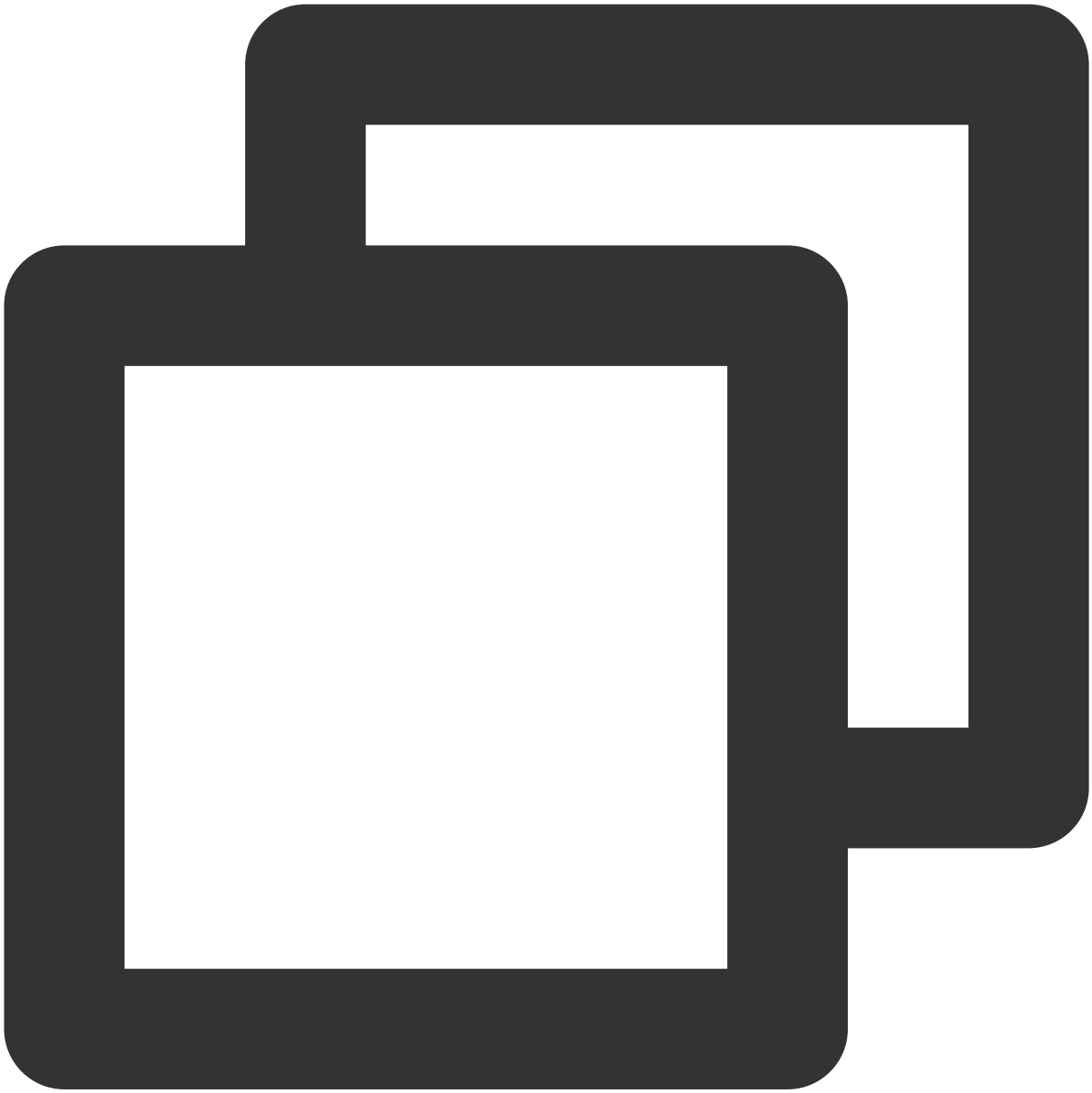
パラメータは下表に示すとおりです：

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。

©2013-2022 Tencent Cloud. All rights reserved.

kickoutJoinAnchor

キャスターがマイク接続の視聴者をキックアウトします。キャスターがこのインターフェースを呼び出し、マイク接続の視聴者をキックアウトすると、キックアウトされたマイク接続の視聴者は、`TRTCLiveRoomDelegate` の `onKickoutJoinAnchor()` のコールバック通知を受信します。



```
public abstract void kickoutJoinAnchor(String userId, TRTCLiveRoomCallback.ActionCa
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

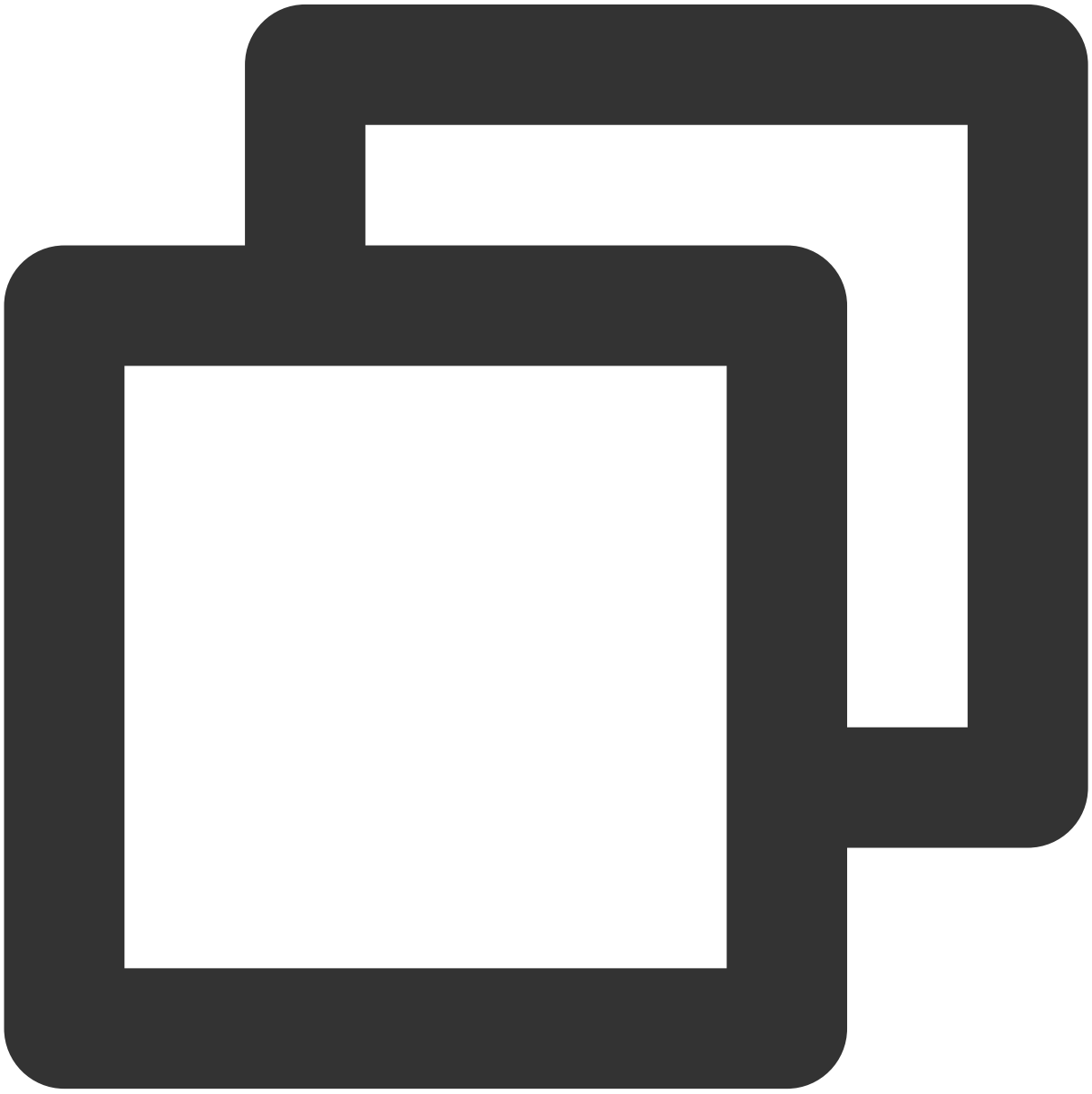
©2013-2022 Tencent Cloud. All rights reserved.

userId	String	マイク接続の視聴者 ID。
callback	ActionCallback	操作コールバック。

キャスターのルーム間PK

requestRoomPK

キャスターがルーム間PKをリクエストします。



©2013-2022 Tencent Cloud. All rights reserved.

```
public abstract void requestRoomPK(int roomId, String userId, TRTCLiveRoomCallback.
```

パラメータは下表に示すとおりです：

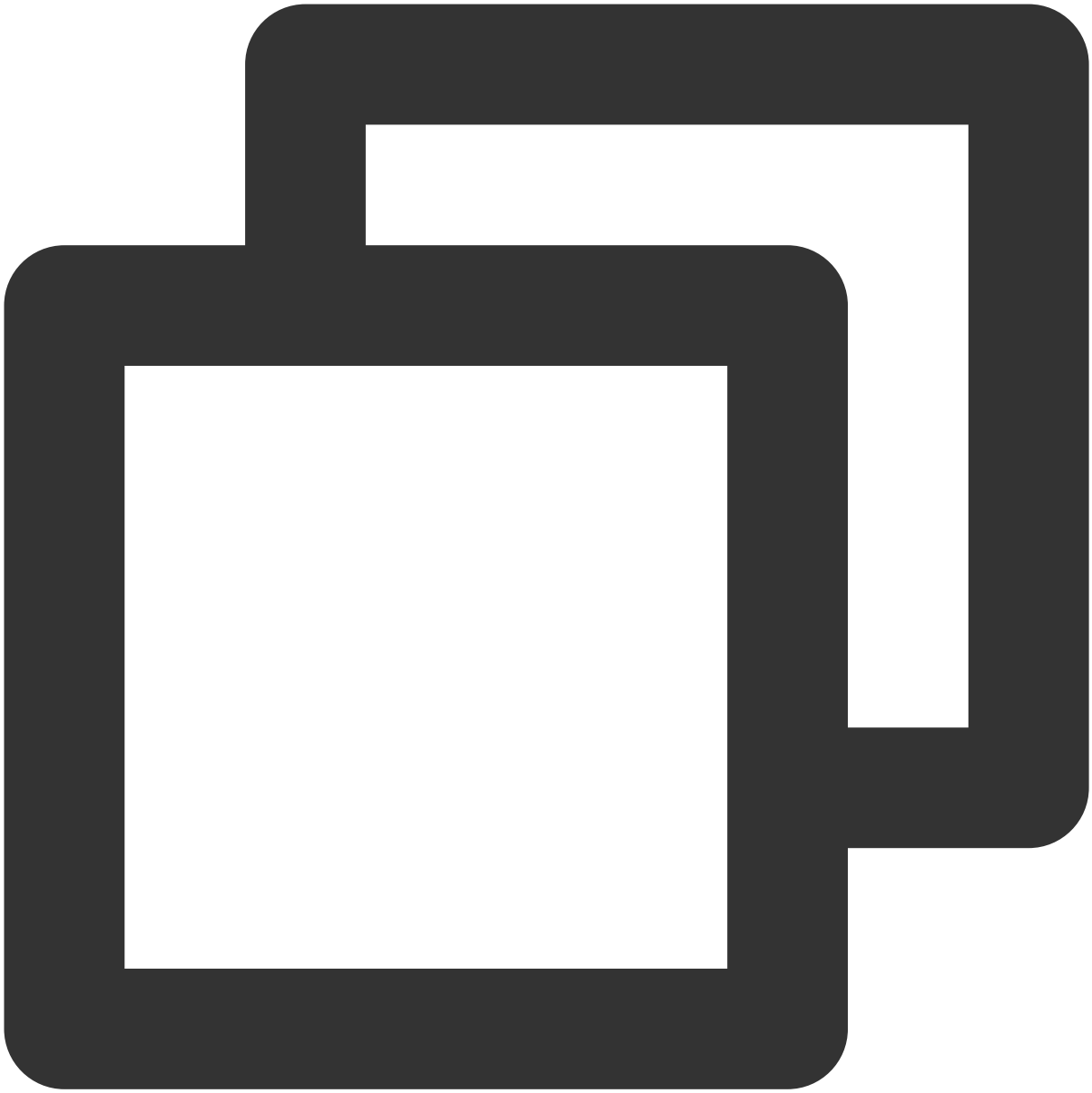
パラメータ	タイプ	意味
roomId	int	招待された側のルーム ID。
userId	String	招待された側のキャスター ID。
callback	ActionCallback	ルーム間PKのリクエスト結果のコールバック。

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。

©2013-2022 Tencent Cloud. All rights reserved.



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

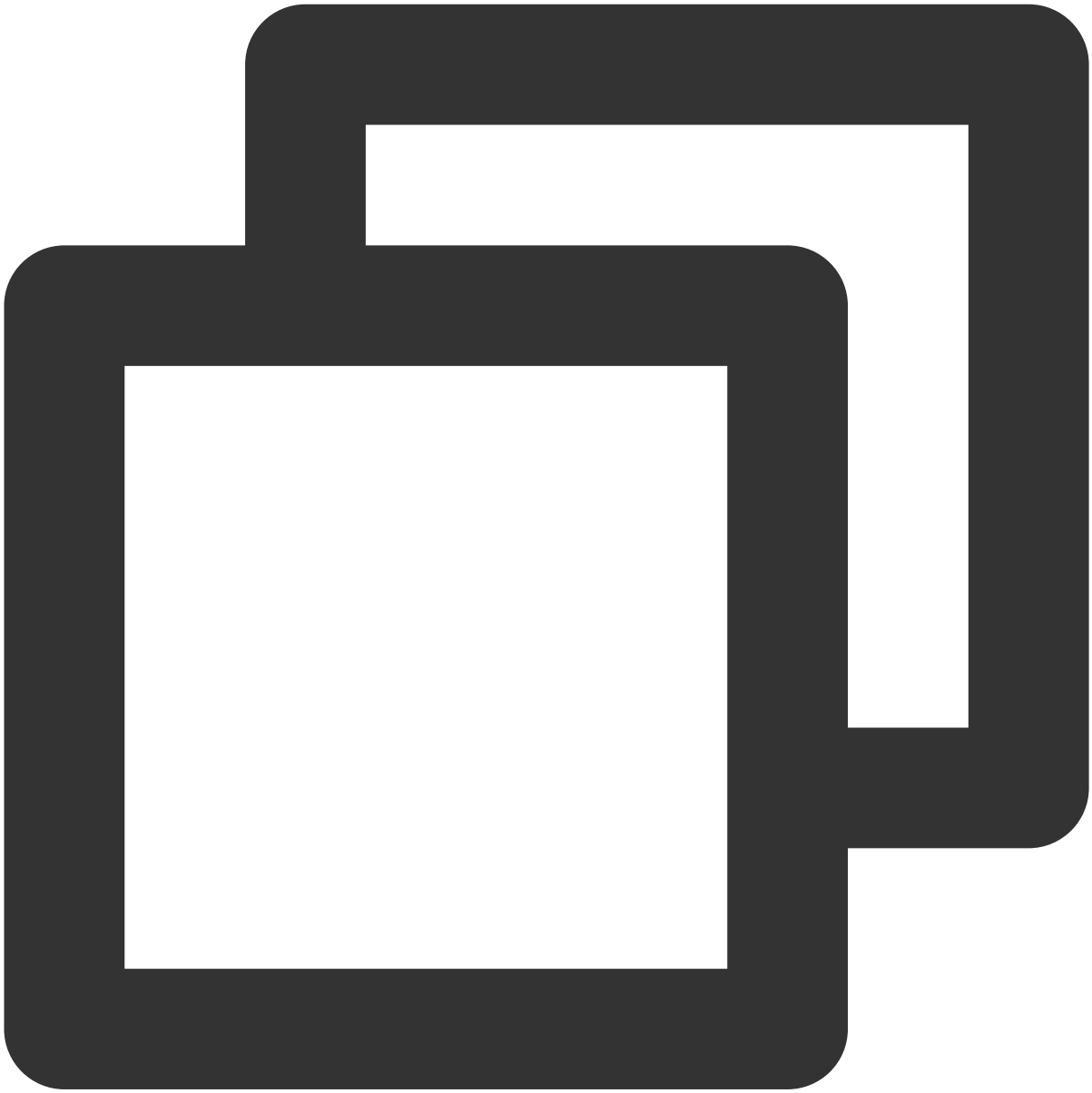
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userId	String	PKをリクエストしたキャスターID。
agree	boolean	true：同意；false：拒否。
reason	String	PKを同意/拒否した理由の説明。

©2013-2022 Tencent Cloud. All rights reserved.

quitRoomPK

ルーム間PKから退出します。PK中のいずれかのキャスターがルーム間PKの状態から退出すると、もう一方のキャスターは `TRTCLiveRoomDelegate` の `onQuitRoomPk()` のコールバック通知を受信します。



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

パラメータは下表に示すとおりです：

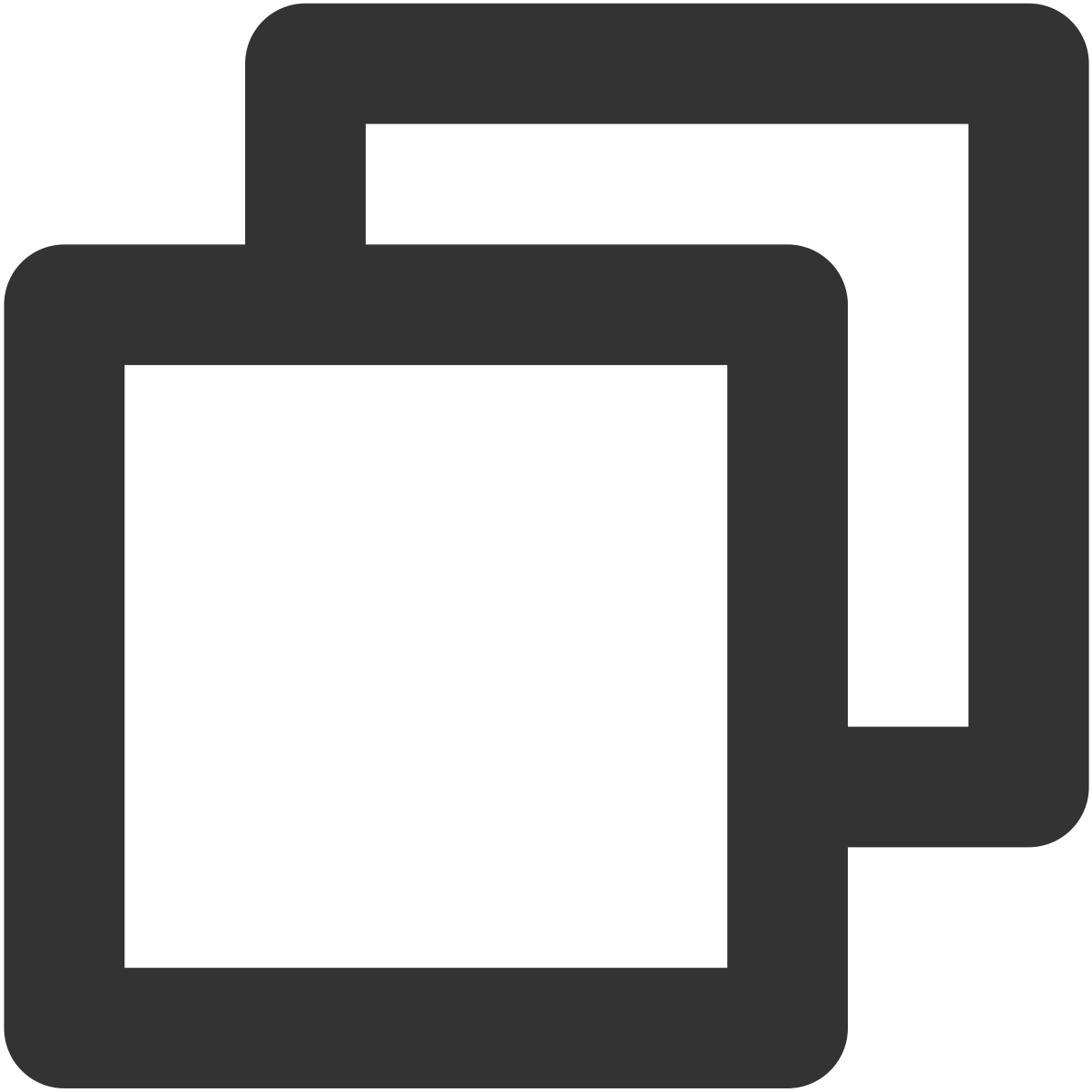
パラメータ	タイプ	意味
callback	ActionCallback	操作コールバック。

©2013-2022 Tencent Cloud. All rights reserved.

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。

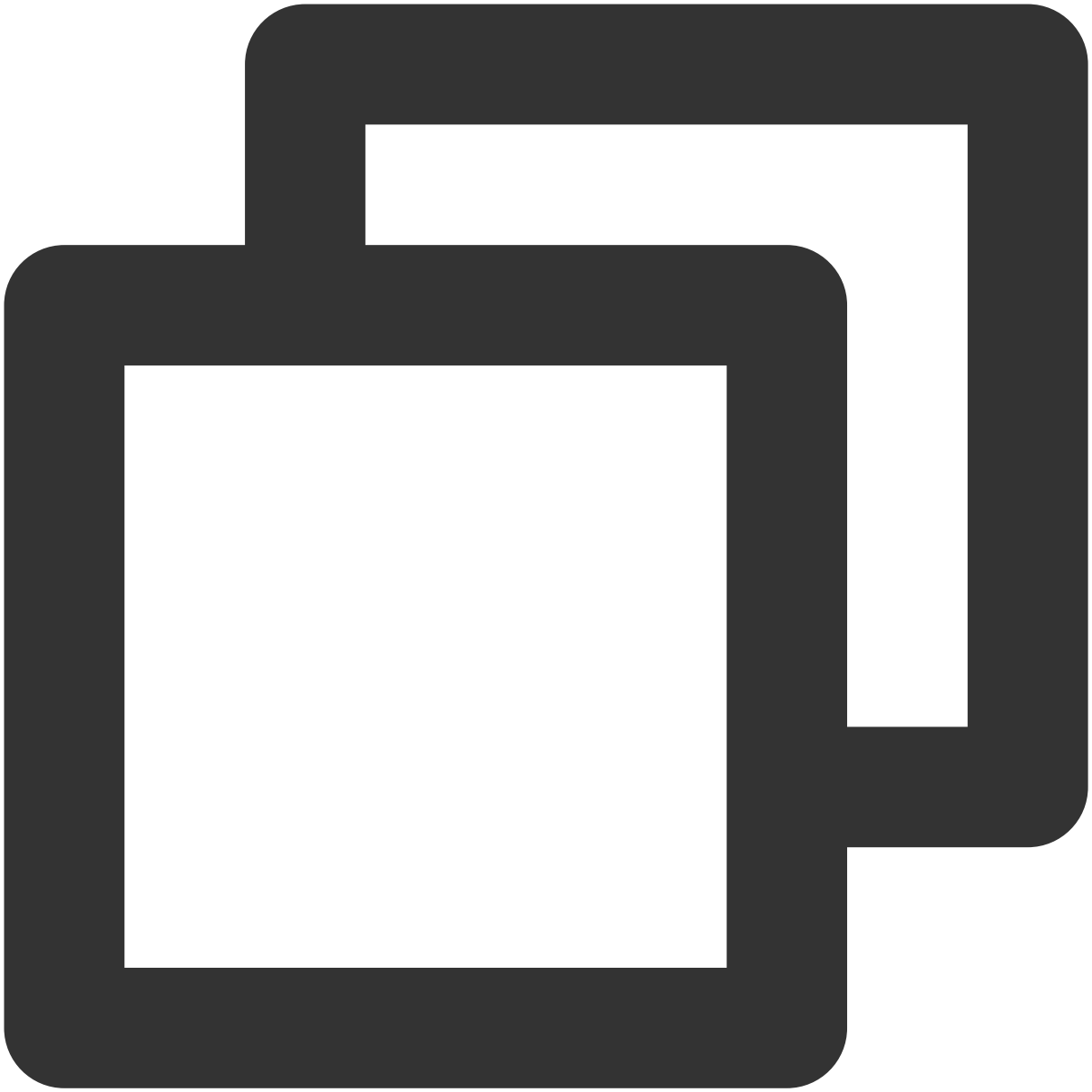


```
public abstract void switchCamera();
```

setMirror

ミラー表示のオン/オフを設定します。

©2013-2022 Tencent Cloud. All rights reserved.



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

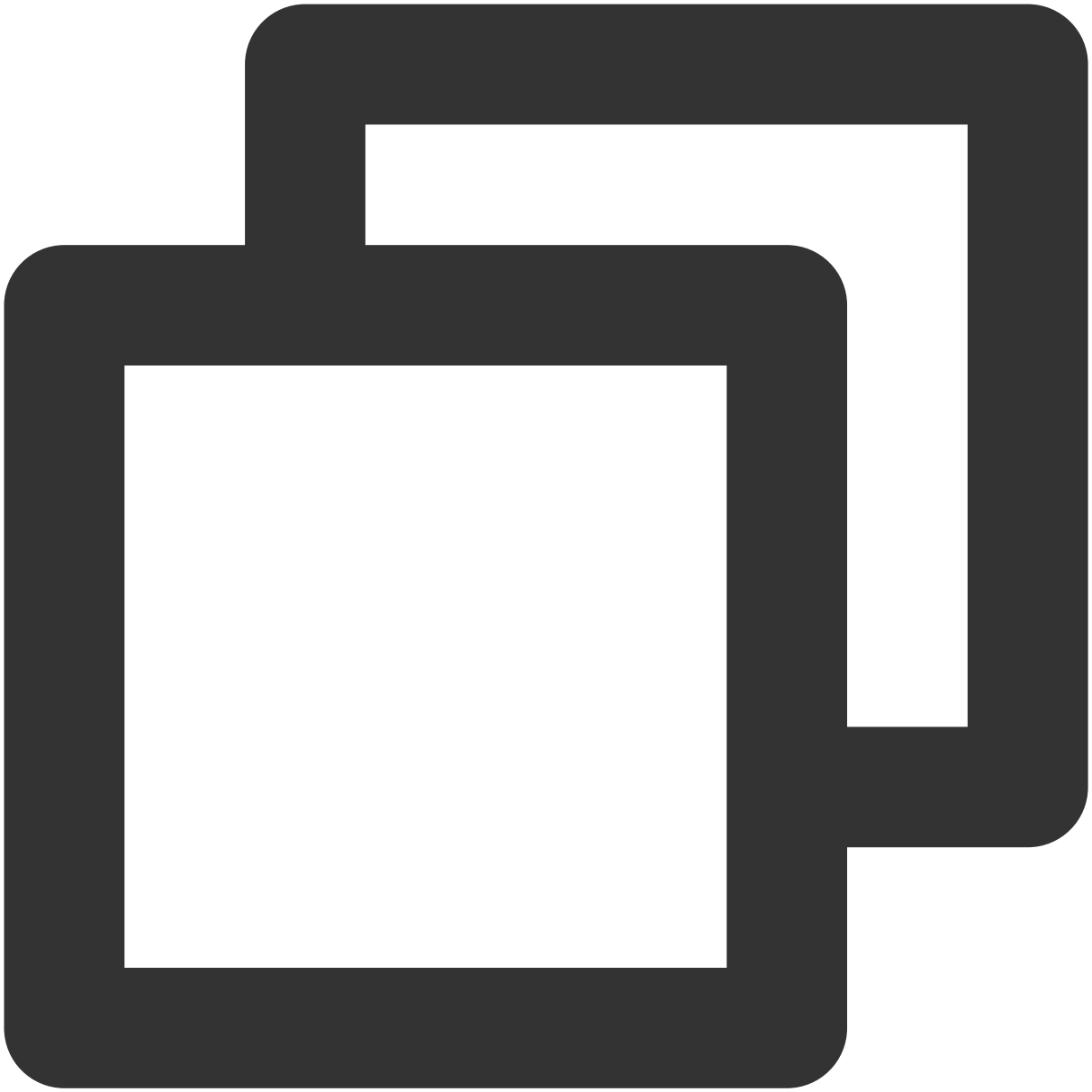
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
isMirror	boolean	ミラーオン/オフ。

muteLocalAudio

ローカルオーディオをミュートにします。

©2013-2022 Tencent Cloud. All rights reserved.



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

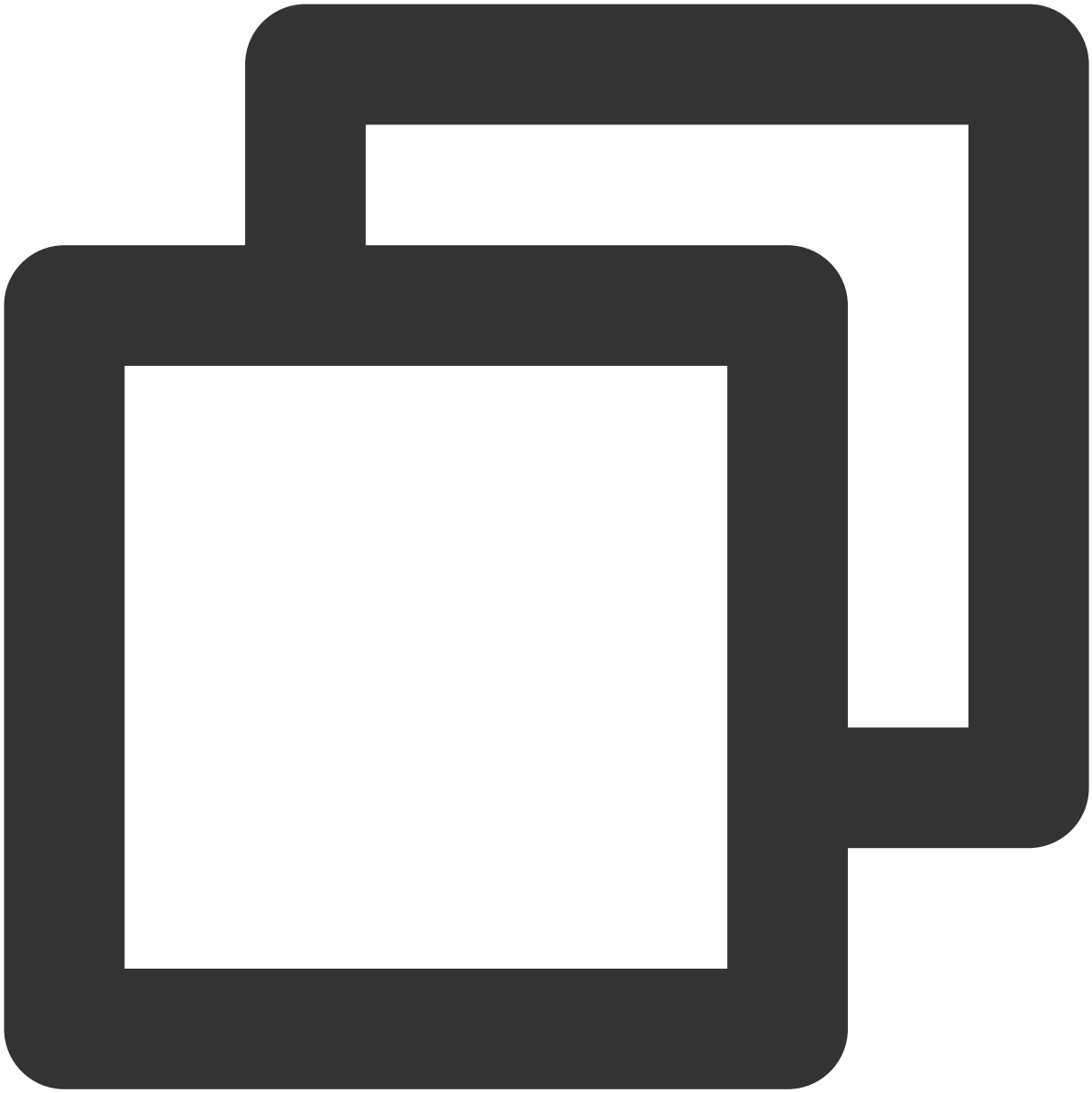
パラメータは下表に示すとおりです：

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。

©2013-2022 Tencent Cloud. All rights reserved.



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

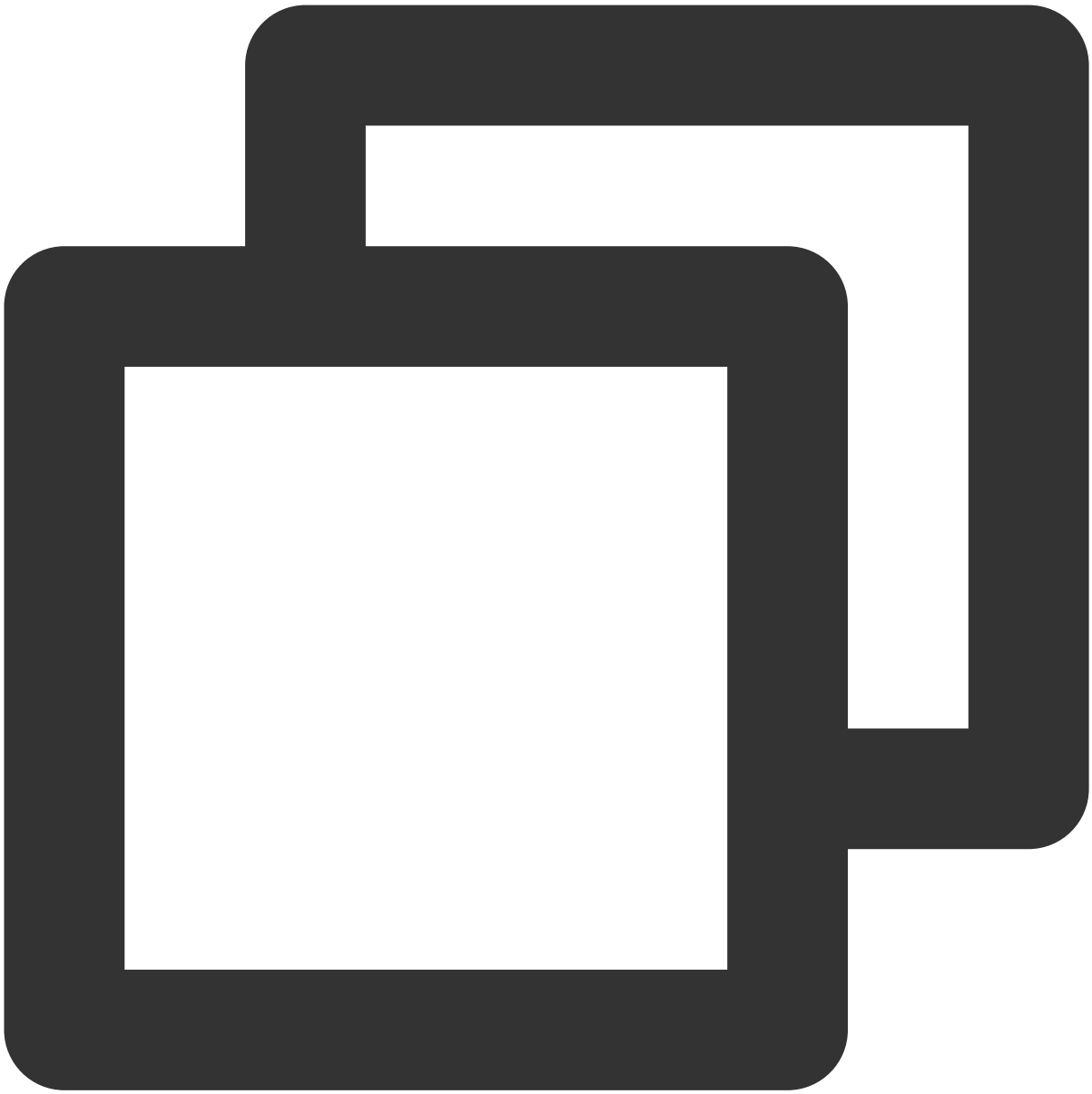
パラメータは下表に示すとおりです：

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

©2013-2022 Tencent Cloud. All rights reserved.

すべてのリモート側のオーディオをミュートにします。



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

パラメータは下表に示すとおりです：

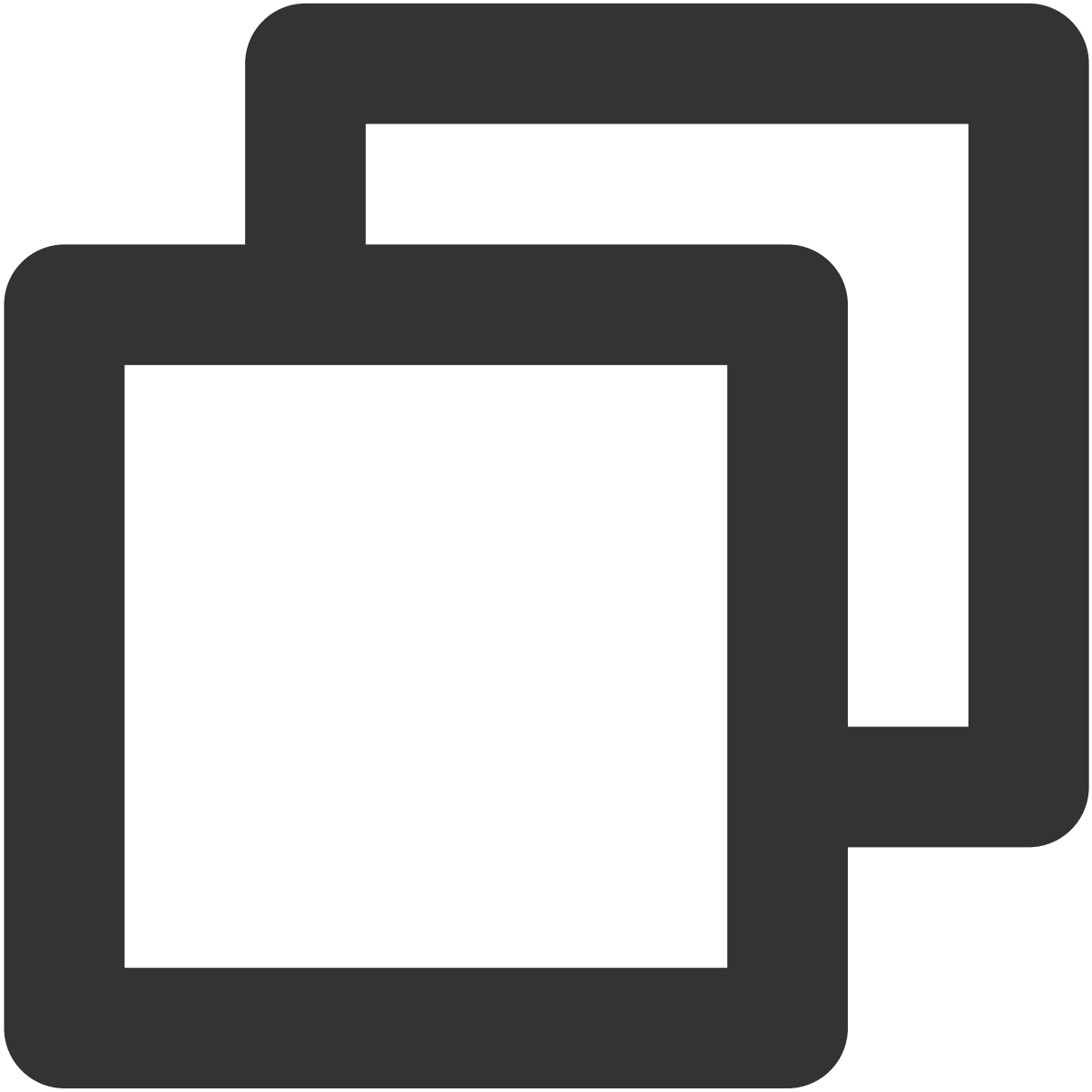
パラメータ	タイプ	意味
mute	boolean	true：ミュート起動、false：ミュート停止。

©2013-2022 Tencent Cloud. All rights reserved.

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。



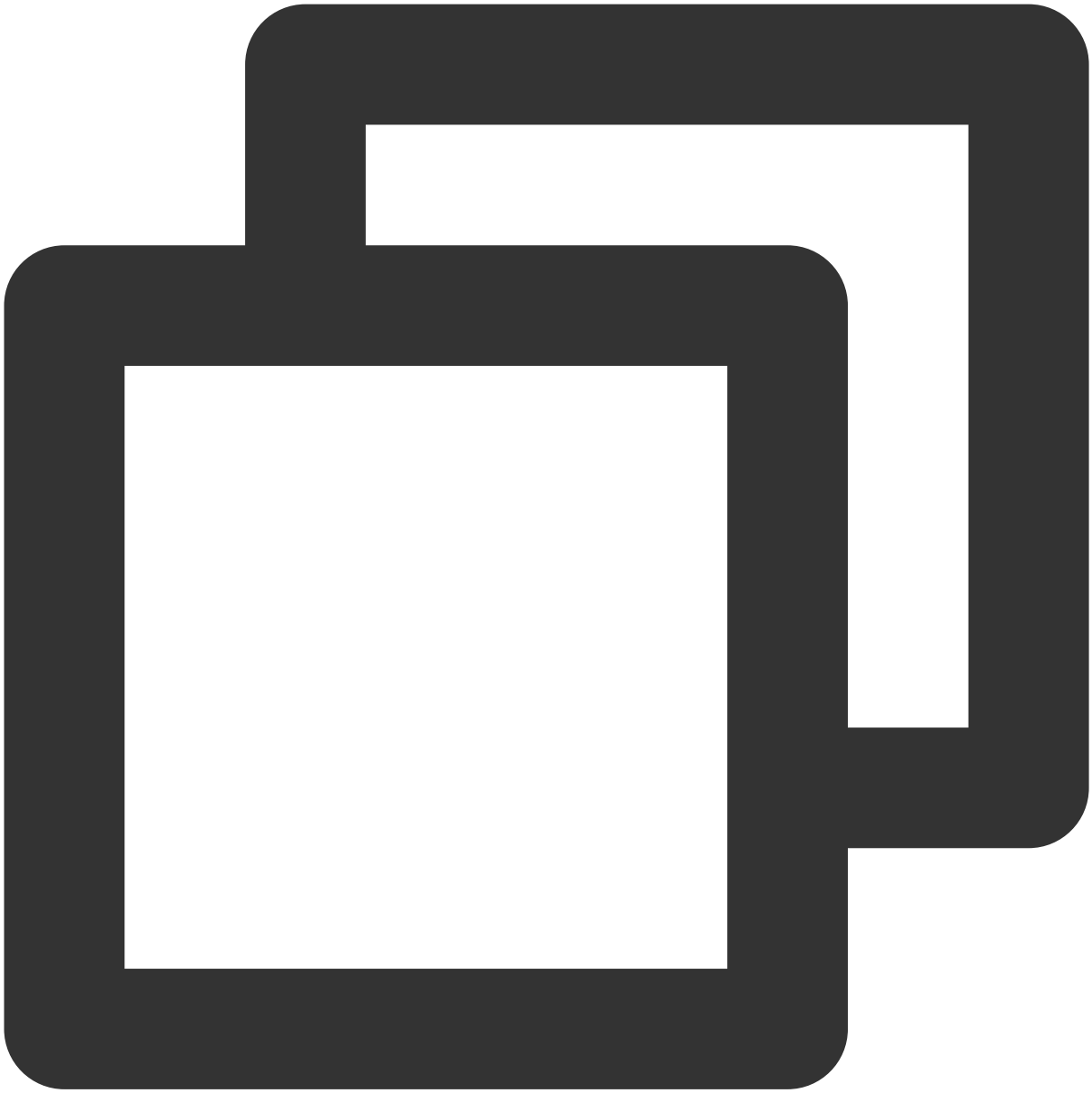
```
public abstract TXAudioEffectManager getAudioEffectManager();
```

美顔フィルタに関するインターフェース関数

©2013-2022 Tencent Cloud. All rights reserved.

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。

©2013-2022 Tencent Cloud. All rights reserved.

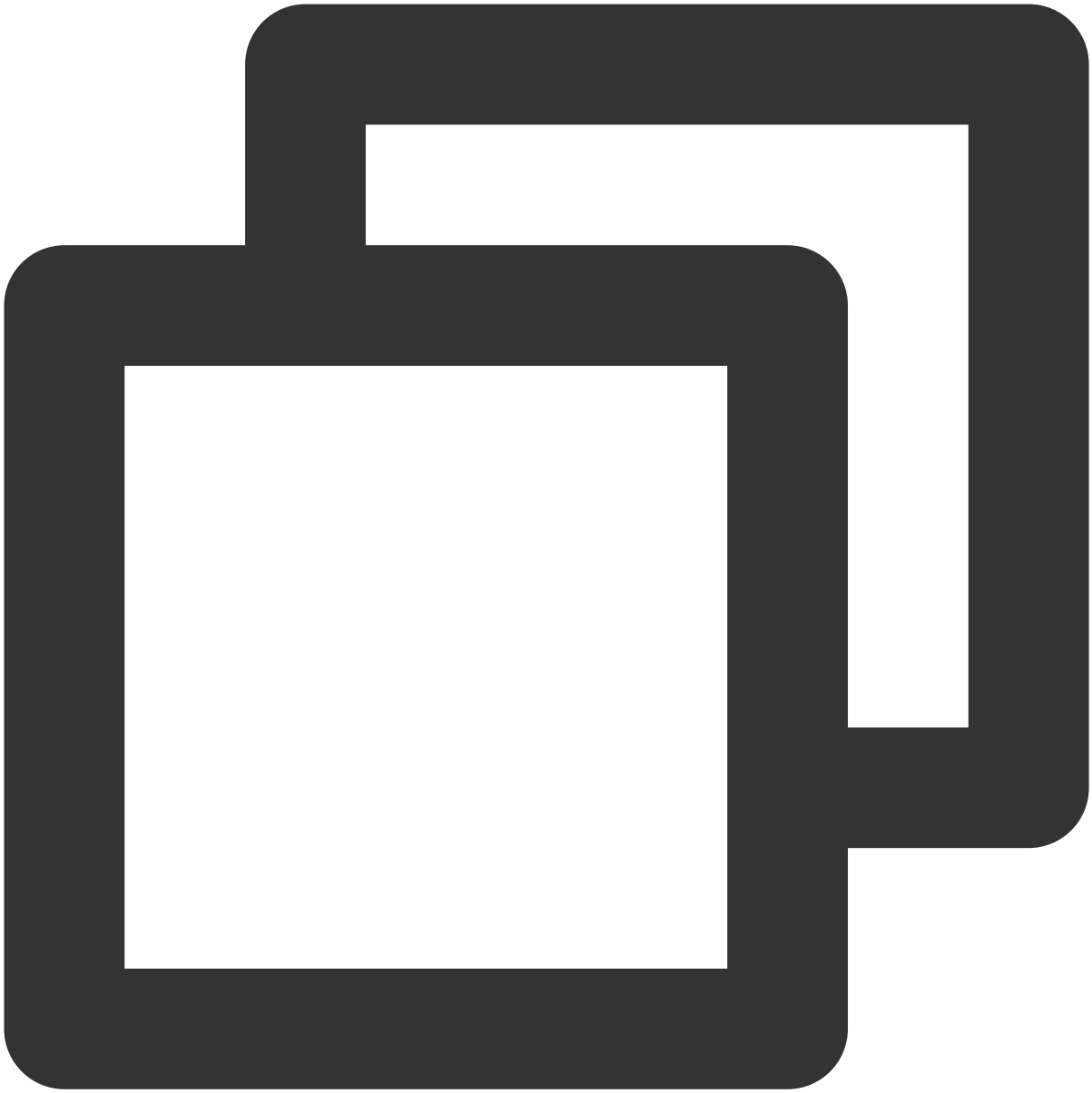
「髪の生え際」、「眼と眼の距離」、「眼角度」、「唇の形」、「鼻翼」、「鼻の位置」、「唇の厚さ」、「顔の形」を調整します。
人の顔のスタンプ（素材）等のダイナミック効果を設定します。
メイクアップを追加します。
ジェスチャー認識を行います。

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。

©2013-2022 Tencent Cloud. All rights reserved.



```
public abstract void sendRoomTextMsg(String message, TRTCLiveRoomCallback.ActionCal
```

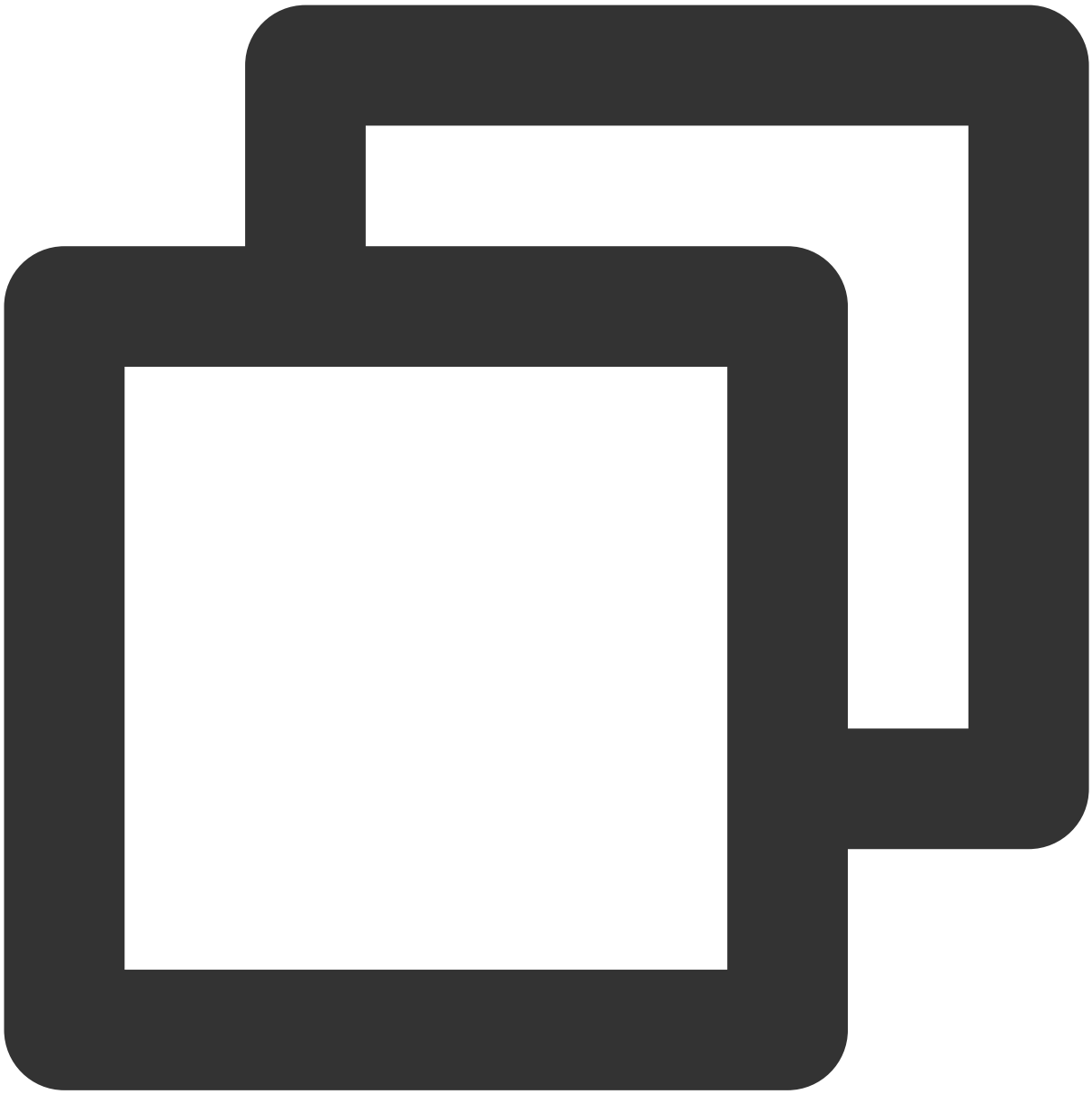
パラメータは下表に示すとおりです：

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, TRTCLiveRoomCall
```

パラ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。

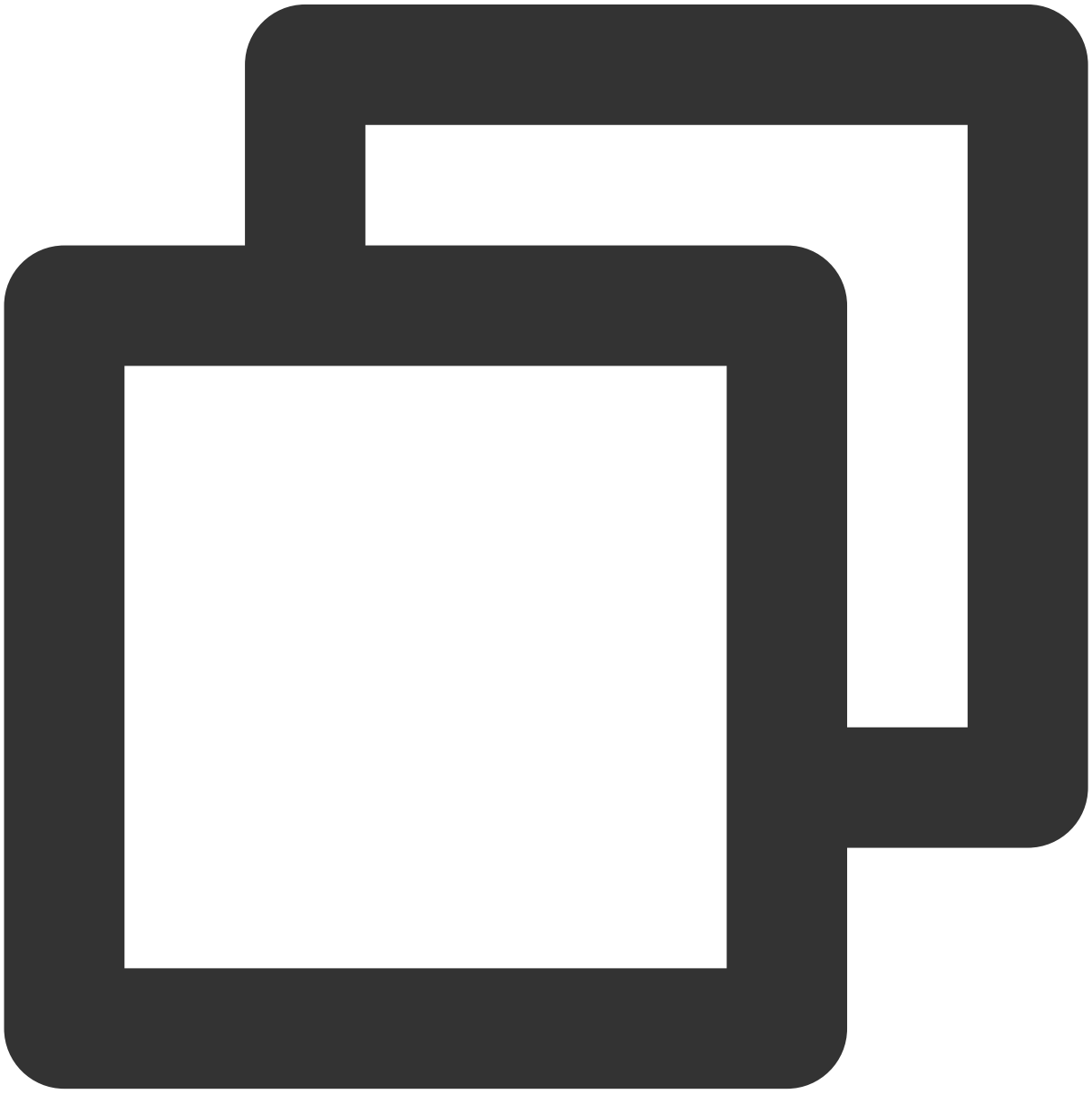
©2013-2022 Tencent Cloud. All rights reserved.

message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

デバックに関するインターフェース関数

showVideoDebugLog

インターフェースの中にdebug情報を表示するかどうか。



©2013-2022 Tencent Cloud. All rights reserved.

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

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
isShow	boolean	Debug情報表示のオン/オフ。

TRTCLiveRoomDelegateイベントのコールバック

一般的なイベントコールバック

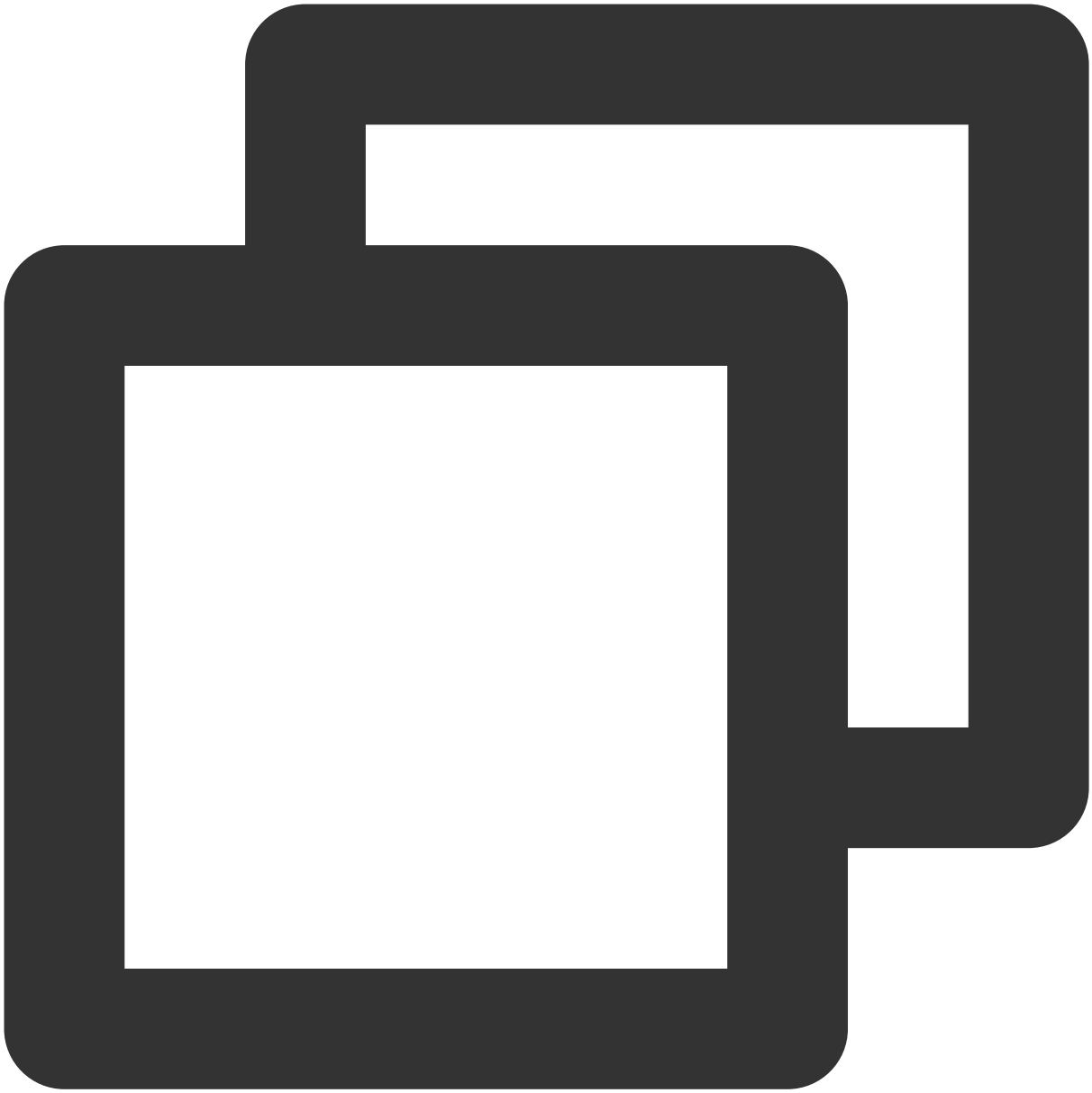
onError

エラーのコールバック。

explain

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

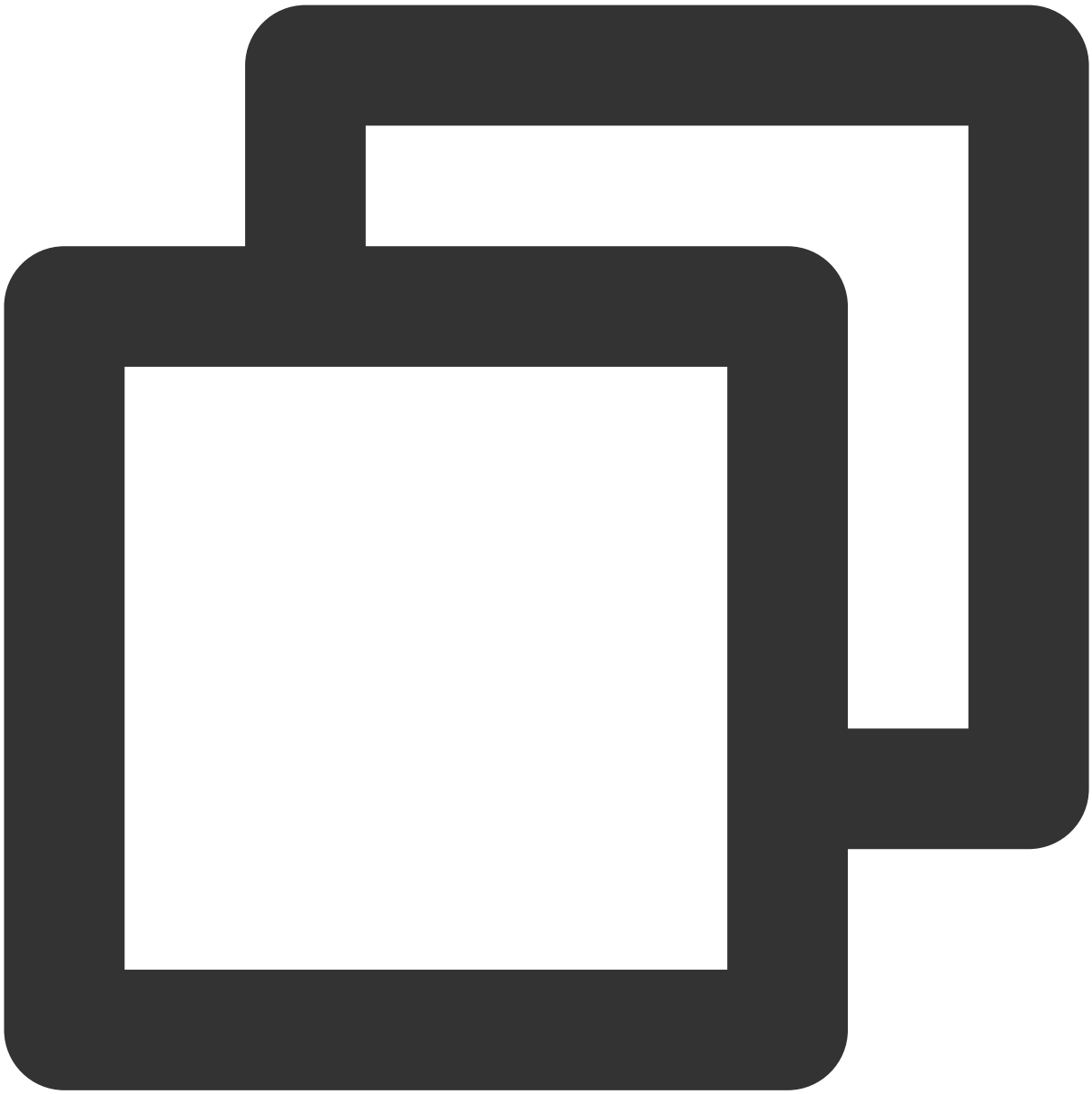
パラメータは下表に示すとおりです：

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

©2013-2022 Tencent Cloud. All rights reserved.

警告のコールバック。



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

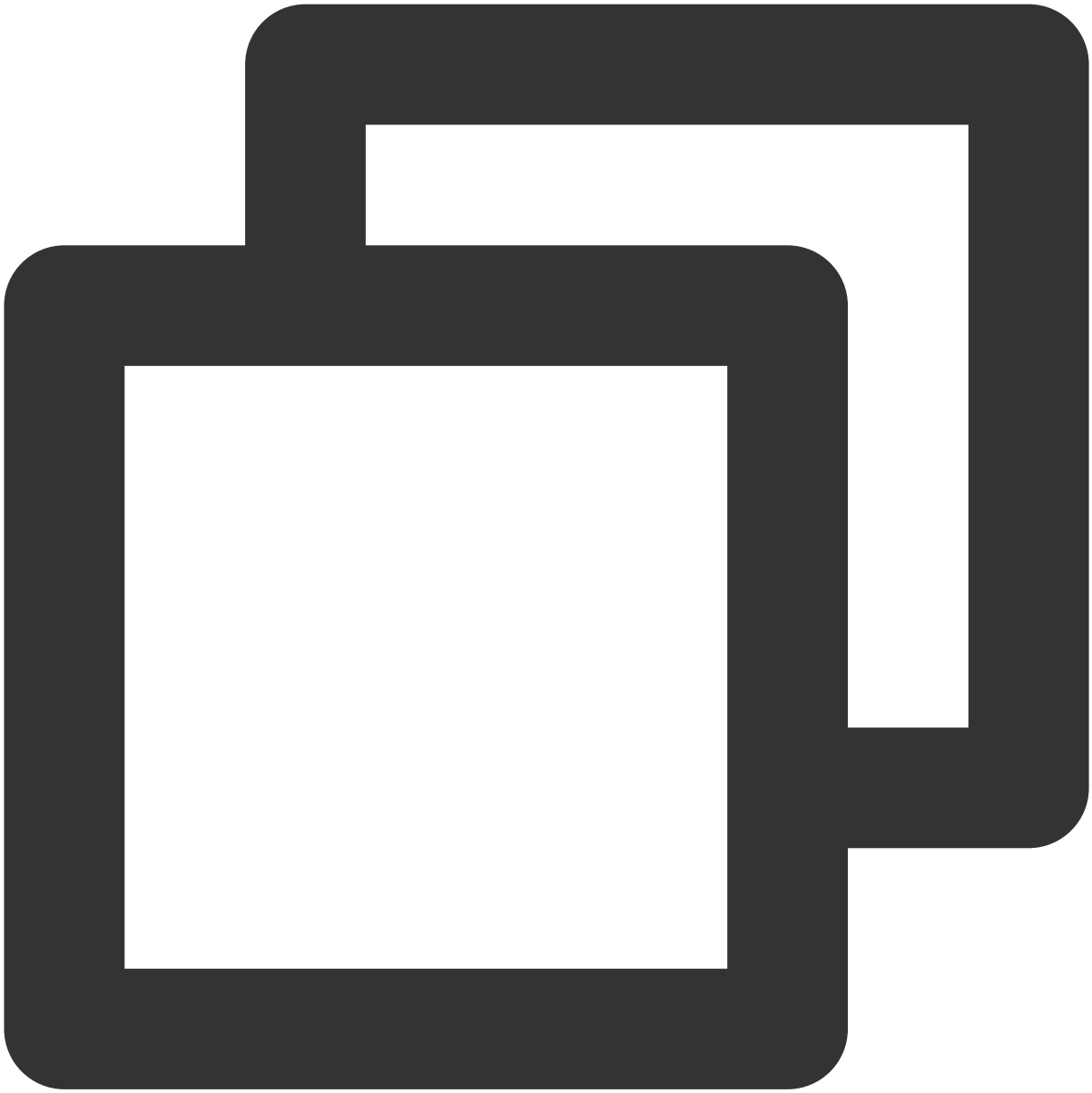
パラメータは下表に示すとおりです：

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。

©2013-2022 Tencent Cloud. All rights reserved.

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：

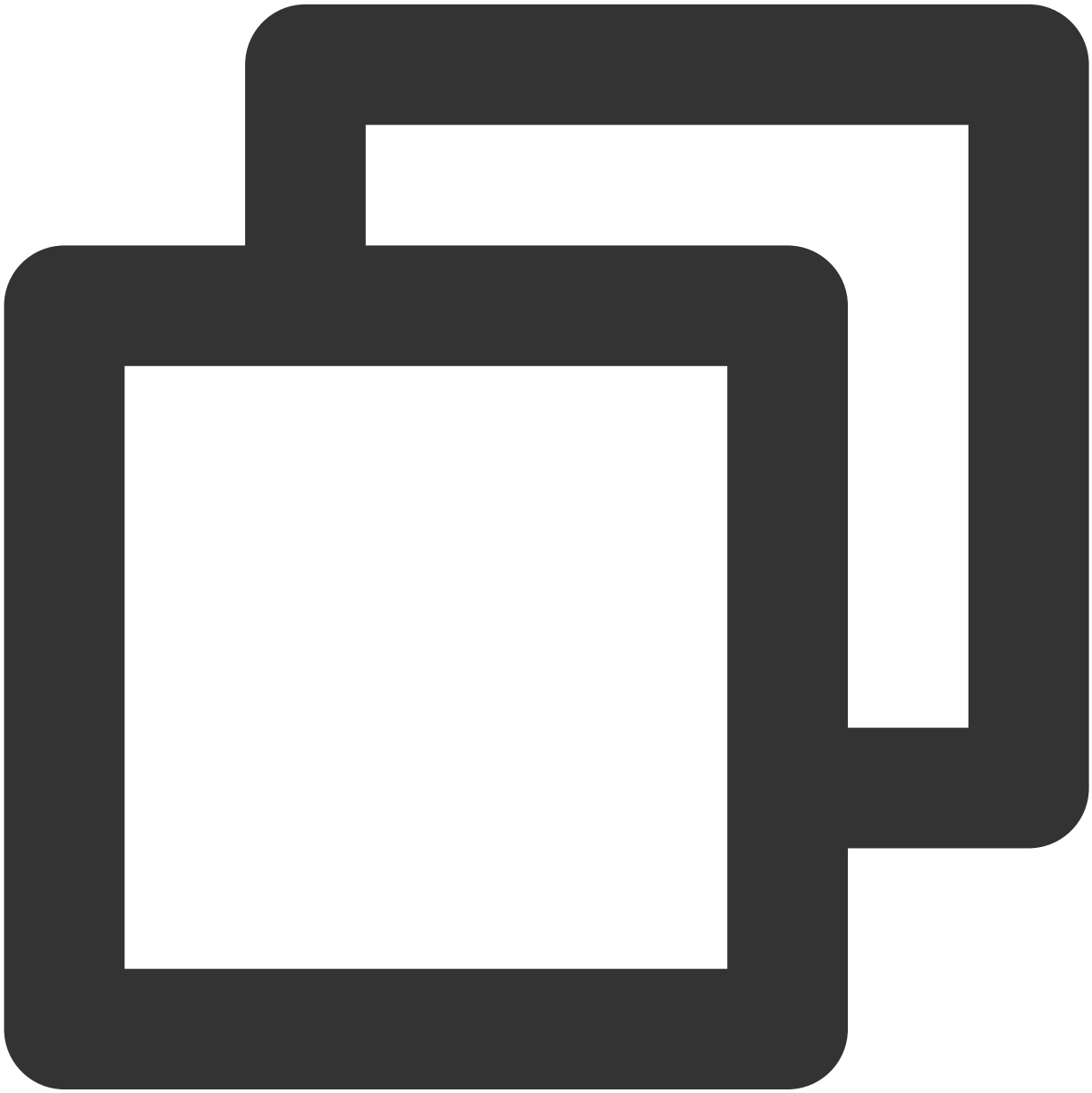
パラメータ	タイプ	意味
message	String	ログ情報。

©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);
```

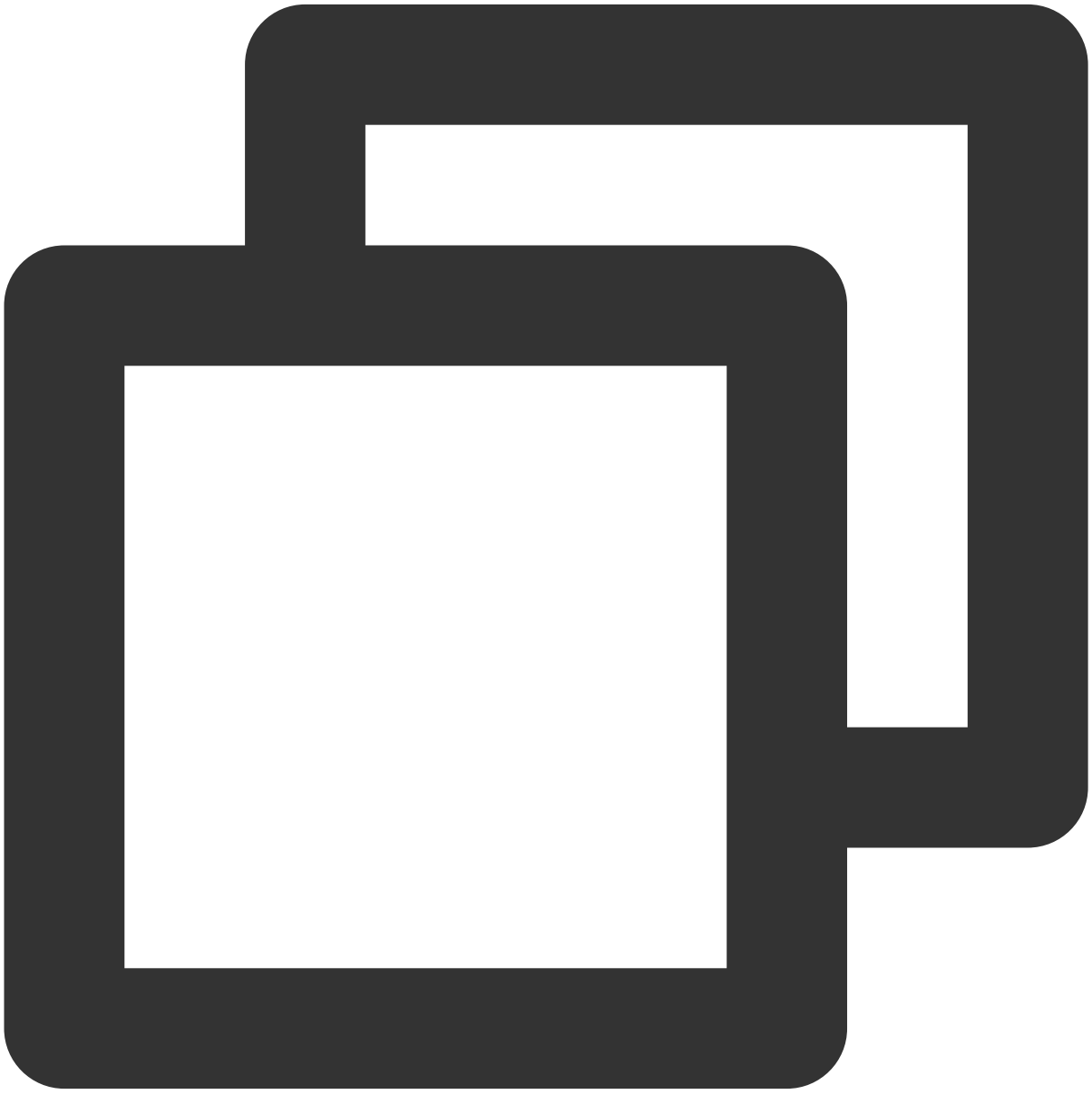
パラメータは下表に示すとおりです：

©2013-2022 Tencent Cloud. All rights reserved.

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);
```

©2013-2022 Tencent Cloud. All rights reserved.

パラメータは下表に示すとおりです：

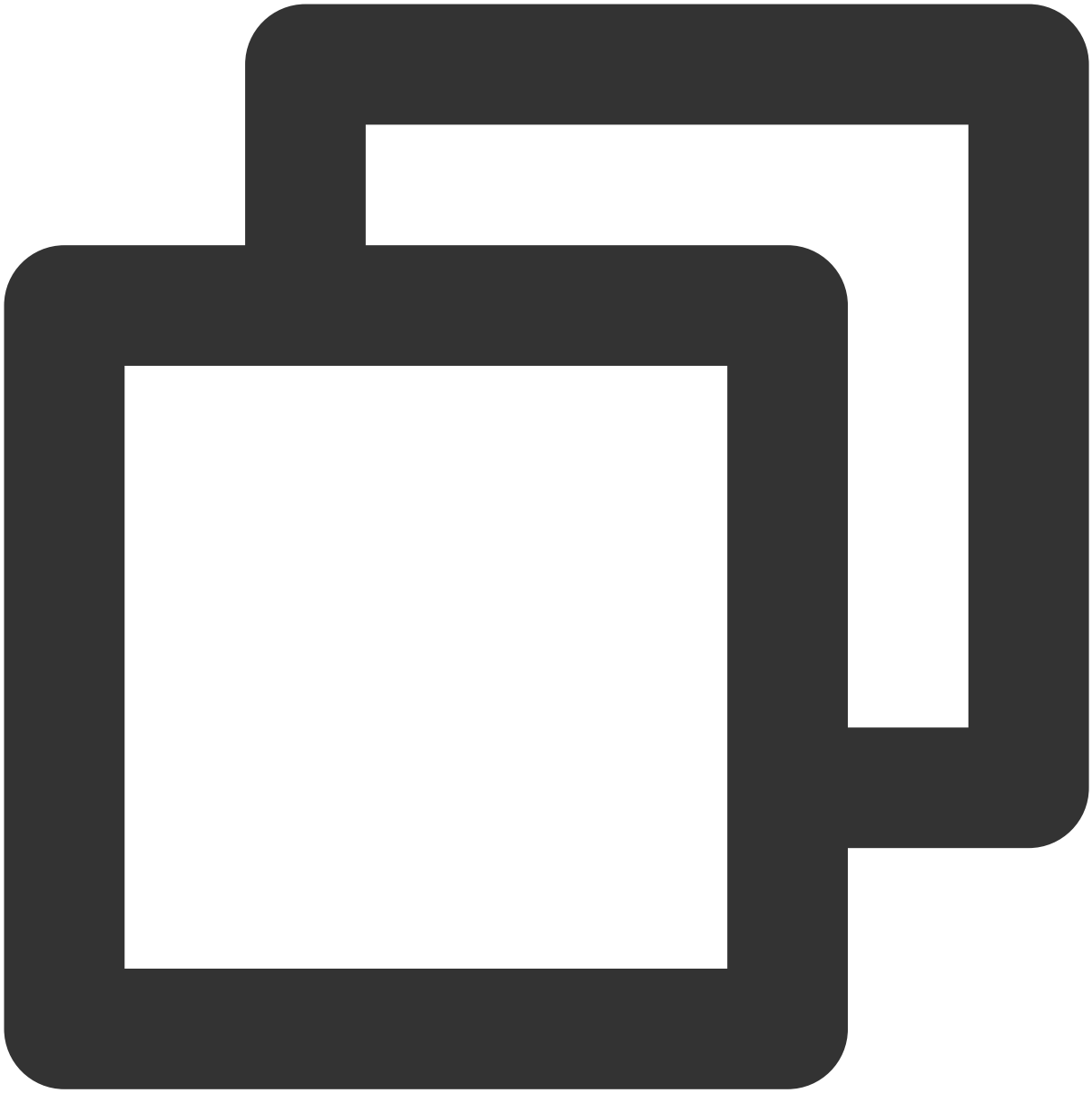
パラメータ	タイプ	意味
roomInfo	TRTCLiveRoomInfo	ルーム情報。

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。

©2013-2022 Tencent Cloud. All rights reserved.



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

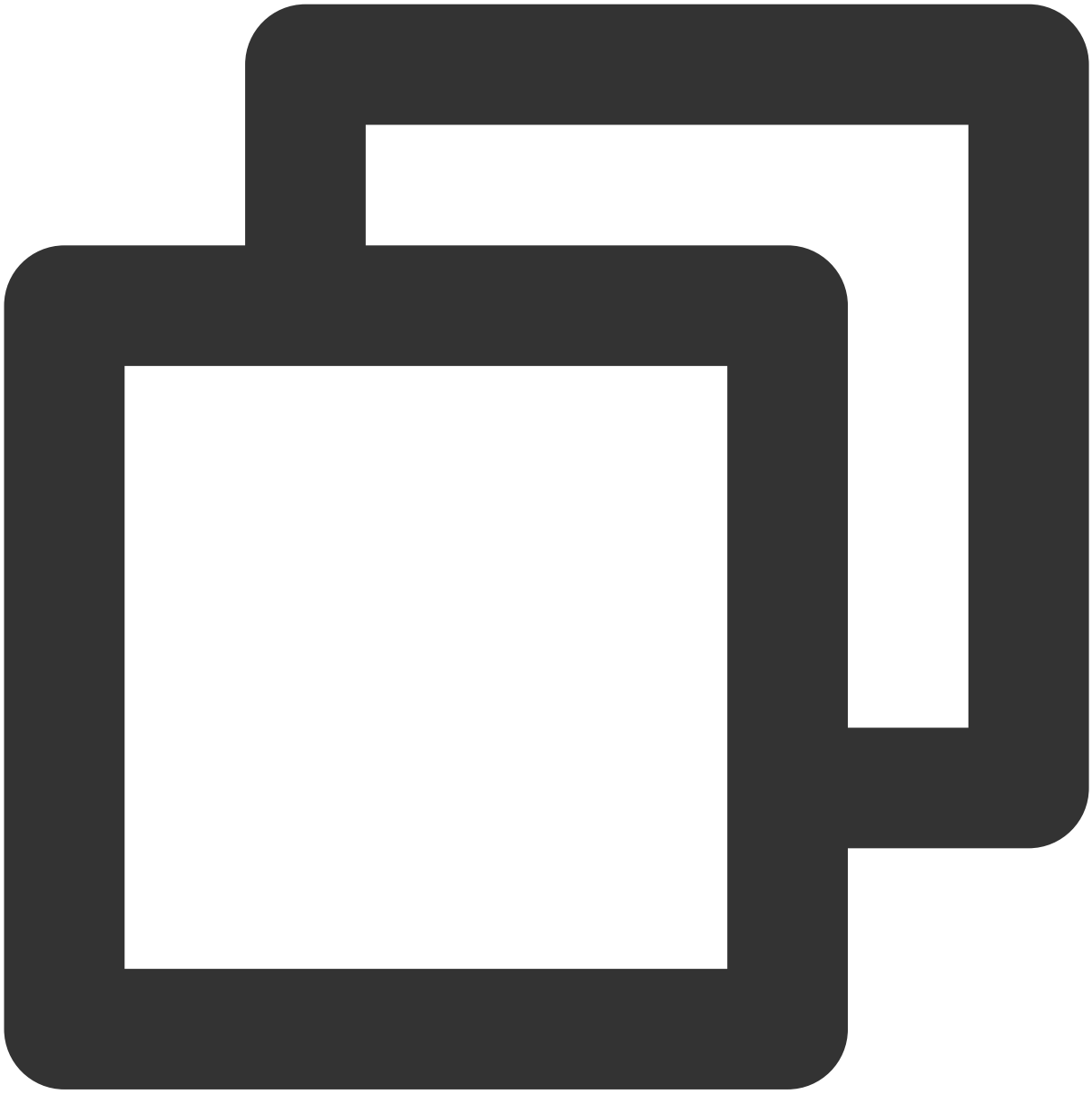
パラメータは下表に示すとおりです：

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。

onAnchorExit

©2013-2022 Tencent Cloud. All rights reserved.

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);
```

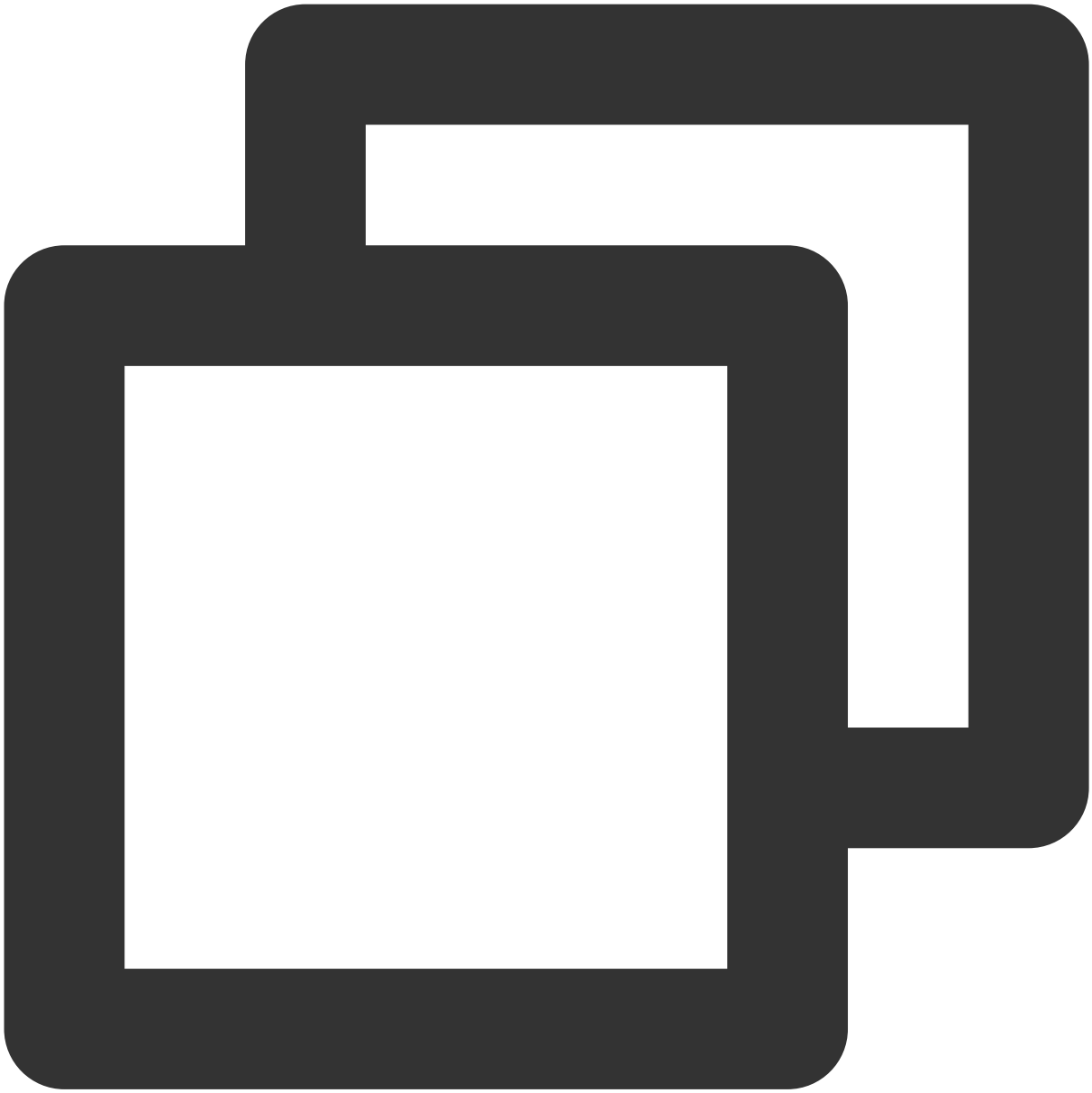
パラメータは下表に示すとおりです：

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。

©2013-2022 Tencent Cloud. All rights reserved.

onAudienceEnter

視聴者入室通知を受信します。



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

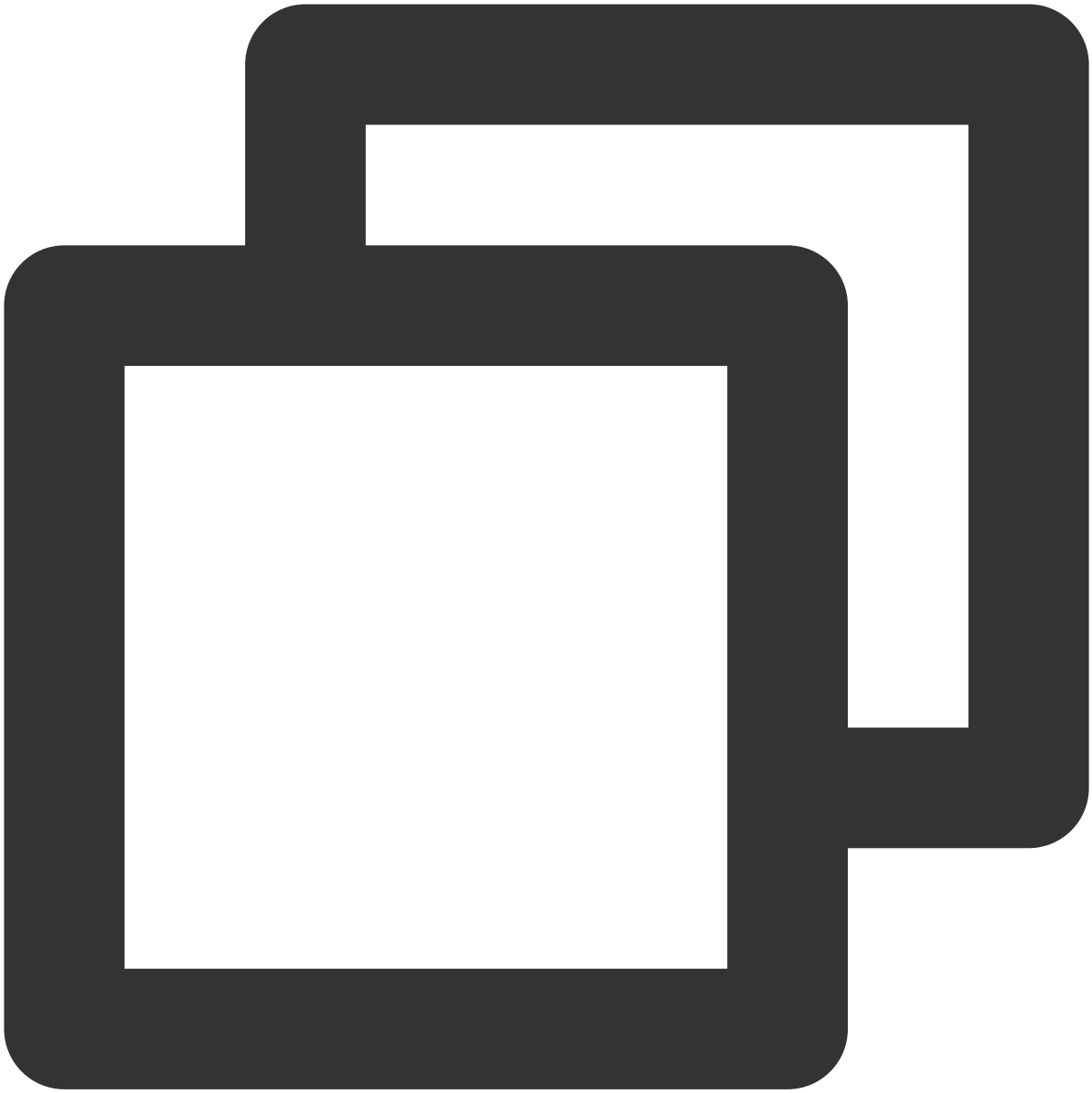
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userInfo	TRTCLiveUserInfo	入室した視聴者の情報。

©2013-2022 Tencent Cloud. All rights reserved.

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：

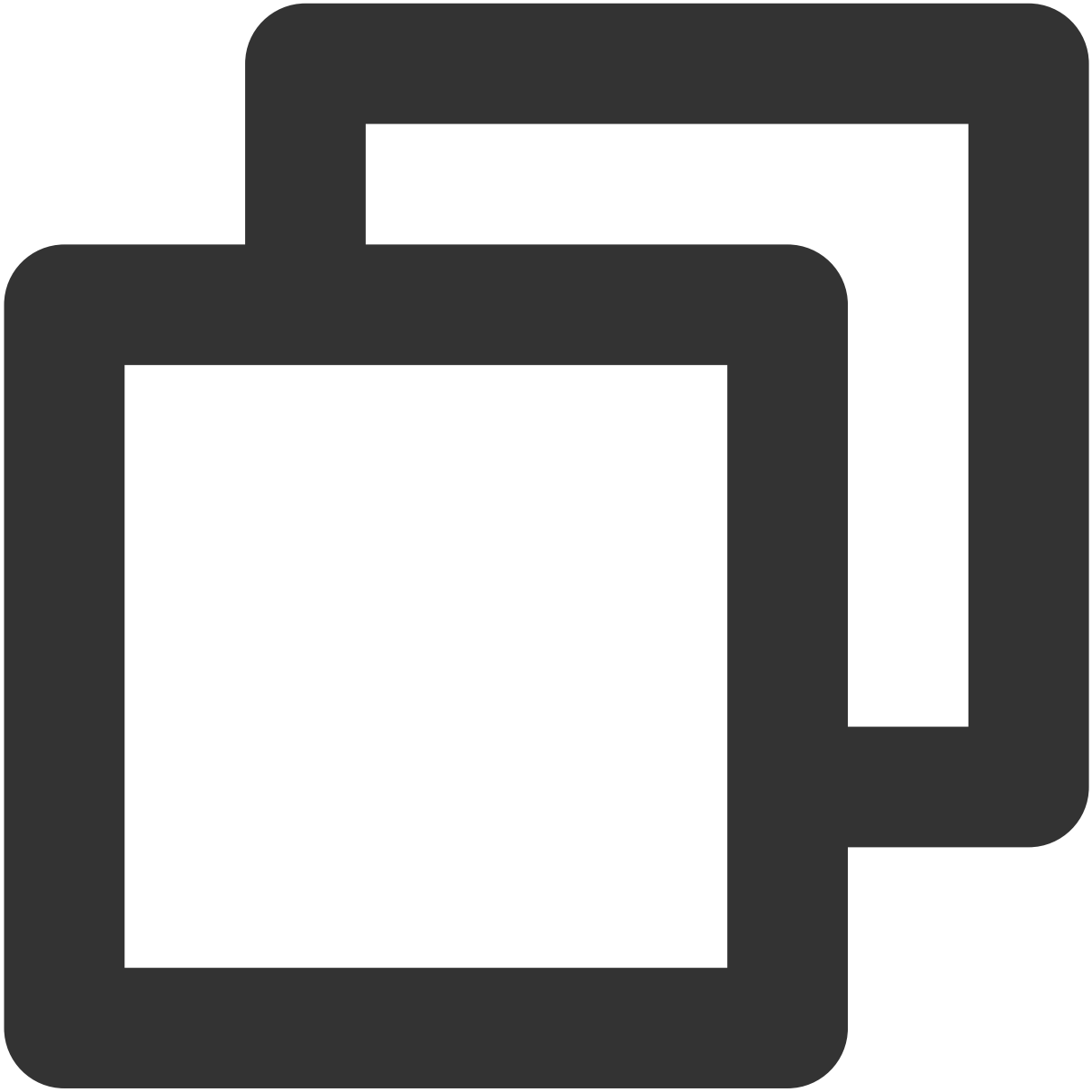
パラメータ	タイプ	意味
userInfo	TRTCLiveUserInfo	退室した視聴者の情報。

©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

onRequestJoinAnchor

キャスターが視聴者のマイク接続リクエストを受信した時のコールバック。



```
void onRequestJoinAnchor(TRTCLiveRoomDef.TRTCLiveUserInfo userInfo, String reason,
```

パラメータは下表に示すとおりです：

パラ	タイプ	意味

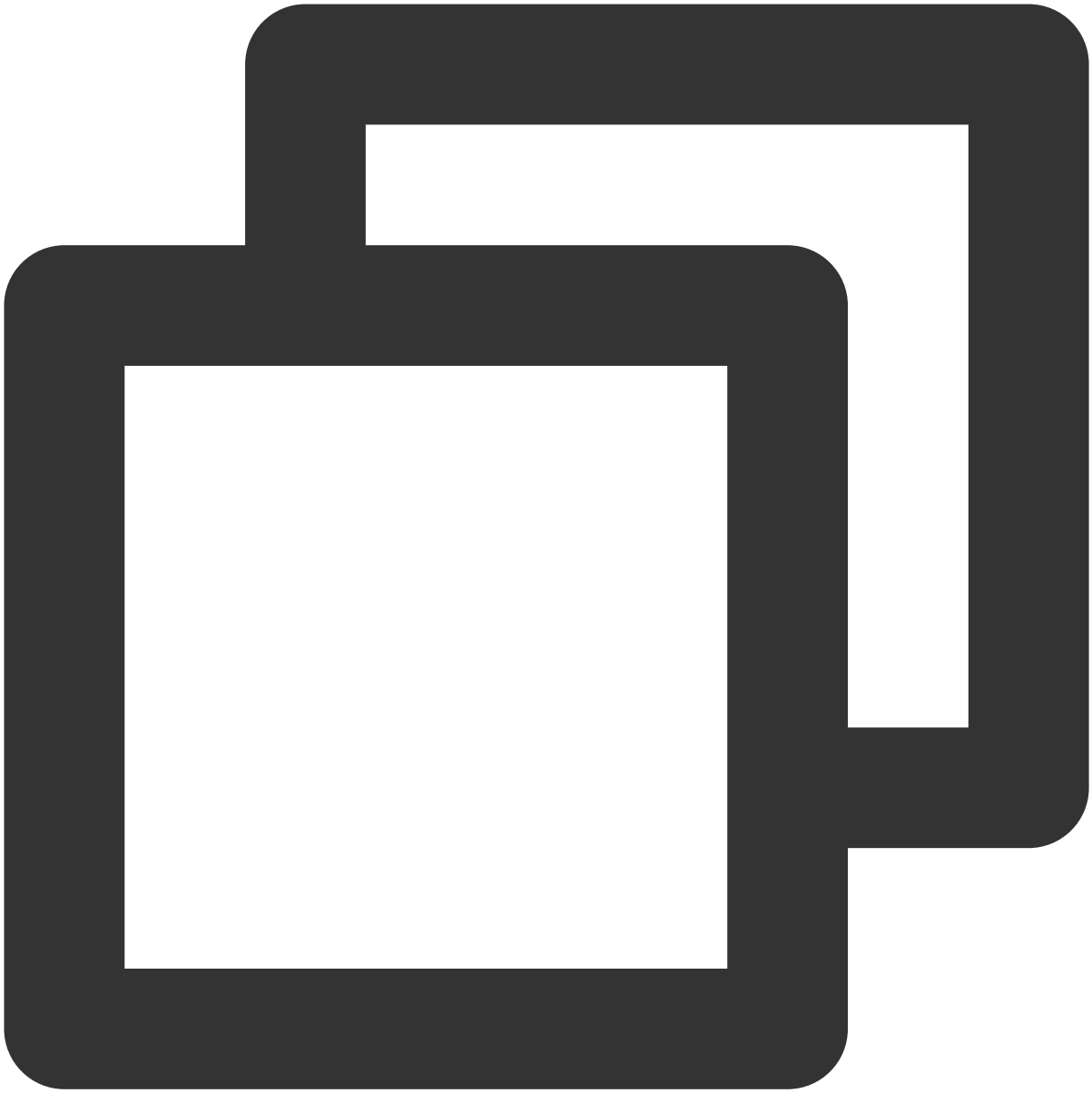
©2013-2022 Tencent Cloud. All rights reserved.

メータ		
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。

©2013-2022 Tencent Cloud. All rights reserved.



```
void onKickoutJoinAnchor();
```

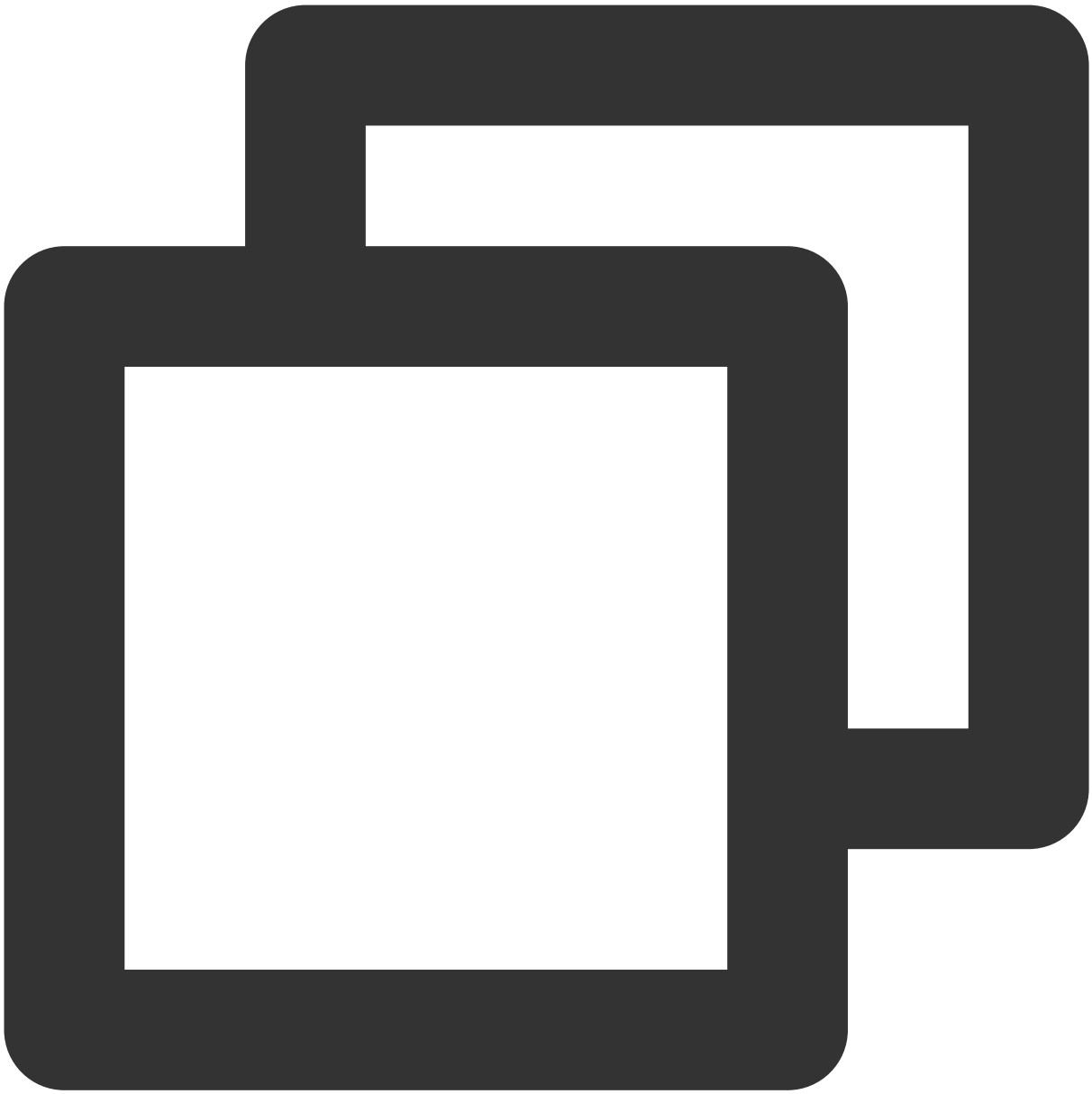
キャスター間PKのイベントコールバック

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

©2013-2022 Tencent Cloud. All rights reserved.

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);
```

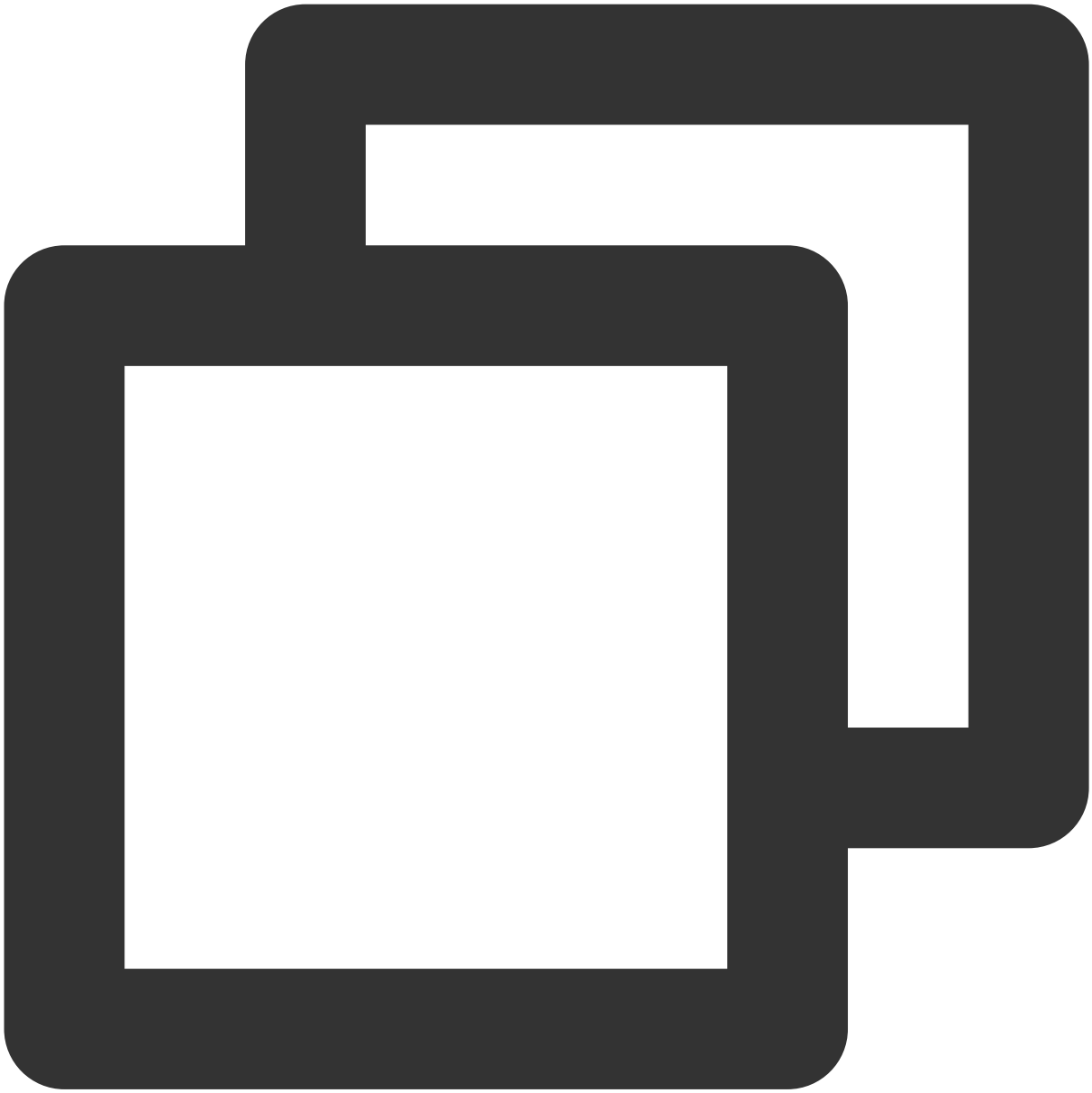
パラメータは下表に示すとおりです：

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。



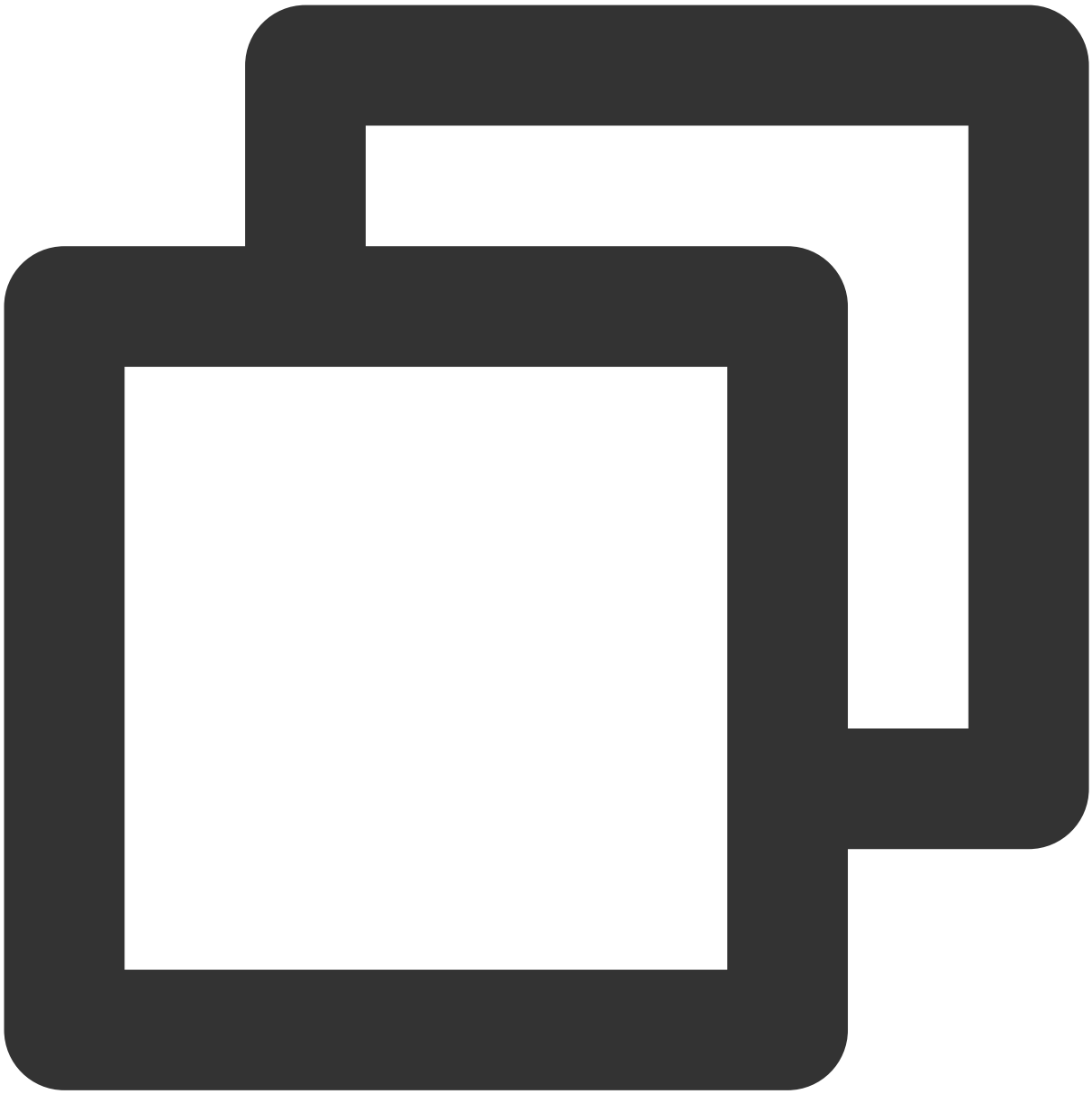
```
void onQuitRoomPK();
```

メッセージイベントのコールバック

onRecvRoomTextMsg

©2013-2022 Tencent Cloud. All rights reserved.

テキストメッセージを受信します。



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

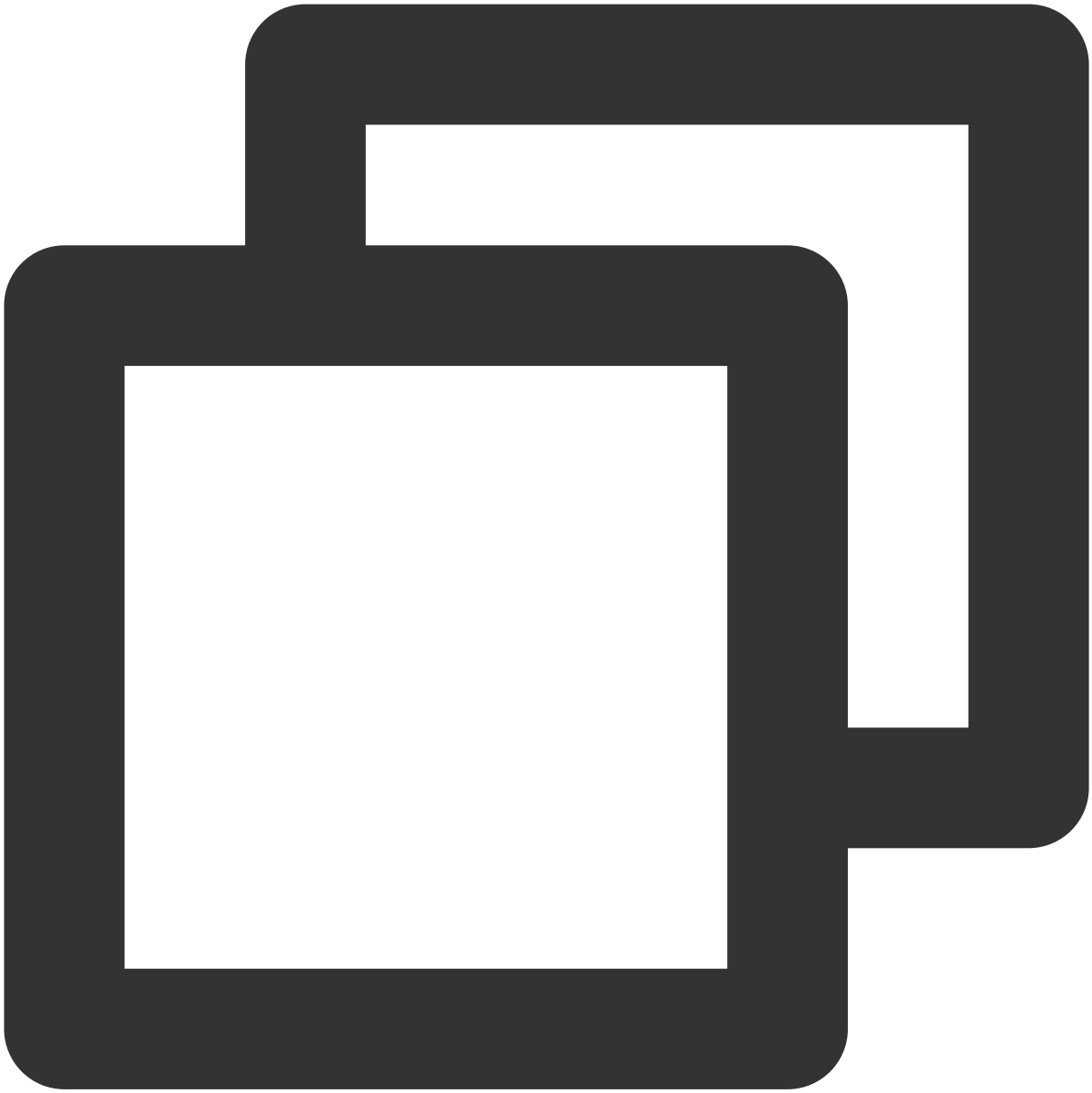
パラメータは下表に示すとおりです：

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。

©2013-2022 Tencent Cloud. All rights reserved.

onRecvRoomCustomMsg

カスタムメッセージを受信します。



```
void onRecvRoomCustomMsg(String cmd, String message, TRTCLiveRoomDef.TRTCLiveUserIn
```

パラ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

©2013-2022 Tencent Cloud. All rights reserved.

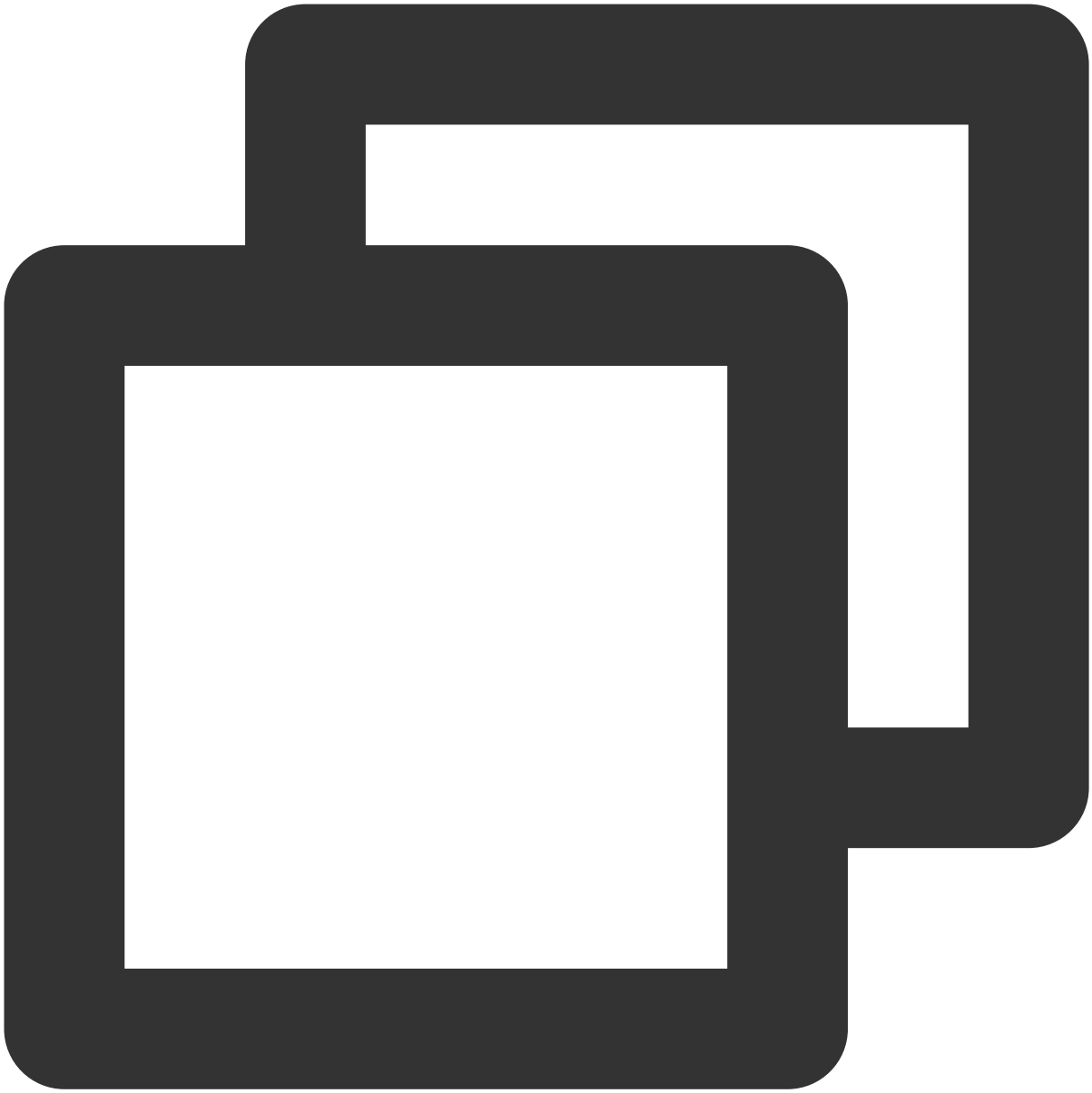
		ざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
userInfo	TRTCLiveUserInfo	送信者のユーザー情報。

TRTCAudioEffectManager

playBGM

BGMを再生します。

©2013-2022 Tencent Cloud. All rights reserved.



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

パラメータは下表に示すとおりです：

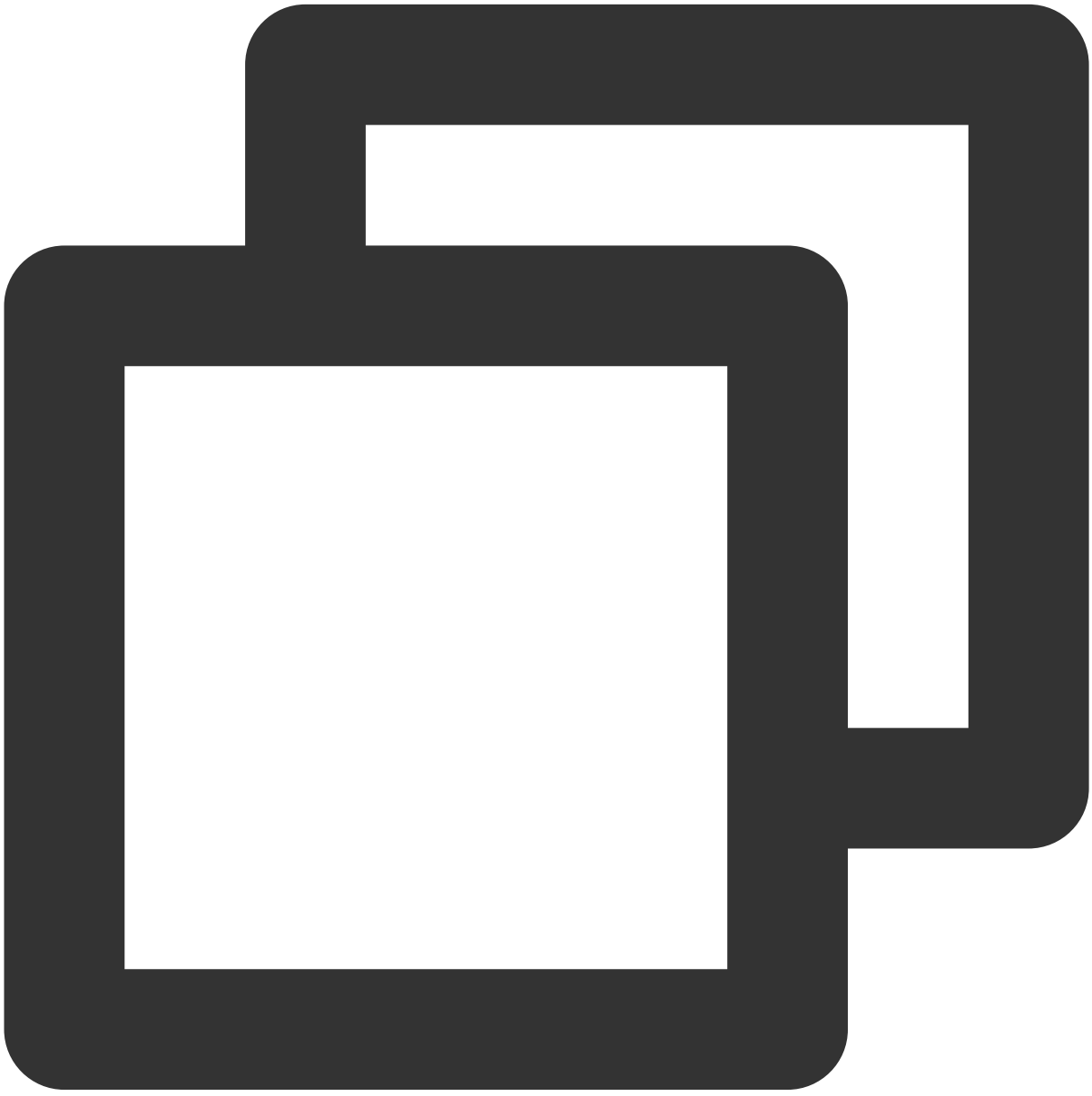
パラメータ	タイプ	意味
url	String	BGMファイルパス。
loopTimes	int	ループ回数
bgmVol	int	BGM音量

©2013-2022 Tencent Cloud. All rights reserved.

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。

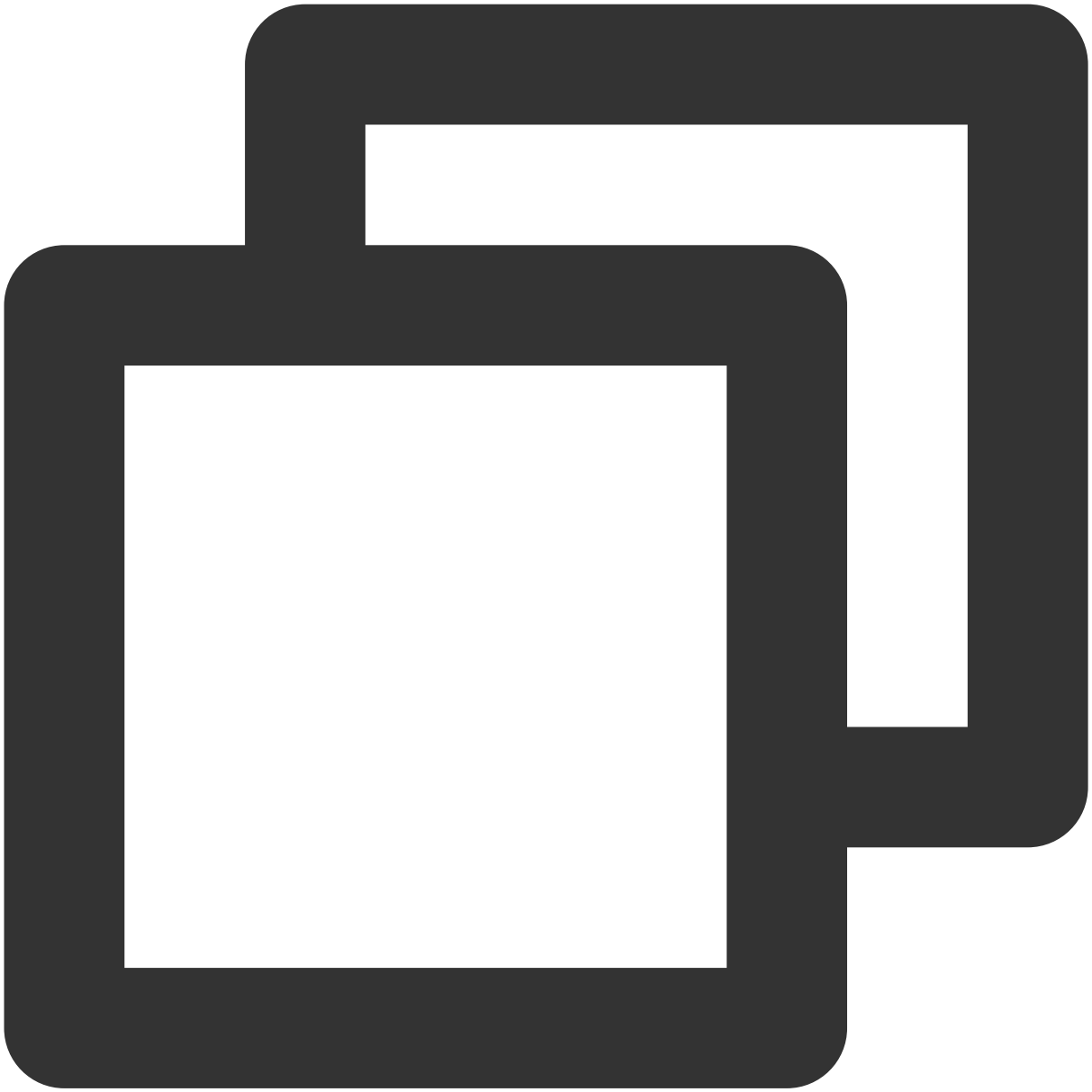


```
void stopBGM();
```

©2013-2022 Tencent Cloud. All rights reserved.

pauseBGM

BGMの再生を一時停止します。

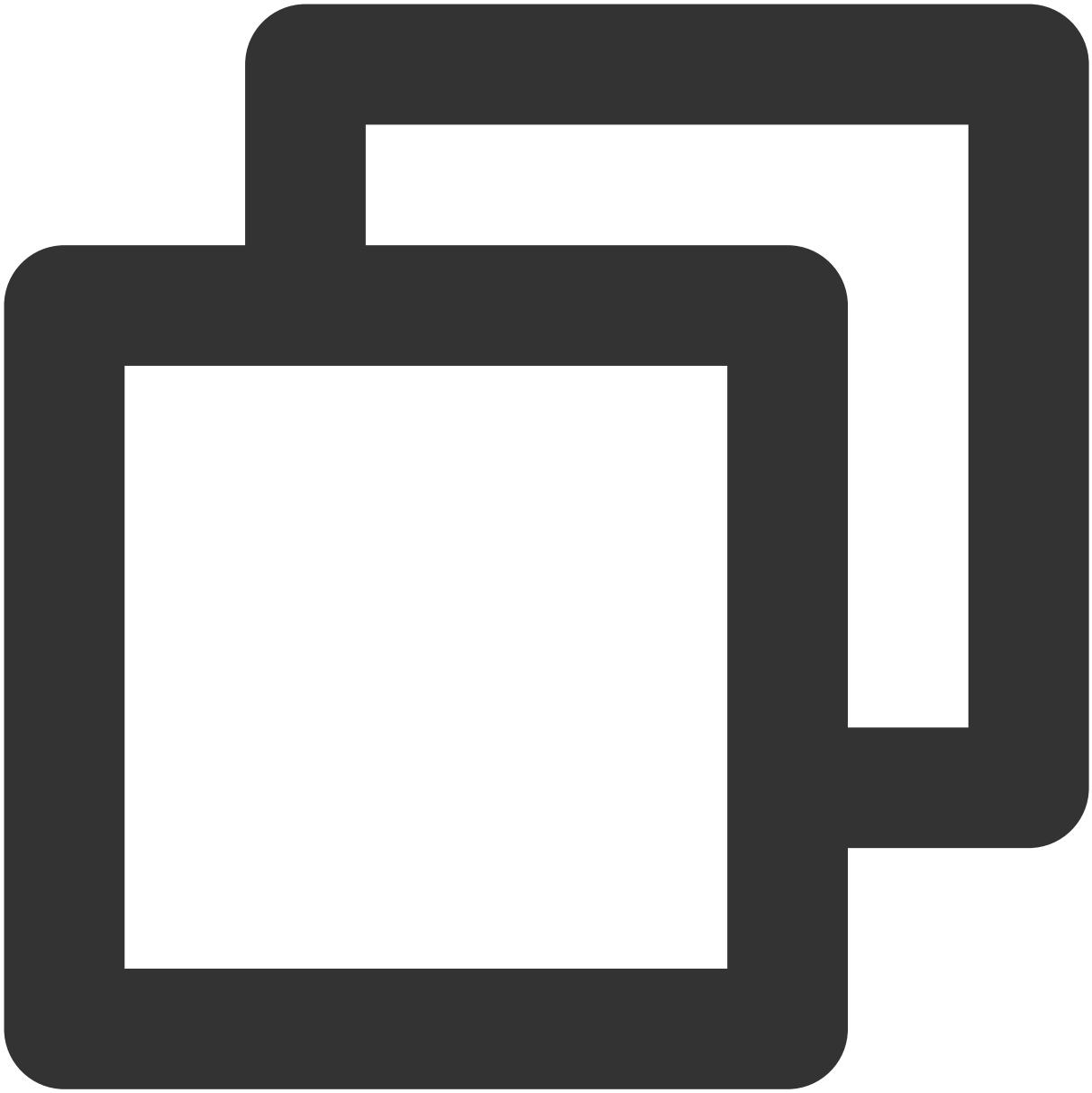


```
void pauseBGM();
```

resumeBGM

BGMの再生を続けます

©2013-2022 Tencent Cloud. All rights reserved.

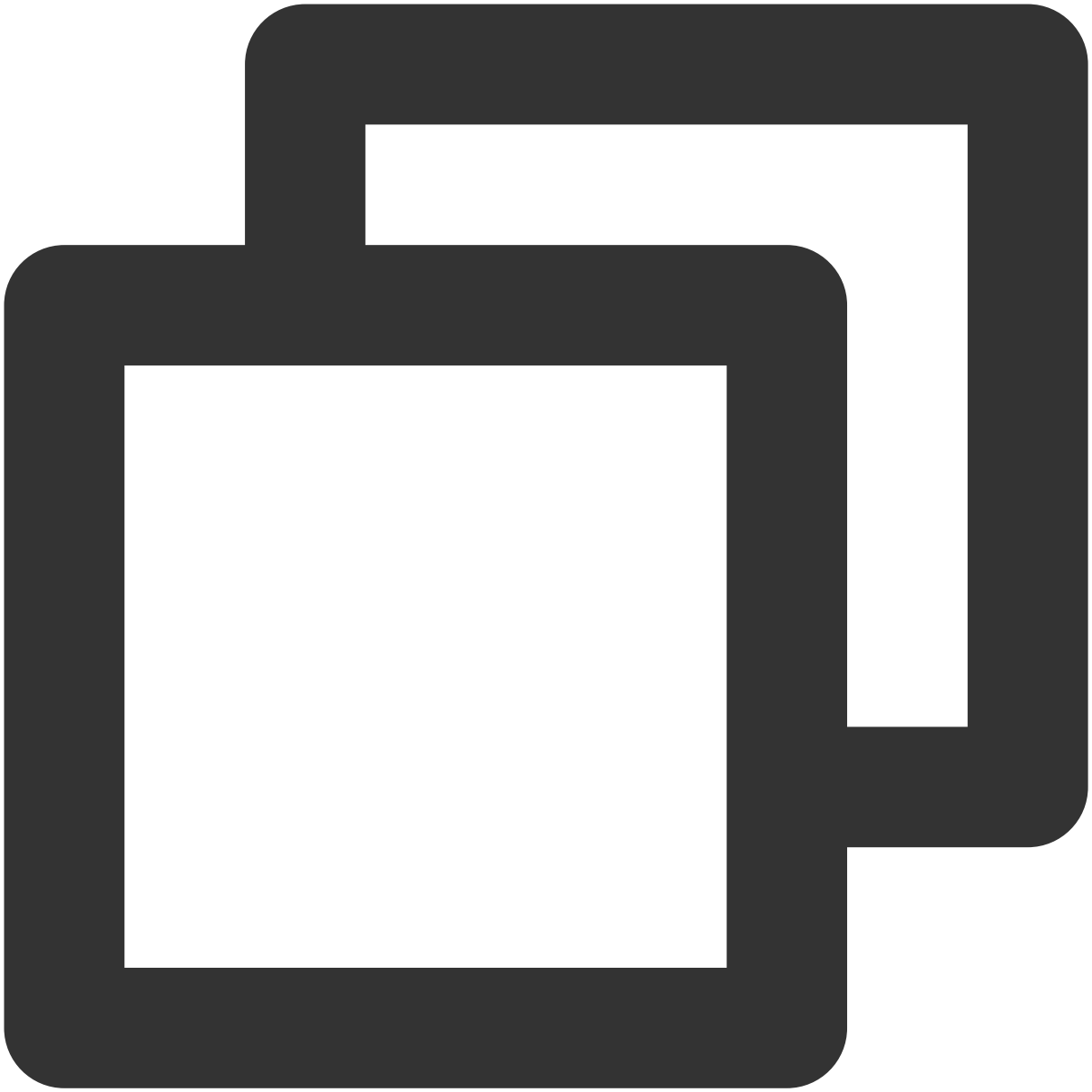


```
void resumeBGM();
```

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);
```

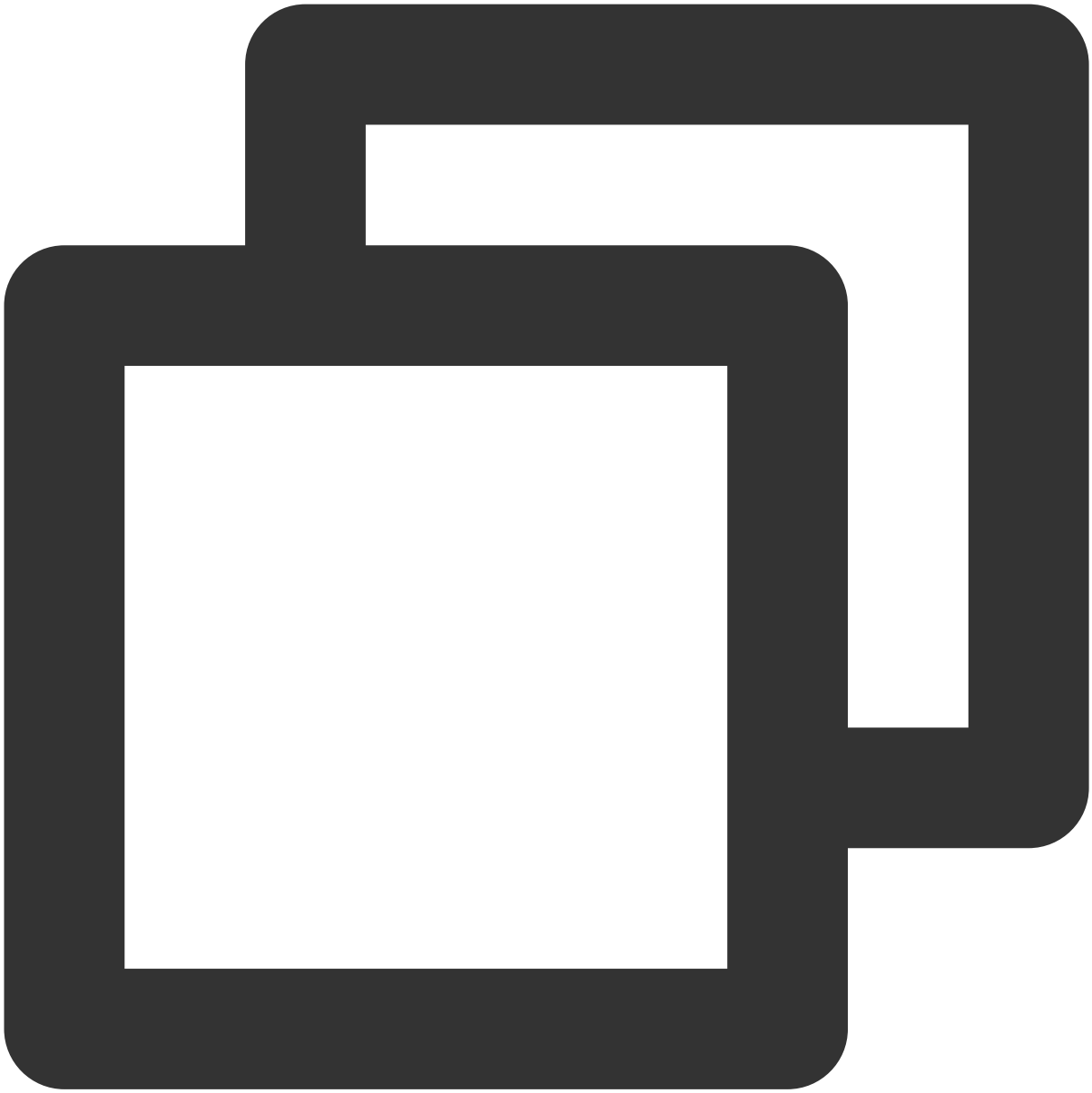
パラメータは下表に示すとおりです：

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。

©2013-2022 Tencent Cloud. All rights reserved.

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）。

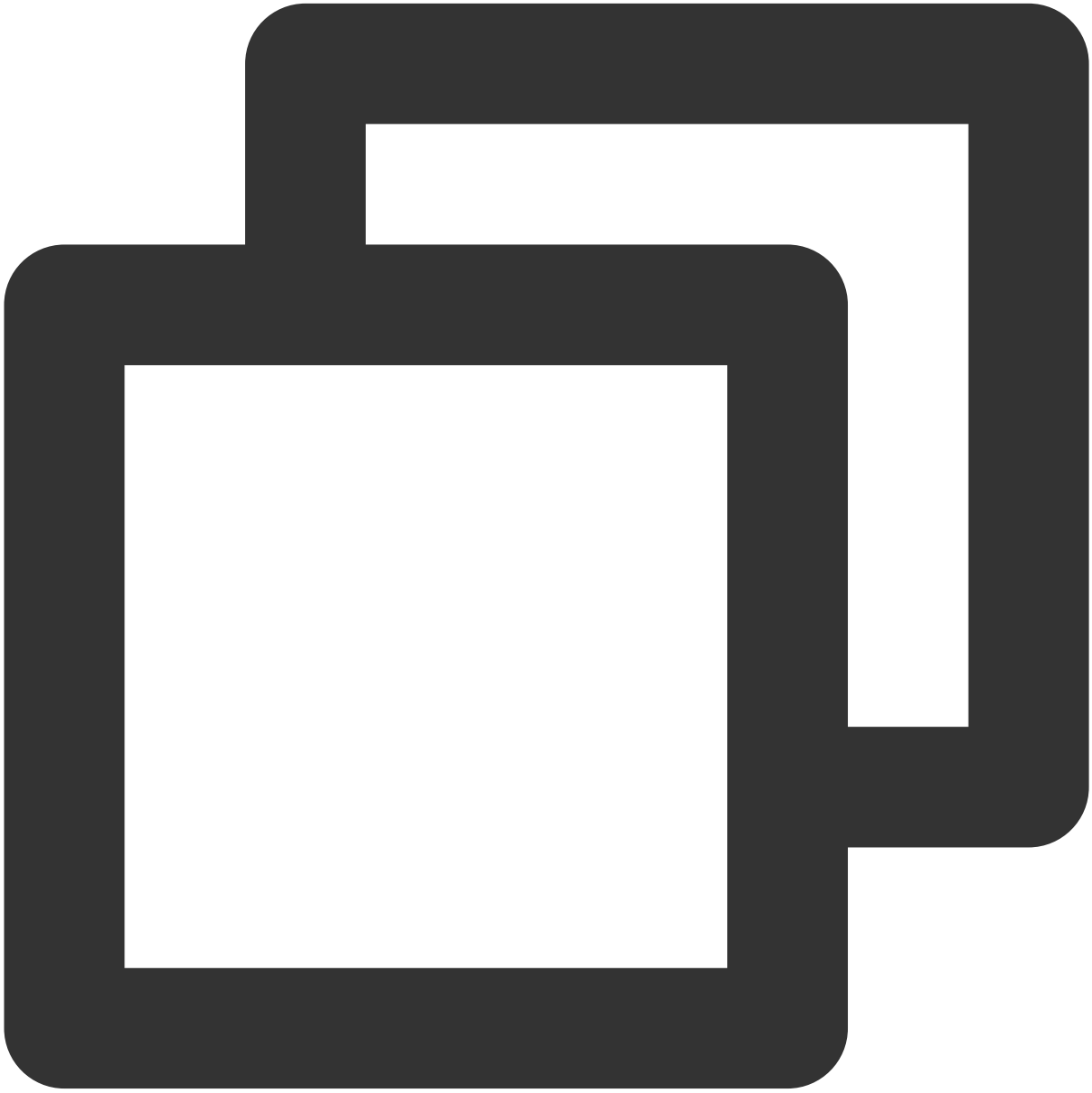
©2013-2022 Tencent Cloud. All rights reserved.

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。



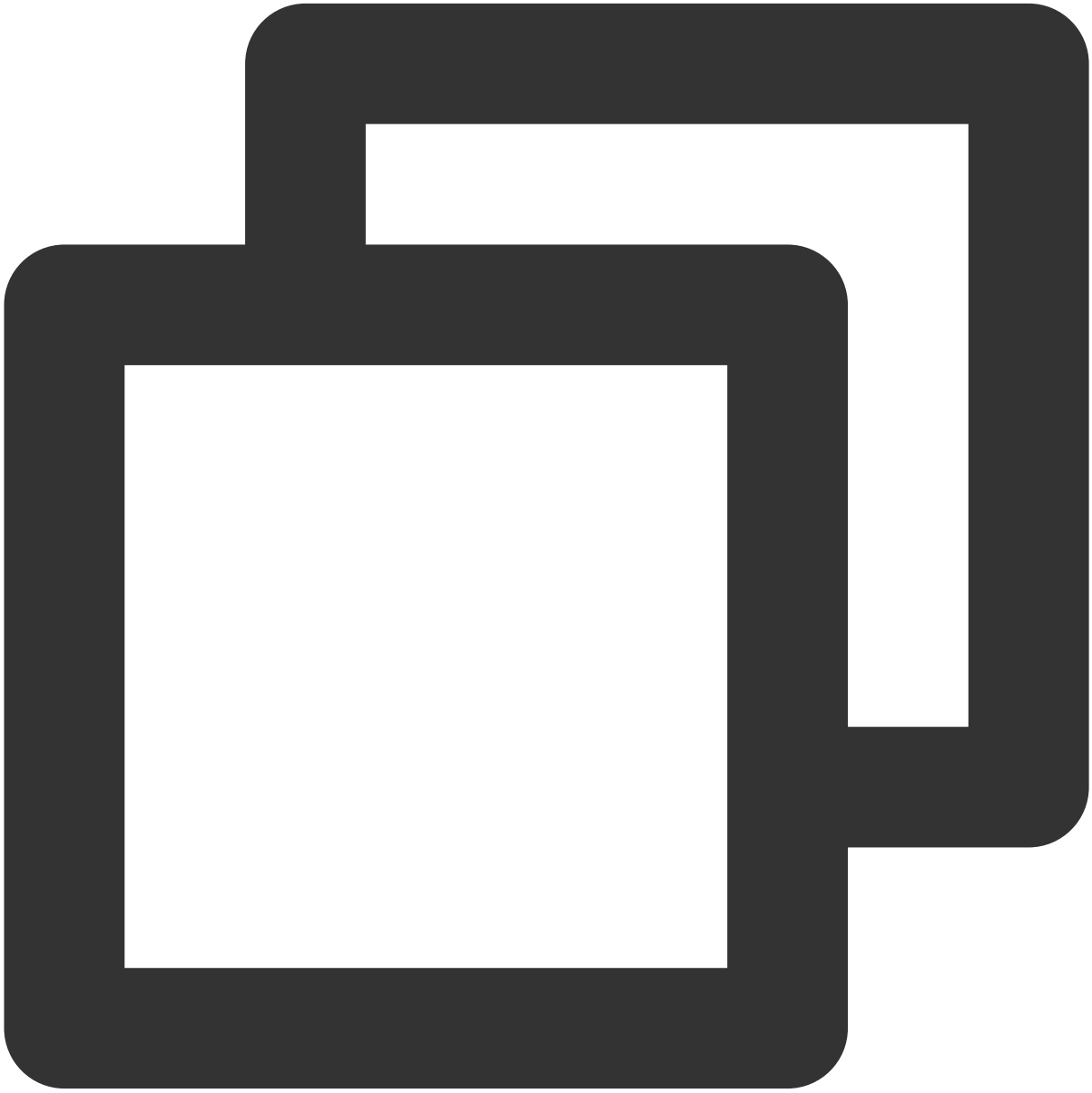
```
void setMicVolume(int volume);
```

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。



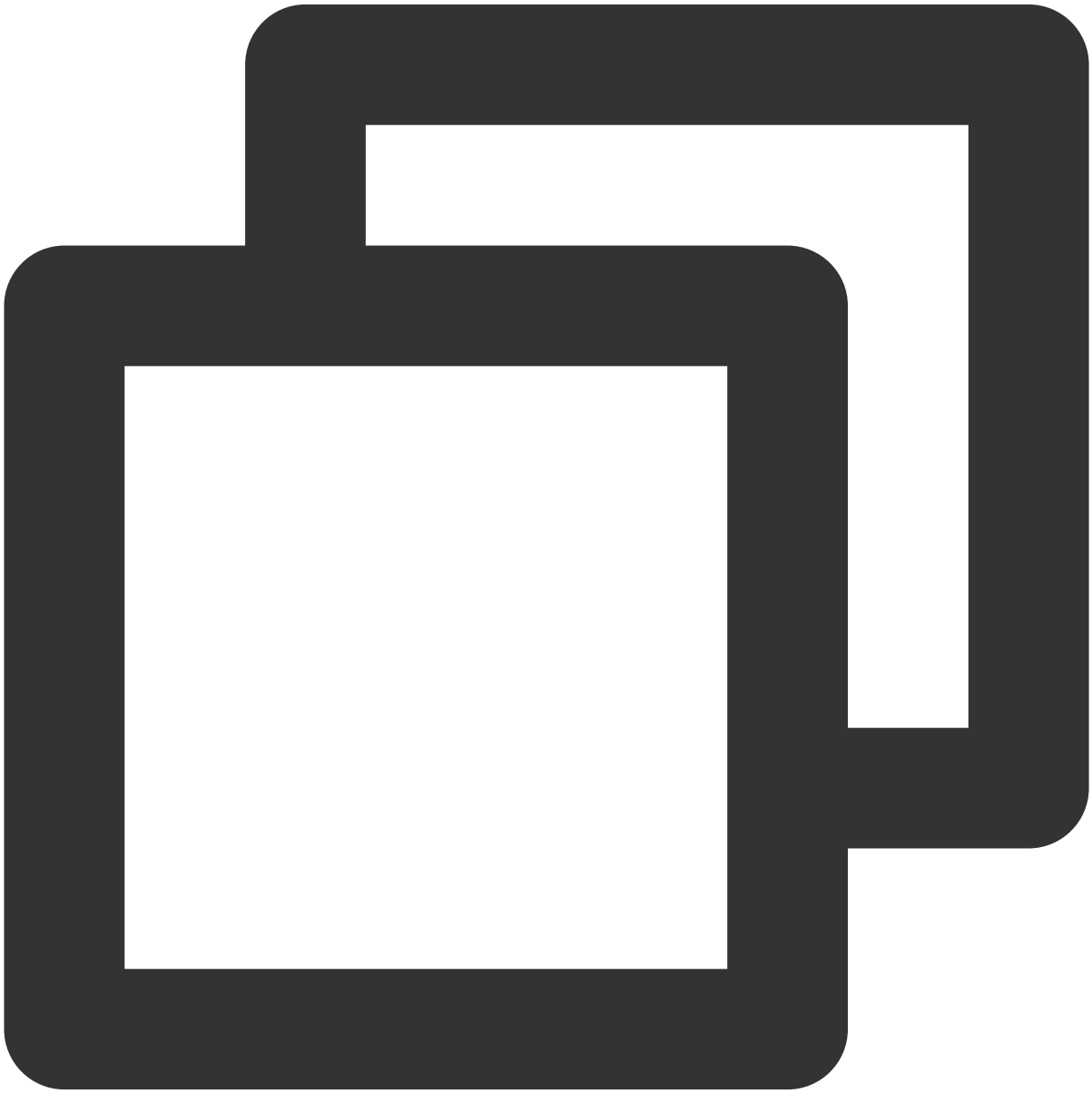
```
void setReverbType(int reverbType);
```

パラ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);
```

©2013-2022 Tencent Cloud. All rights reserved.

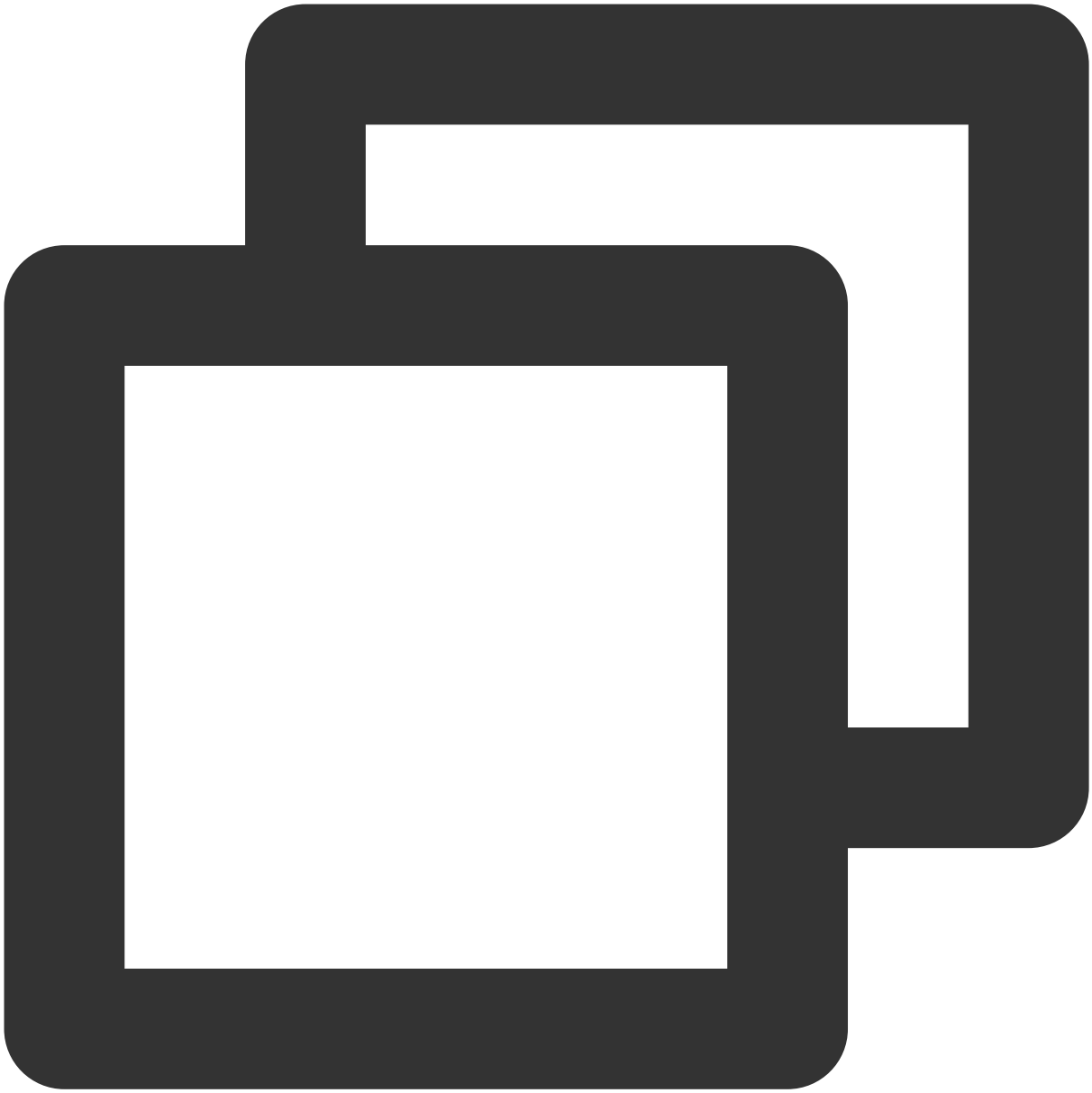
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
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。

©2013-2022 Tencent Cloud. All rights reserved.



```
void playAudioEffect(int effectId, String path, int count, boolean publish, int vol
```

パラメータは下表に示すとおりです：

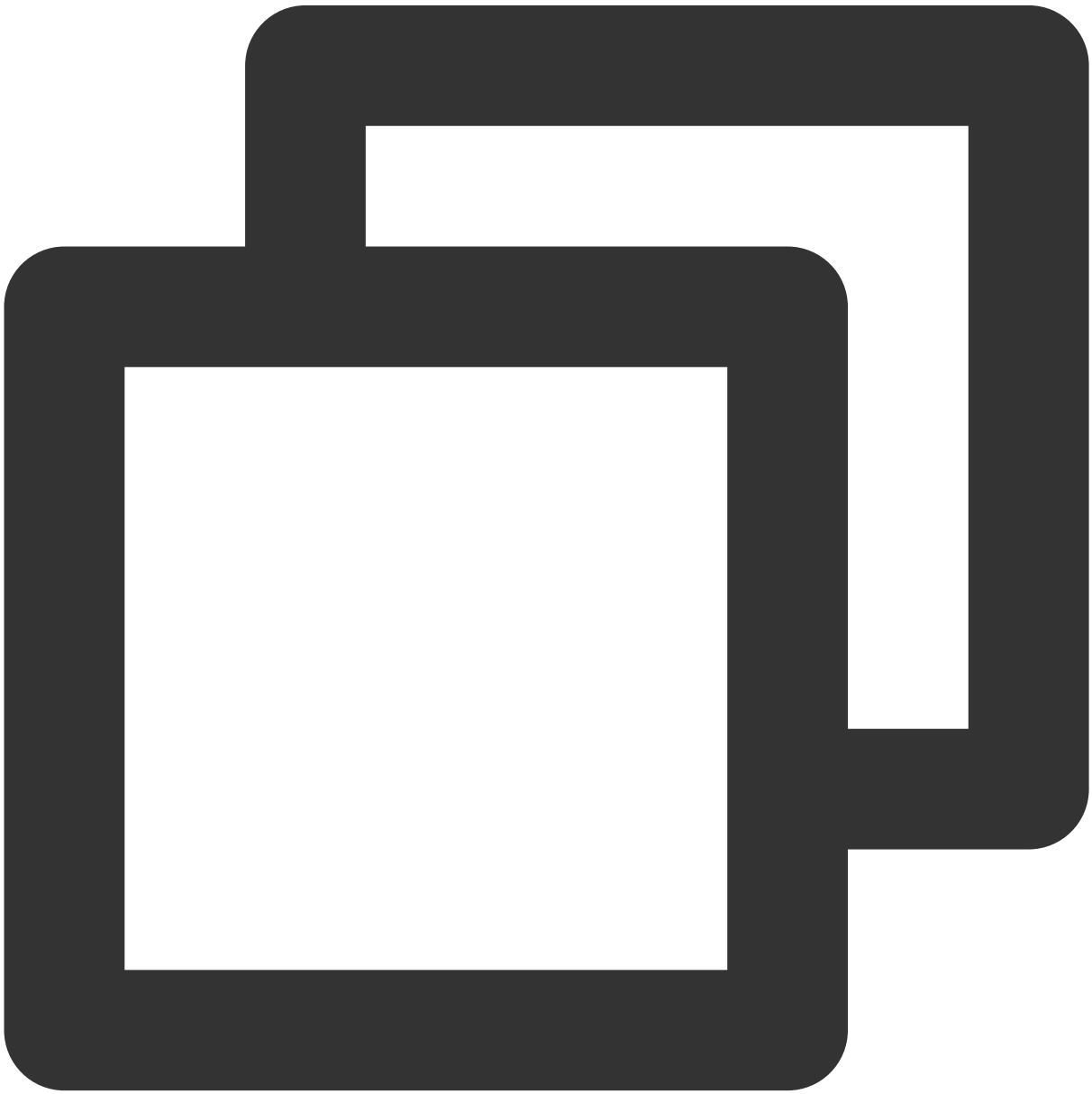
パラメータ	タイプ	意味
effectId	int	サウンドエフェクト ID。
path	String	サウンドエフェクトパス。

©2013-2022 Tencent Cloud. All rights reserved.

count	int	ループ回数。
publish	boolean	プッシュかどうか / trueは視聴者に対するプッシュ、 falseはローカルプレビューです。
volume	int	音量の大きさ。数値の範囲は0 - 100、デフォルト値は100です。

pauseAudioEffect

サウンドエフェクトの再生を一時停止します。



©2013-2022 Tencent Cloud. All rights reserved.

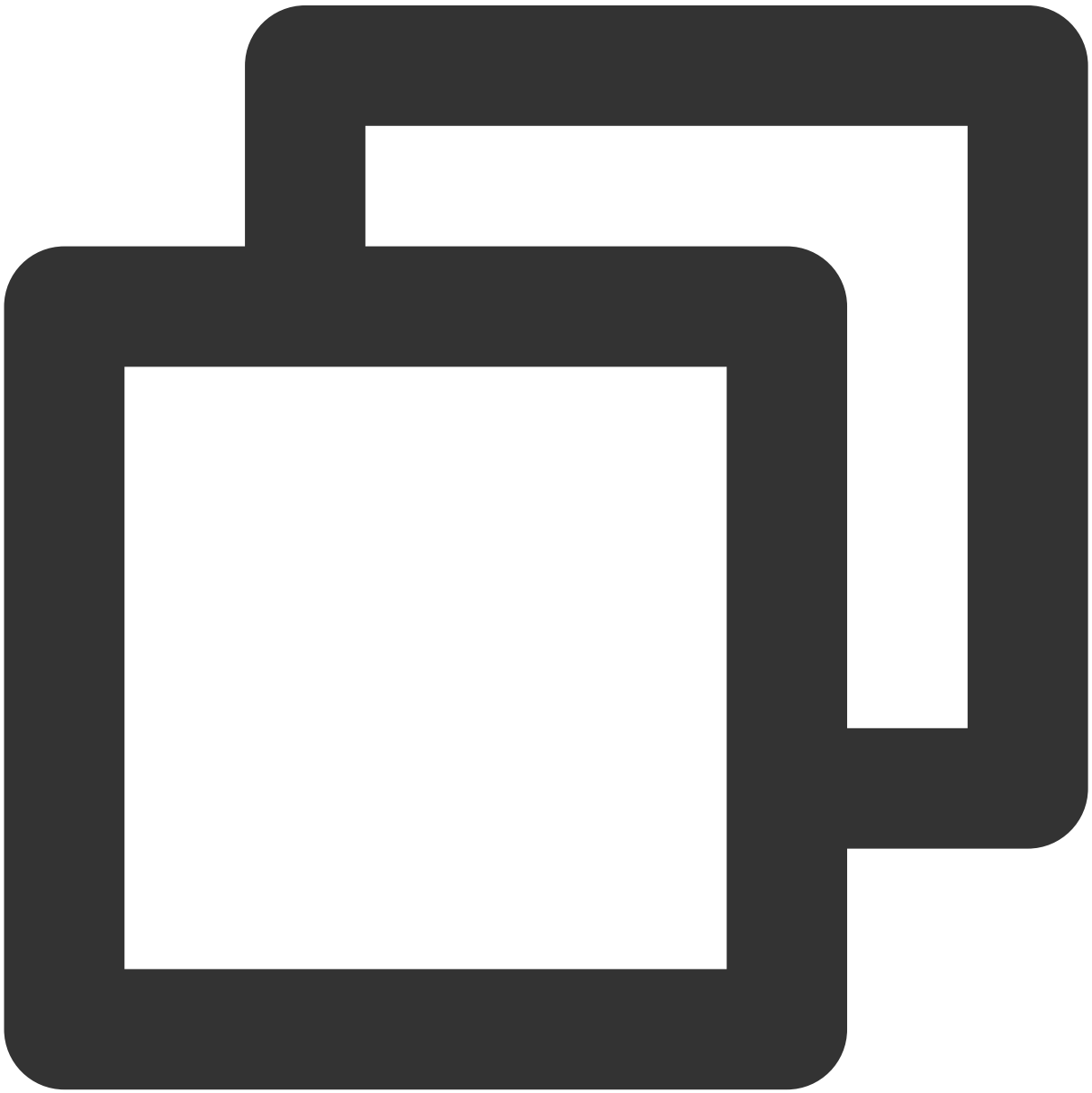
```
void pauseAudioEffect(int effectId);
```

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。



©2013-2022 Tencent Cloud. All rights reserved.

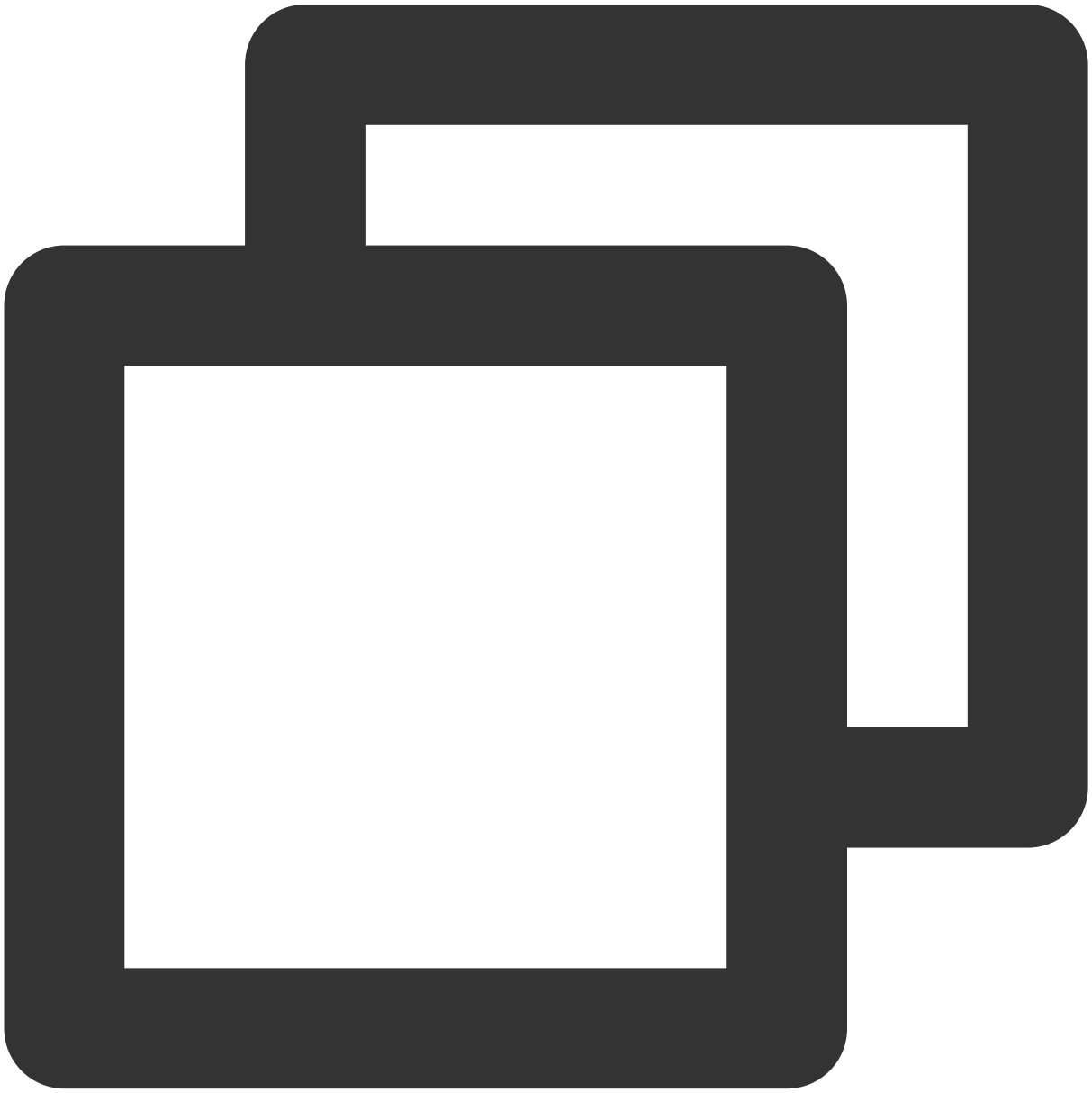
```
void resumeAudioEffect(int effectId);
```

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。



©2013-2022 Tencent Cloud. All rights reserved.

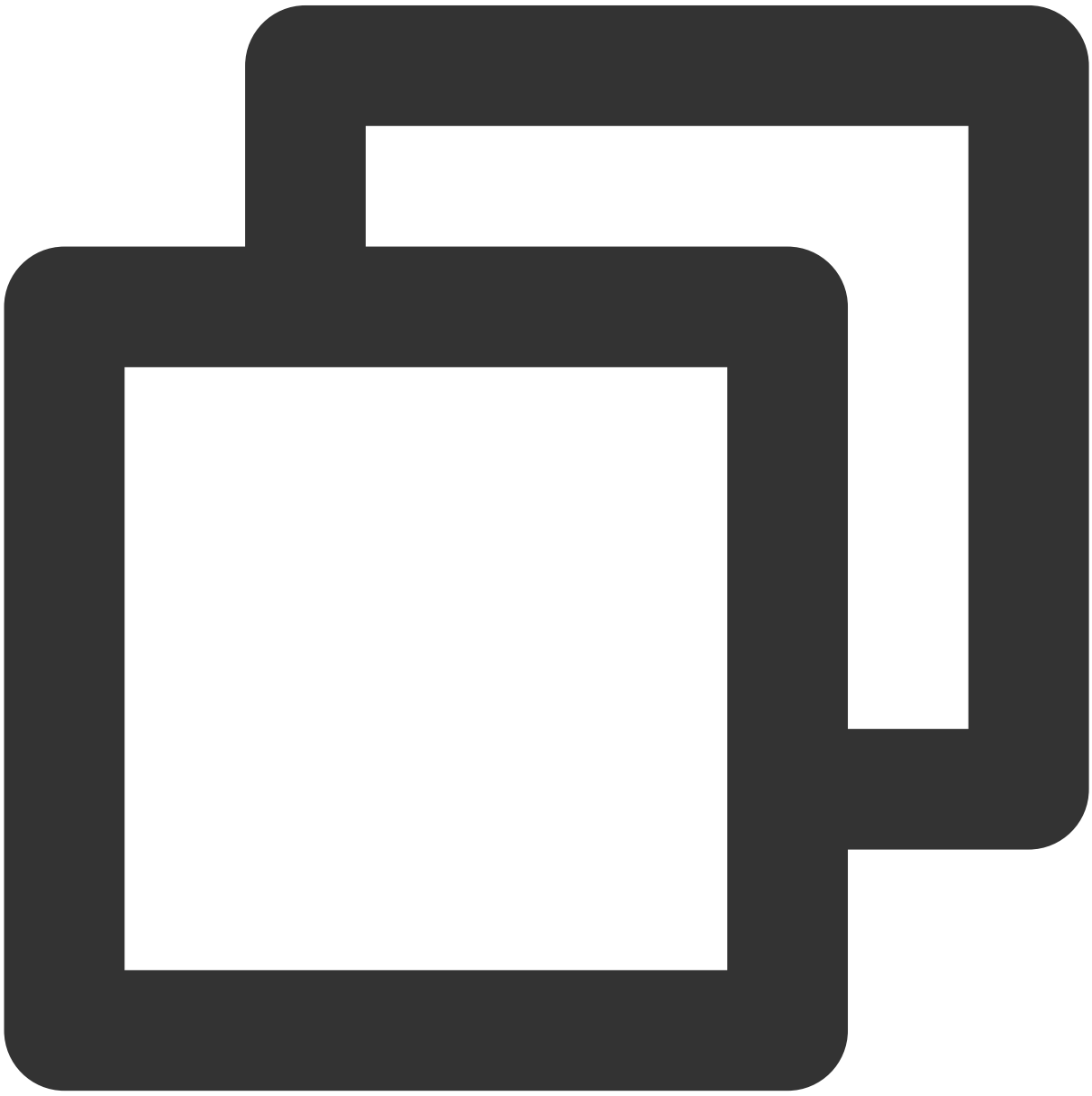
```
void stopAudioEffect(int effectId);
```

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。

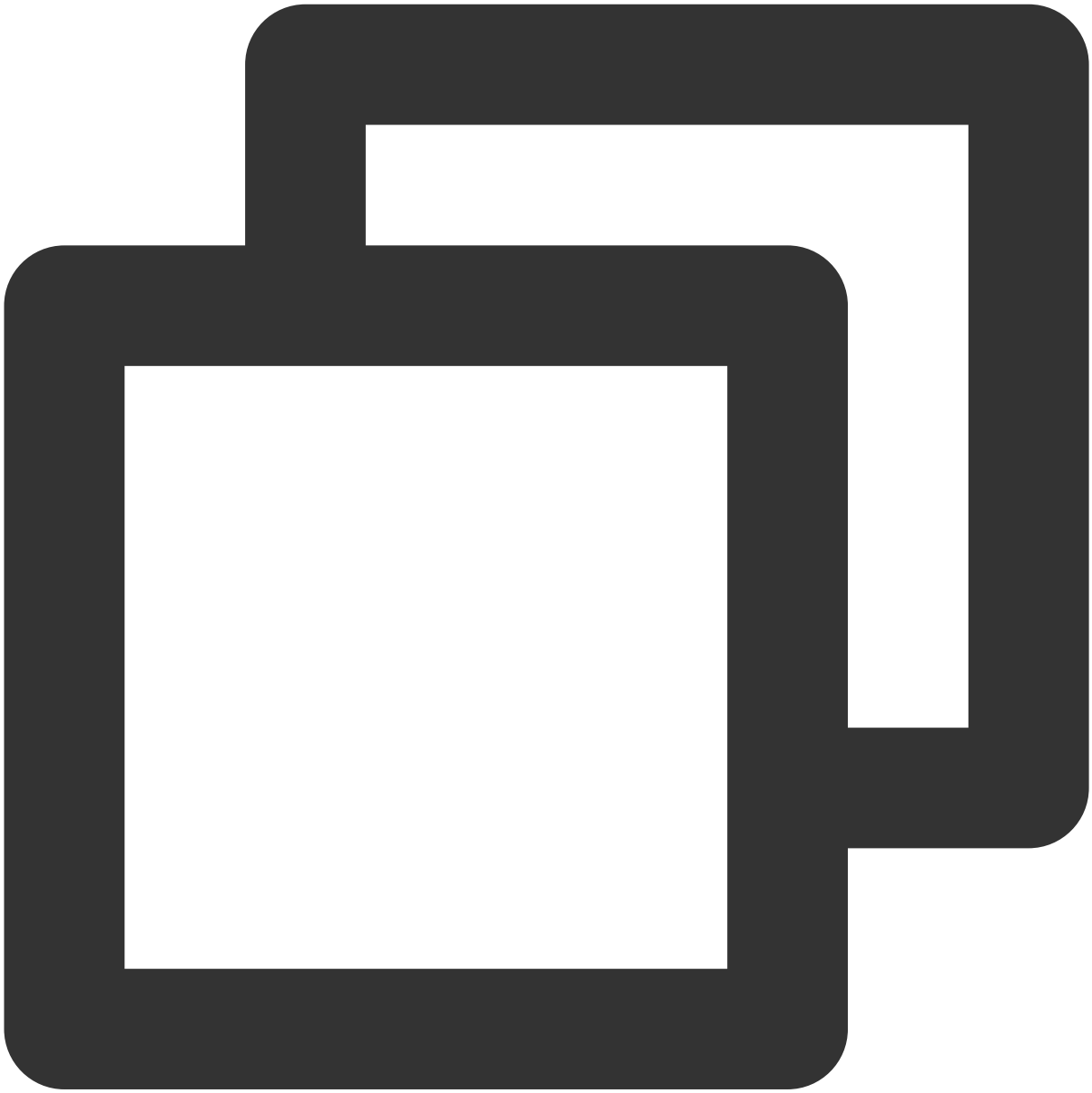


©2013-2022 Tencent Cloud. All rights reserved.

```
void stopAllAudioEffects();
```

setAudioEffectVolume

サウンドエフェクトの音量を設定します。



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

パラメータは下表に示すとおりです：

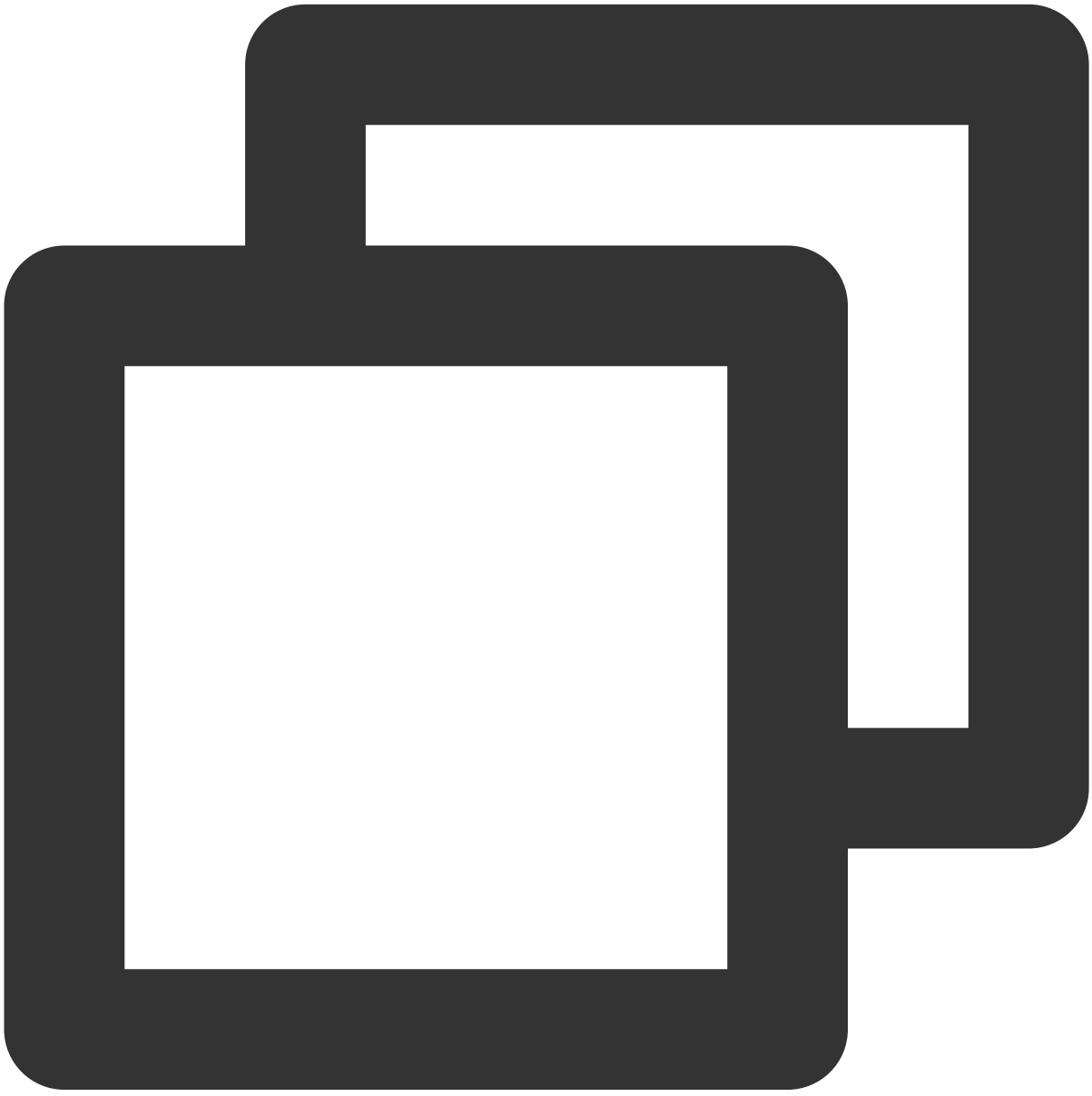
パラメータ	タイプ	意味

©2013-2022 Tencent Cloud. All rights reserved.

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.