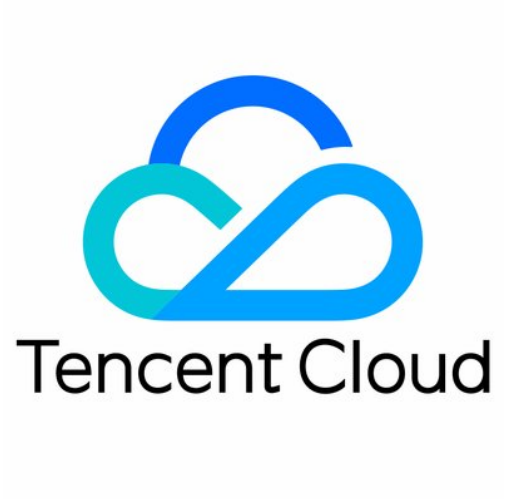
Tencent Real-Time Communication Solution

製品ドキュメント



カタログ：

Solution
Real-Time Chorus
Quick Integration (TUIKaraoke)
TUIKaraoke (iOS)の統合

最終更新日：：2024-07-19 15:32:54

コンポーネントの説明

TUIKaraokeはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIKaraokeコンポーネントを統合することにより、数行のコードを書くだけで、アプリケーションにオンラインカラオケシーンを組み込むことができ、カラオケ、マイク管理、ギフトの送付と受領、テキストチャットなどのTRTCのKTVシーンでの関連機能を体験できるようになります。TUIKaraokeはAndroidプラットフォーム用のソースコードもサポートしています。基本機能は下図のとおりです：

説明：

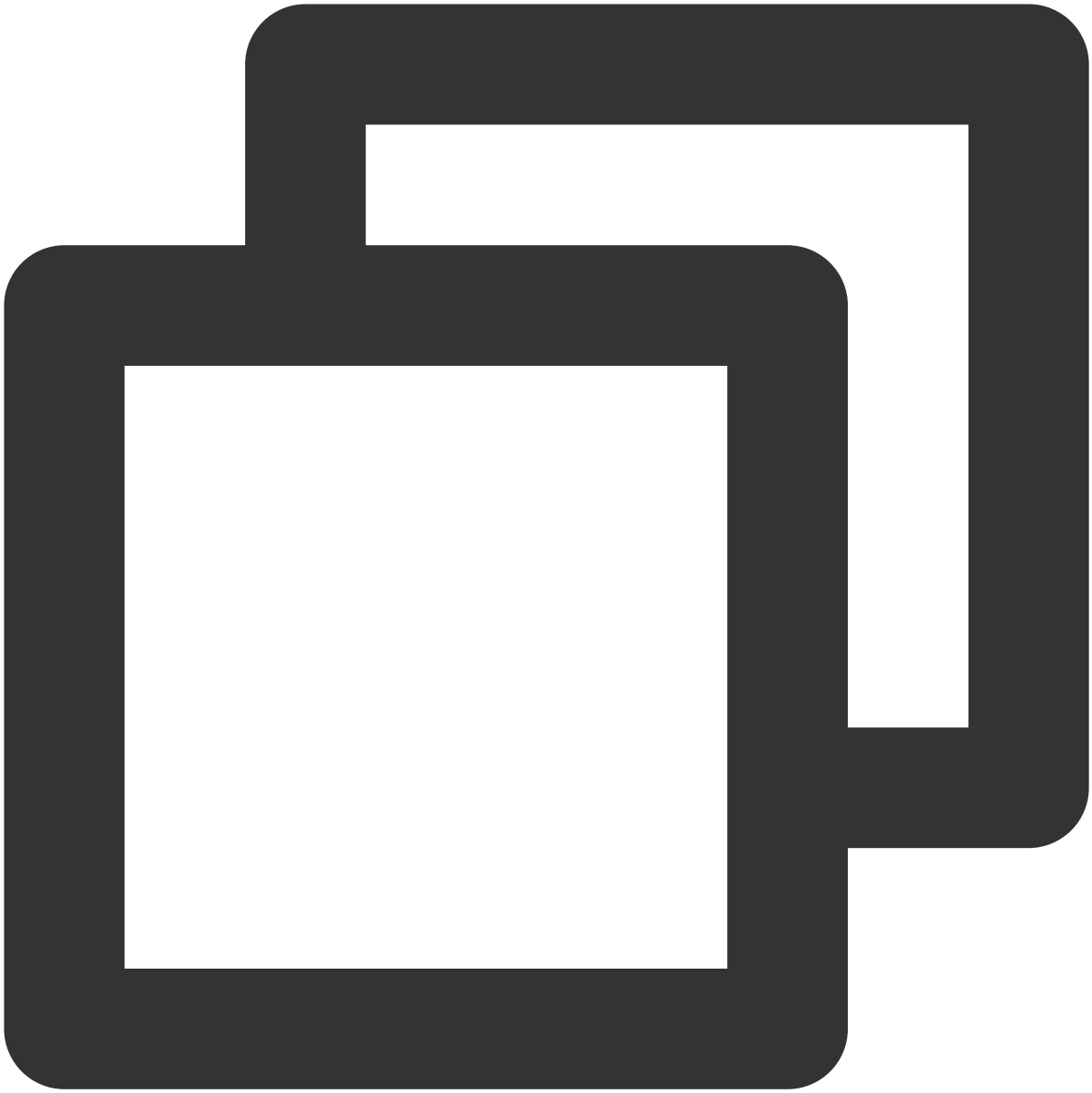
TUIKitシリーズコンポーネントはTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMサービスの課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。

コンポーネントの統合

ステップ1：TUIKaraokeコンポーネントのダウンロードとインポート

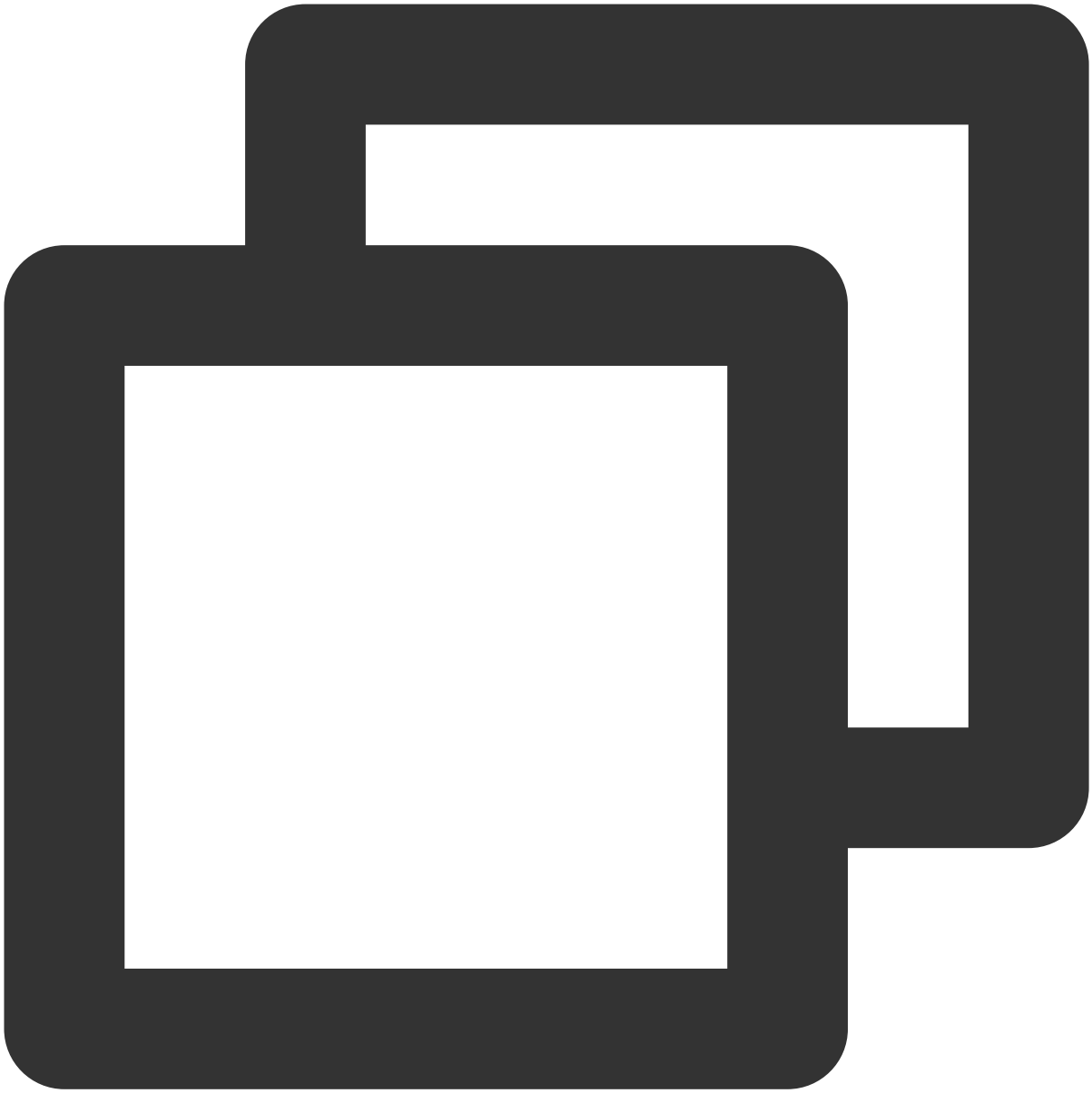
クリックしてGithubに進み、コードのクローン/ダウンロードを選択した後、iOSディレクトリ下の `Source` 、 `Resources` 、 `TXAppBasic` フォルダ、 `TUIKaraoke.podspec` ファイルをプロジェクトにコピーし、次のようにインポート動作を完了します。

`Podfile` ファイルにインポートコマンドを追加します。次をご参照ください：



```
pod 'TUIKaraoke', :path => "./", :subspecs => ["TRTC"]
pod 'TXLiteAVSDK_TRTC'
pod 'TXAppBasic', :path => "TXAppBasic/"
```

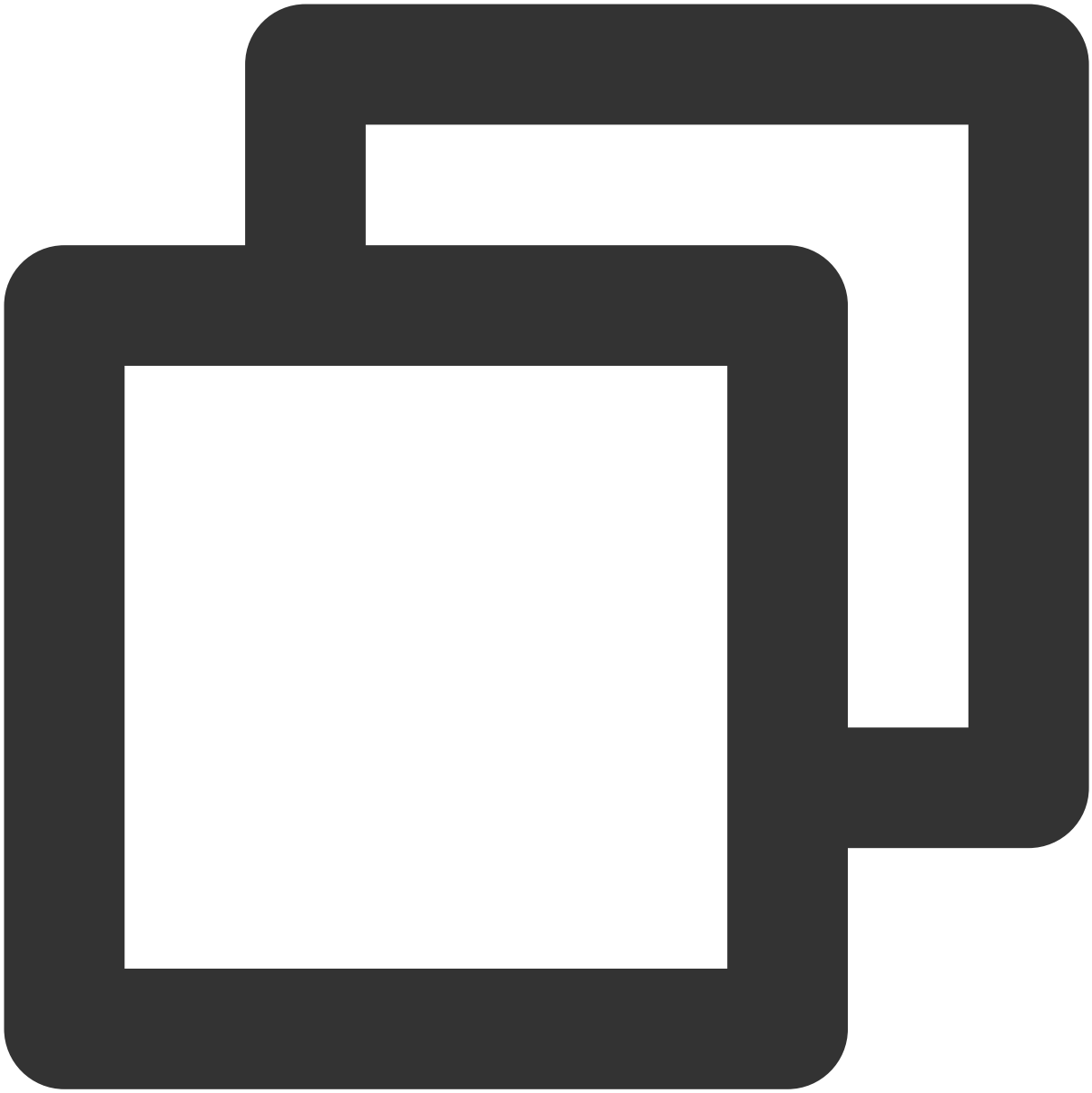
端末を開き、 `Podfile` ファイルのあるディレクトリ下に進み、インストールコマンドを実行します。次をご参照ください：



```
pod install
```

ステップ2：権限の設定

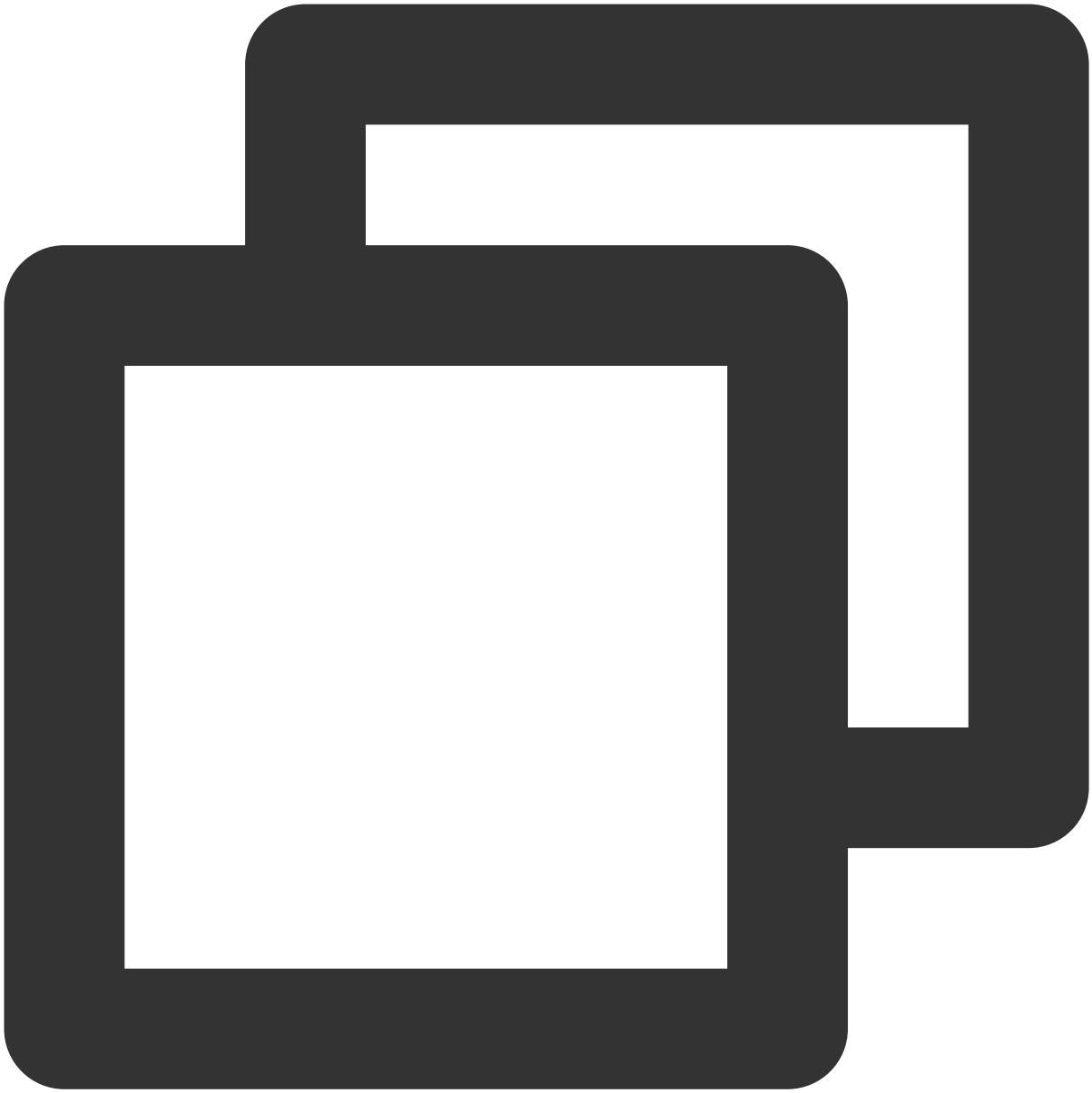
プロジェクトのinfo.plistファイルの中でAppの権限を設定します。SDKには以下の権限が必要です（iOSシステムではマイクを動的に申請してください）：



```
<key>NSMicrophoneUsageDescription</key>
    <string>Karaokeにはマイクへのアクセス権限が必要です</string>
```

ステップ3：初期化およびログイン

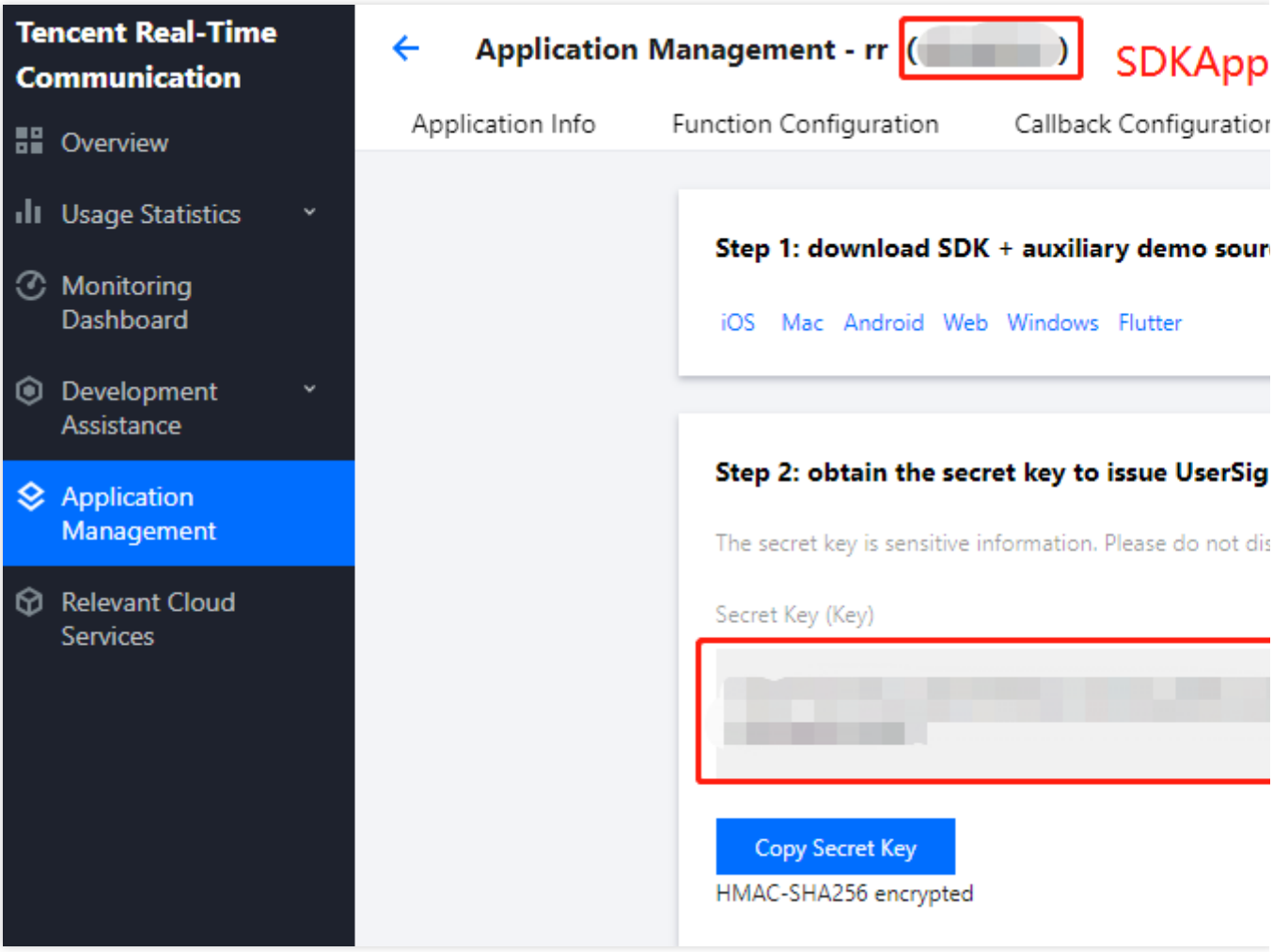
インターフェースに関する説明については、TUIKaraokeをご参照ください。



```
// 1.初期化
let karaokeRoom = TRTCKaraokeRoom.shared()
karaokeRoom.setDelegate(delegate: self)
// 2.ログイン
karaokeRoom.login(SDKAppID: Int32(SDKAppID), UserId: UserId, UserSig: ProfileMana
      if code == 0 {
          //ログイン成功
      }
}
```

パラメータの説明：

SDKAppID：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション情報**をクリックすると、SDKAppID情報が次の図のように表示されます：



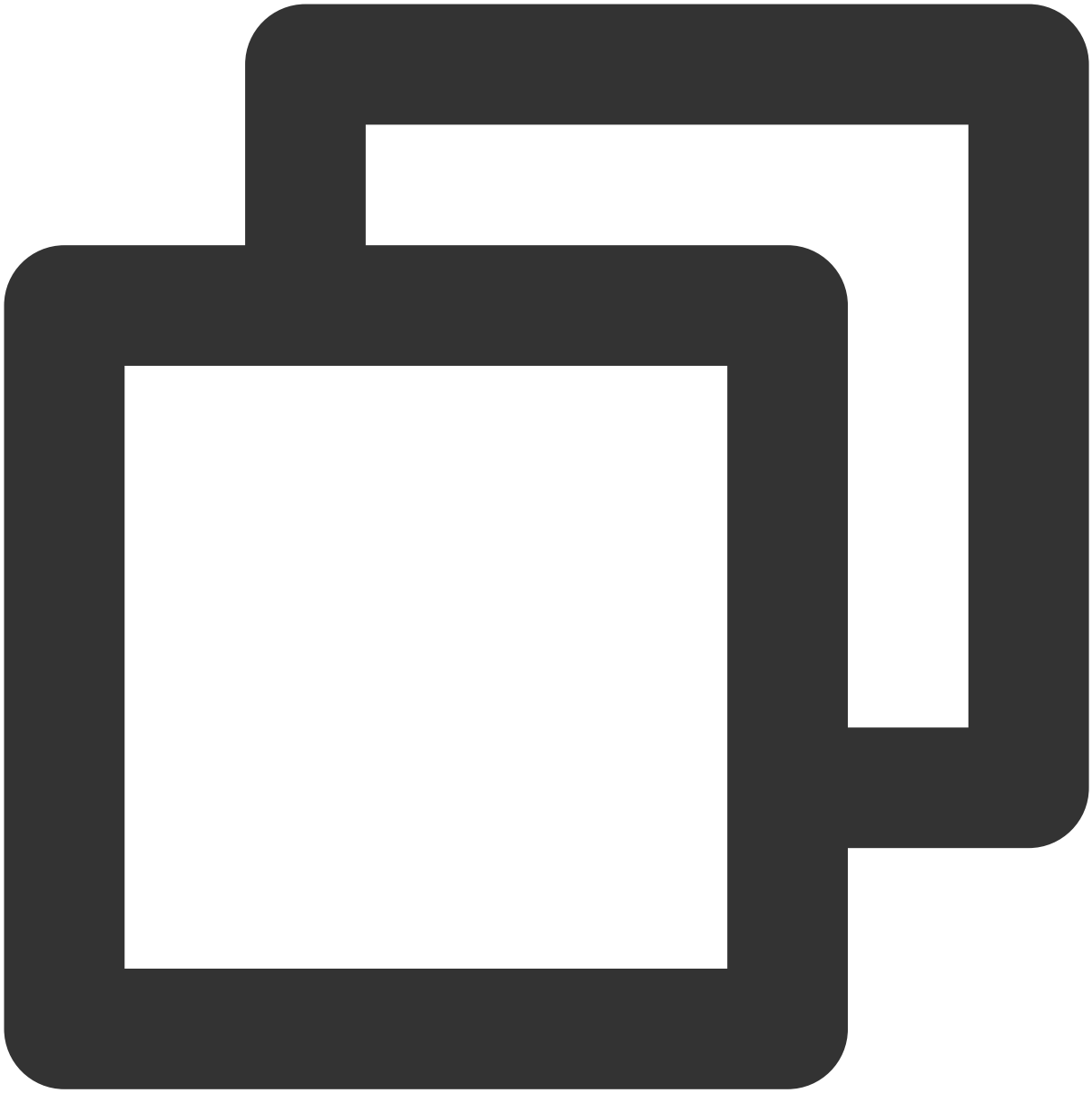
Secretkey：**TRTCアプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます：

userId：現在のユーザーのIDです。文字列タイプであり、長さは32バイト以内とし、特殊文字の使用はサポートしていません。英語または数字の使用をお勧めします。業務の実際のアカウントシステムと組み合わせてご自身で設定することができます。

userSig：SDKAppId、userId、Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のUserSigがオンラインで直接生成されます。その他の情報については、UserSigの計算、使用方法をご参照ください。

ステップ4：オンラインKTVシーンの実装

1. **キャスターがルームを作成**TUIKaraoke.createRoom

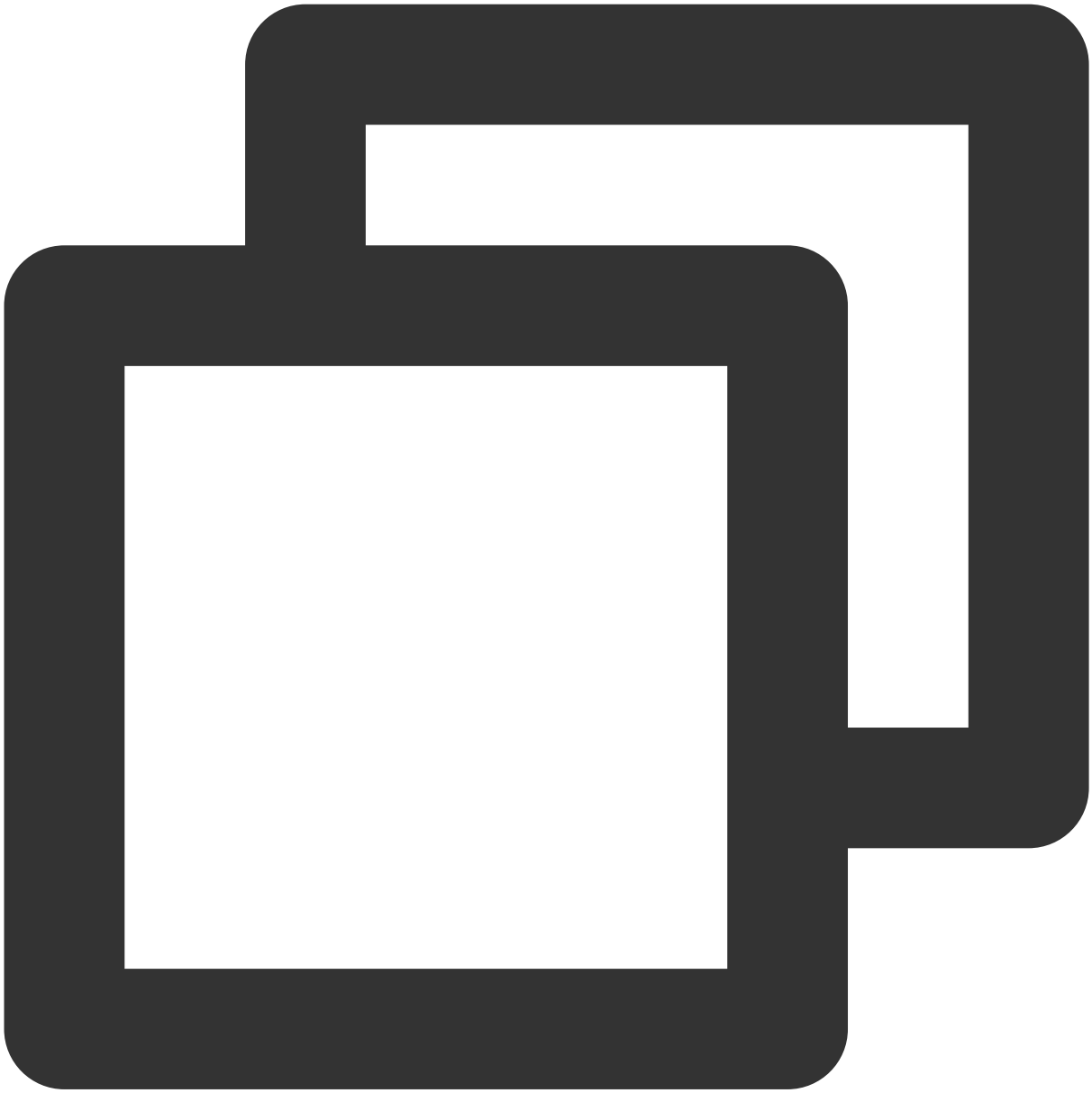


```
int roomId = "ルームID";
let param = RoomParam.init()
param.roomName = "ルーム名";
param.needRequest = false; // 管理者によるマイク・オン確認の要否
param.seatCount = 8;        //ルームの座席数。計8席あります
param.coverUrl = "ルームカバー図のURL";

karaokeRoom.createRoom(roomID: Int32(roomInfo.roomID), roomParam: param) { [weak se
 guard let `self` = self else { return }
 if code == 0 {
     //作成に成功
```

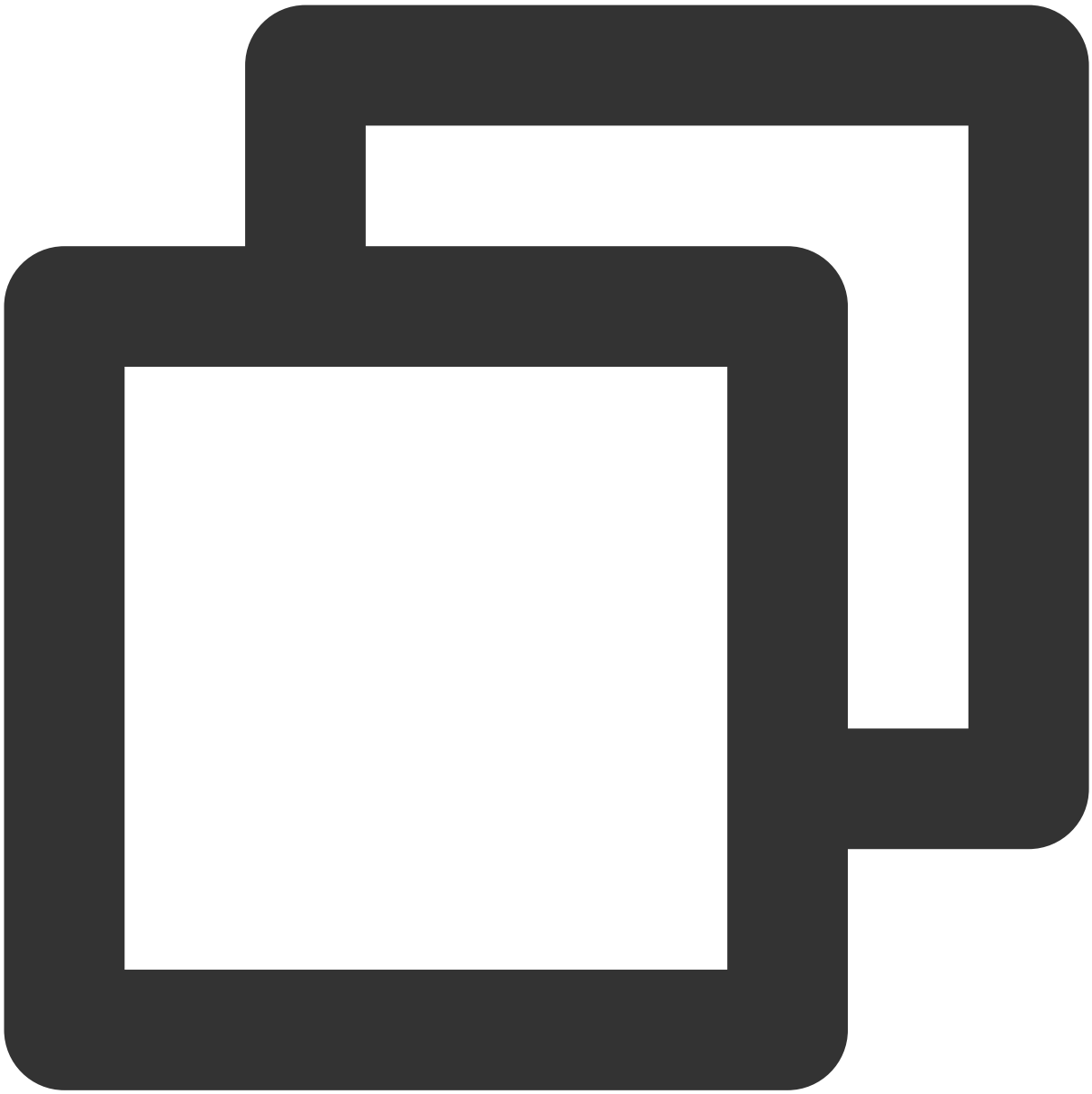
```
  }
}
```

2. リスナーが入室TUIKaraoke.enterRoom



```
karaokeRoom.enterRoom(roomID: roomInfo.roomID) { [weak self] (code, message) in
 guard let `self` = self else { return }
 if code == 0 {
     //入室に成功
 }
}
```

3. リスナーが自主的にマイク・オンTUIKaraoke.enterSeat



```
// 1.リスナーが呼び出してマイク・オン
int seatIndex = 1;
karaokeRoom.enterSeat(seatIndex: seatIndex) { [weak self] (code, message) in
 guard let `self` = self else { return }
 if code == 0 {
     //マイク・オン成功
 }
}
// 2.onSeatListChangeコールバックを受信し、マイクリストを更新します
```

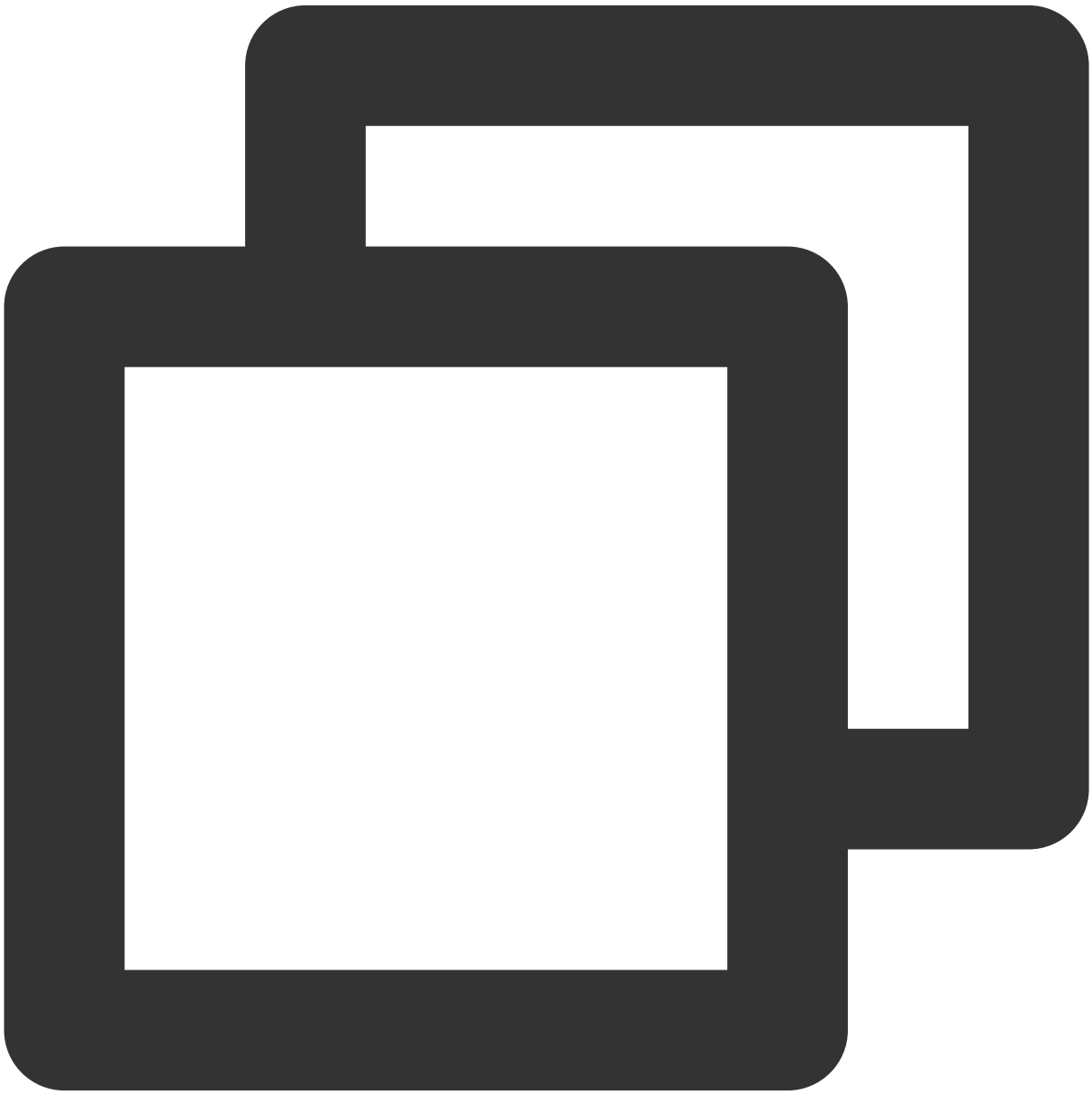
```
func onSeatListChange(seatInfoList: [SeatInfo]) {
}
```

説明：

マイク管理に関連するその他の操作については、TUIKaraokeインターフェースドキュメントをご参照の上、必要に応じて実装するか、またはTUIKaraokeデモプロジェクトをご参照ください。

4. **音楽再生とKTV体験シーンの実装**

ご自身の業務に応じて音楽IDおよびURLリンクを取得し、楽曲を再生できます。詳細についてはTUIKaraoke音楽再生インターフェースをご参照ください。

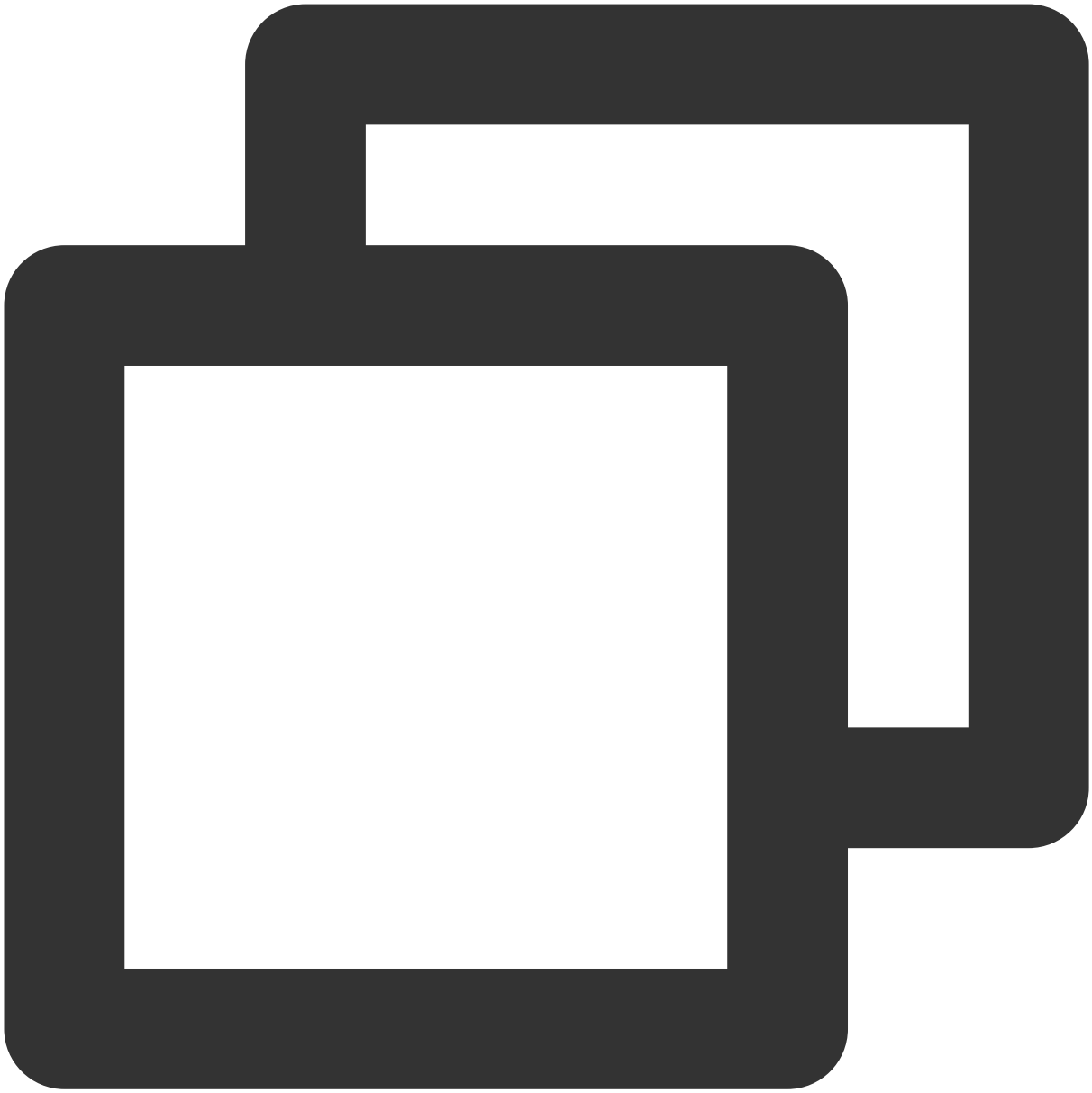


```
//音楽の再生
karaokeRoom.startPlayMusic(musicID: musicID, originalUrl: muscicLocalPath, accompan
//音楽の停止
karaokeRoom.stopPlayMusic();
```

上記の手順が完了すると、KTVの基本機能を実装できます。業務上、チャット、ギフト送付などの機能も必要な場合、次の機能を統合することができます。

ステップ6：テキストチャット機能（オプション）

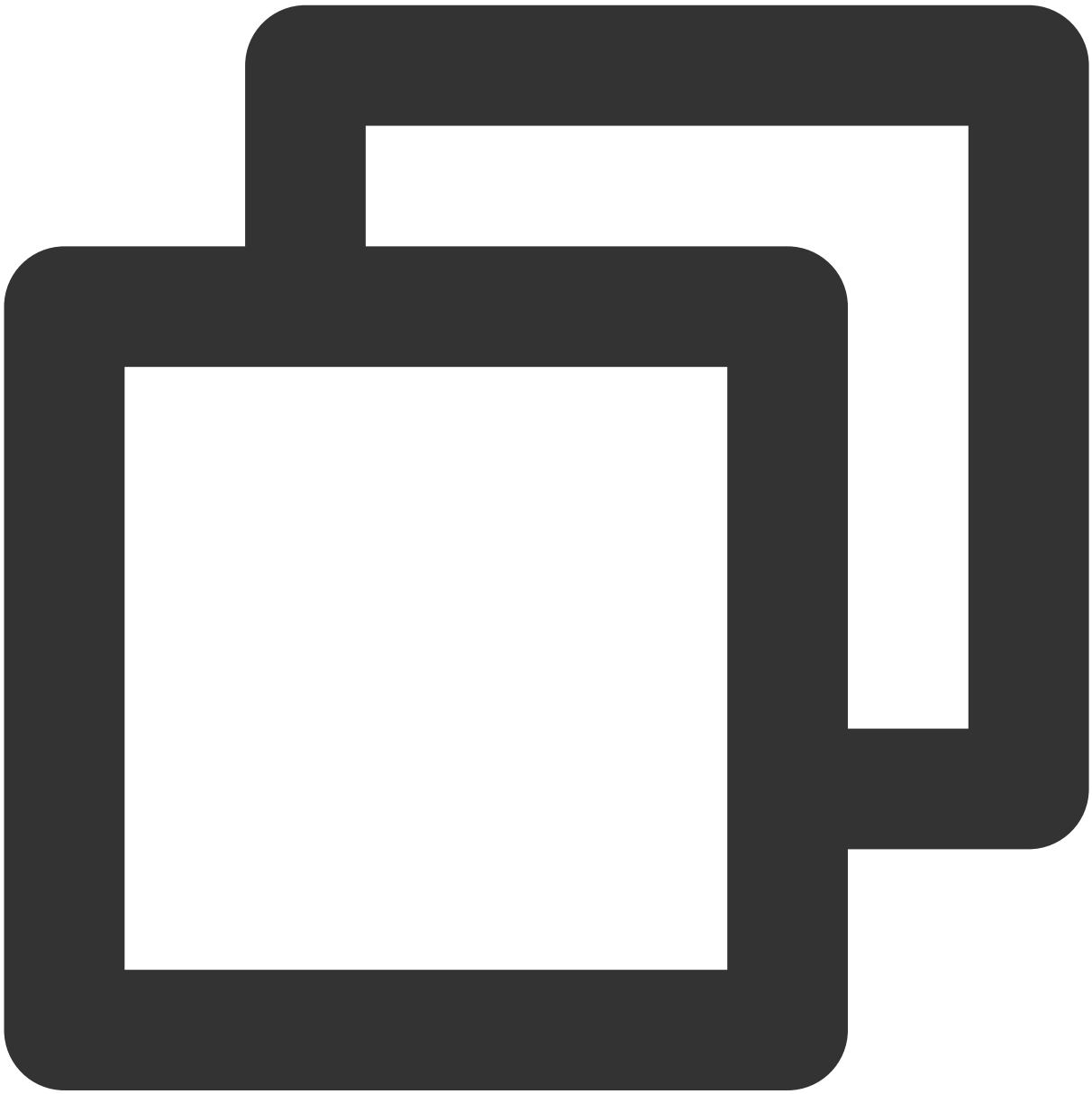
各キャスターまたはリスナー間のテキストチャット機能を実装したい場合は、次のメソッドによってチャットメッセージを送受信することができます。
インターフェースに関する説明については、TRTCKaraokeRoom.sendRoomTextMsgをご参照ください。



```
// 発信側：テキストメッセージの発信
karaokeRoom.sendRoomTextMsg(message: message) { [weak self] (code, message) in
    if code == 0 {
        //送信に成功
    }
}
// 受信側：テキストメッセージのモニタリング
karaokeRoom.setDelegate(delegate: self)
func onRecvRoomTextMsg(message: String, userInfo: UserInfo) {
    debugPrint("" + userInfo.userName + "から受信したメッセージ：" + message)
}
```

ステップ7：ギフト送付機能（オプション）

ギフト送付および受領機能を実装したい場合は、次のメソッドによってギフトを送付または受領し、表示することができます。



```
// 送信側：カスタマイズした「IMCMD_GIFT」によってギフトメッセージを区別
karaokeRoom.sendRoomCustomMsg(cmd: kSendGiftCmd, message: message) { code, msg in
    if (code == 0) {
        //送信に成功
    }
```

```
}

// 受信側：ギフトメッセージのモニタリング
karaokeRoom.setDelegate(delegate: self)
func onRecvRoomCustomMsg(cmd: String, message: String, userInfo: UserInfo) {
    if cmd == kSendGiftCmd {
        debugPrint("" + userInfo.userName + "から受領したギフト：" + message)
    }
}
```

よくあるご質問

TUIKaraokeコンポーネントはボイスチェンジ、キー調整、リバーブなどのオーディオエフェクト機能をサポートしていますか？

サポートしています。具体的にはTUIKaraokeデモプロジェクトをご参照ください。

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

TUIKaraoke (Android)の統合

最終更新日：：2024-07-19 15:32:55

コンポーネントの説明

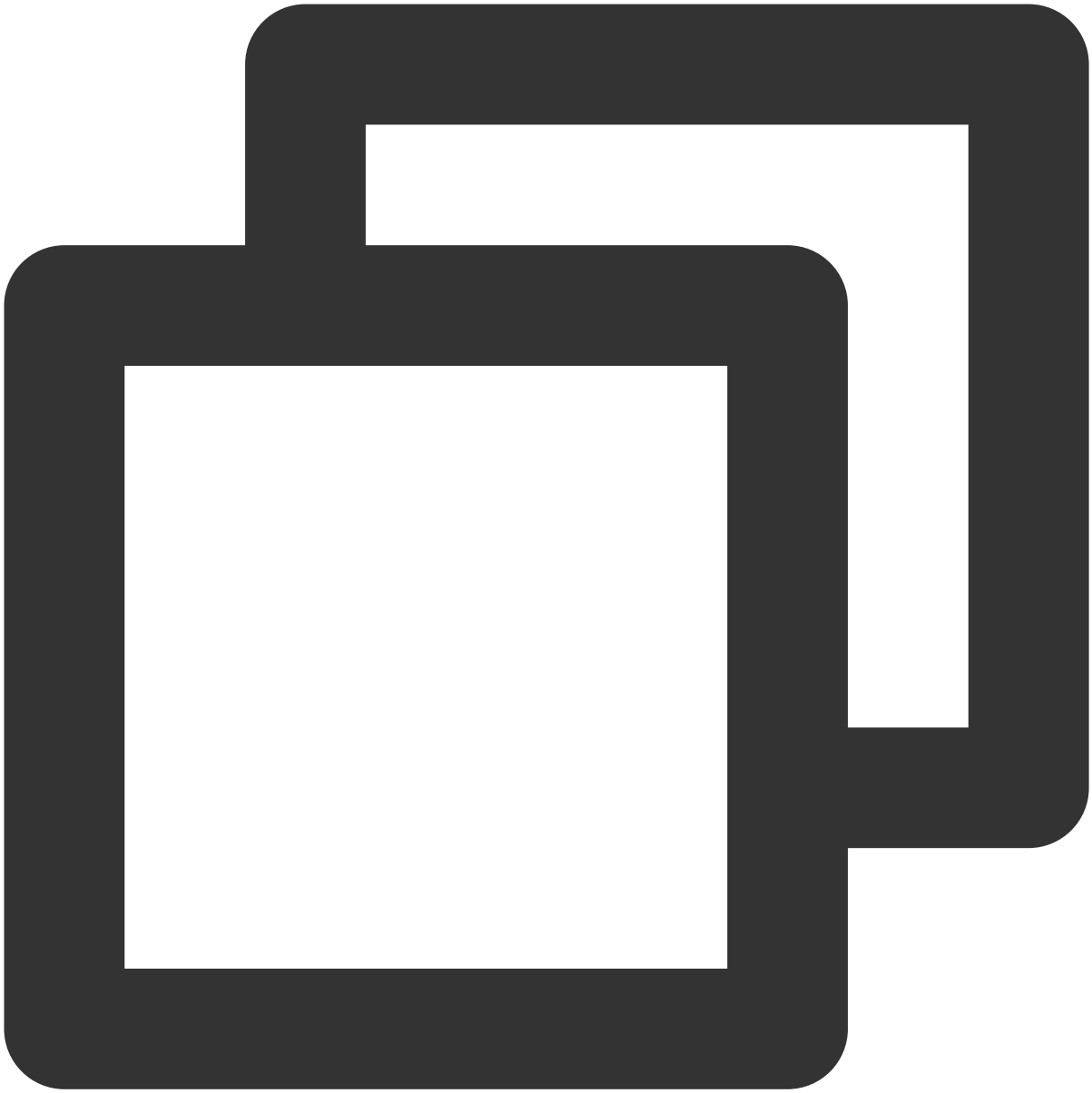
TUIKaraokeはオープンソースのオーディオビデオUIコンポーネントであり、プロジェクトにTUIKaraokeコンポーネントを統合することにより、数行のコードを書くだけで、アプリケーションにオンラインカラオケシーンを組み込むことができ、カラオケ、マイク管理、ギフトの送付と受領、テキストチャットなどのTRTCのKTVシーンでの関連機能を体験できるようになります。TUIKaraokeはiOSプラットフォーム用のソースコードもサポートしています。基本機能は下図のとおりです。

コンポーネントの統合

ステップ1：TUIKaraokeコンポーネントのダウンロードとインポート

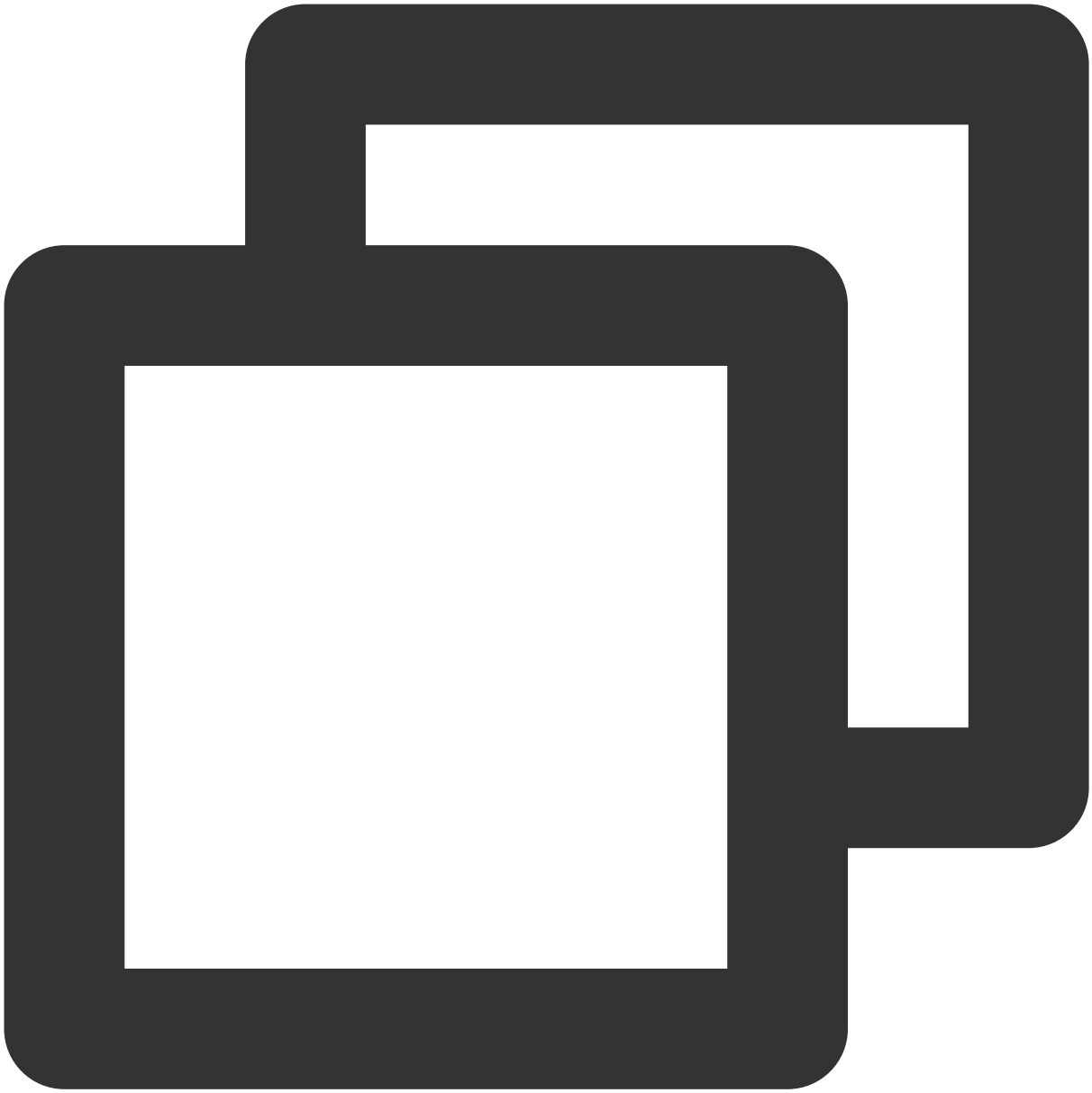
クリックしてGithubに進み、コードのクローン/ダウンロードを選択した後、Androidディレクトリ下のSource、Debugディレクトリをプロジェクトにコピーし、次のようにインポート動作を完了します。

`setting.gradle` へのインポートを完了します。以下をご参照ください。



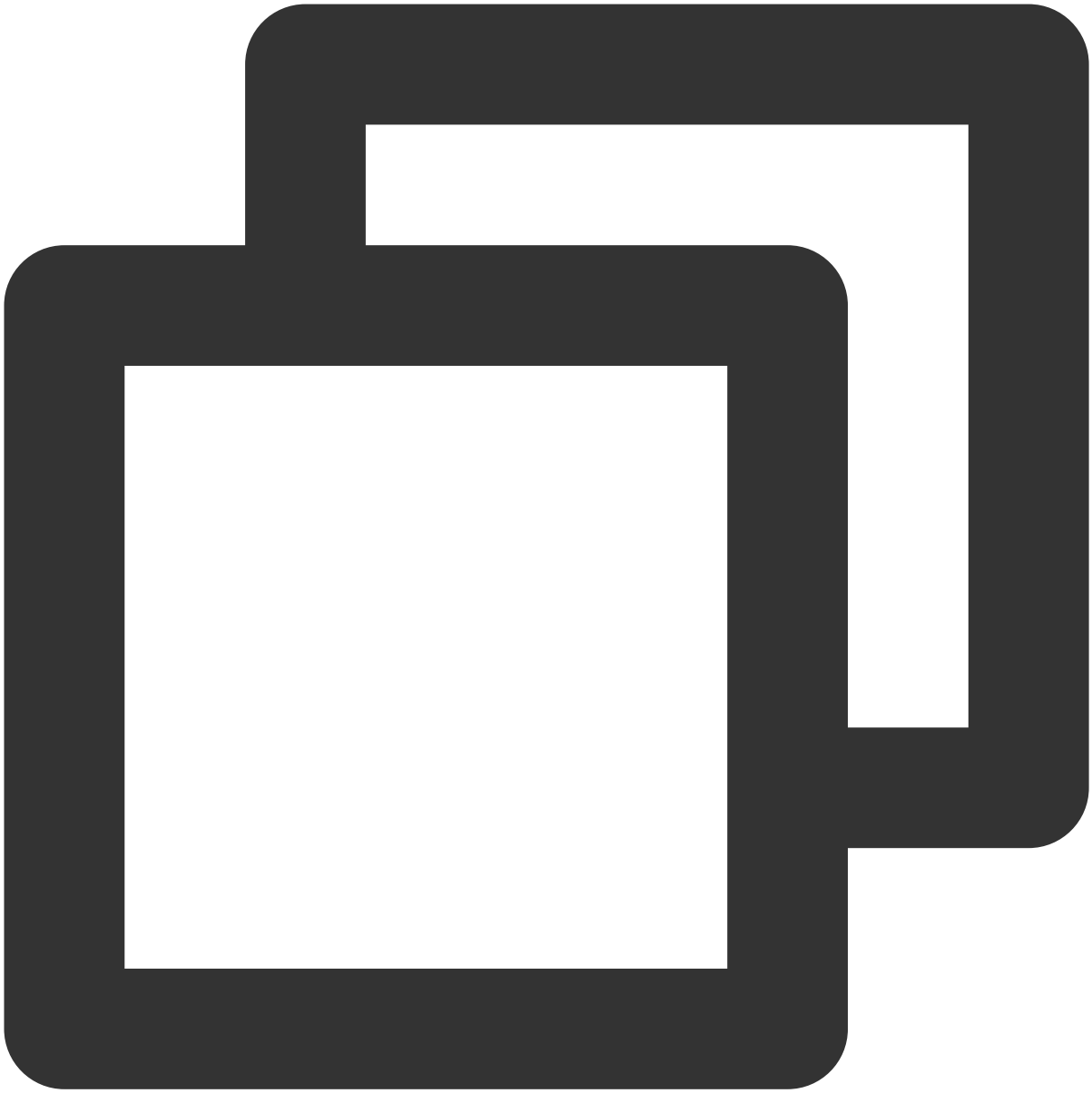
```
include ':Source'
include ':Debug'
```

appのbuild.gradleファイルにTUIKaraokeに対する依存関係を追加します。



```
api project(':Source')
```

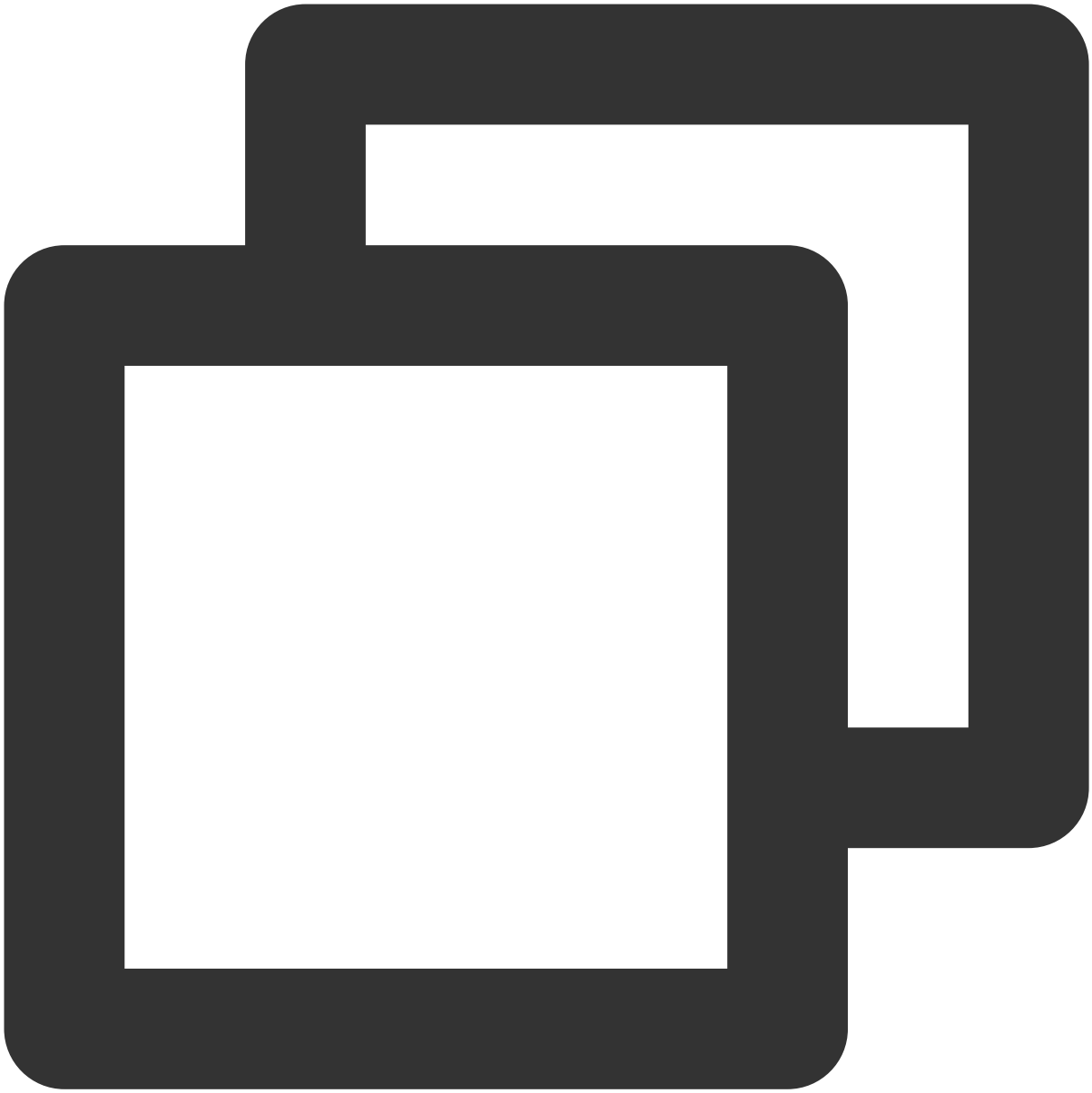
ルートディレクトリの `build.gradle` ファイルに `TRTC SDK` および `IM SDK` の依存関係を追加します。



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

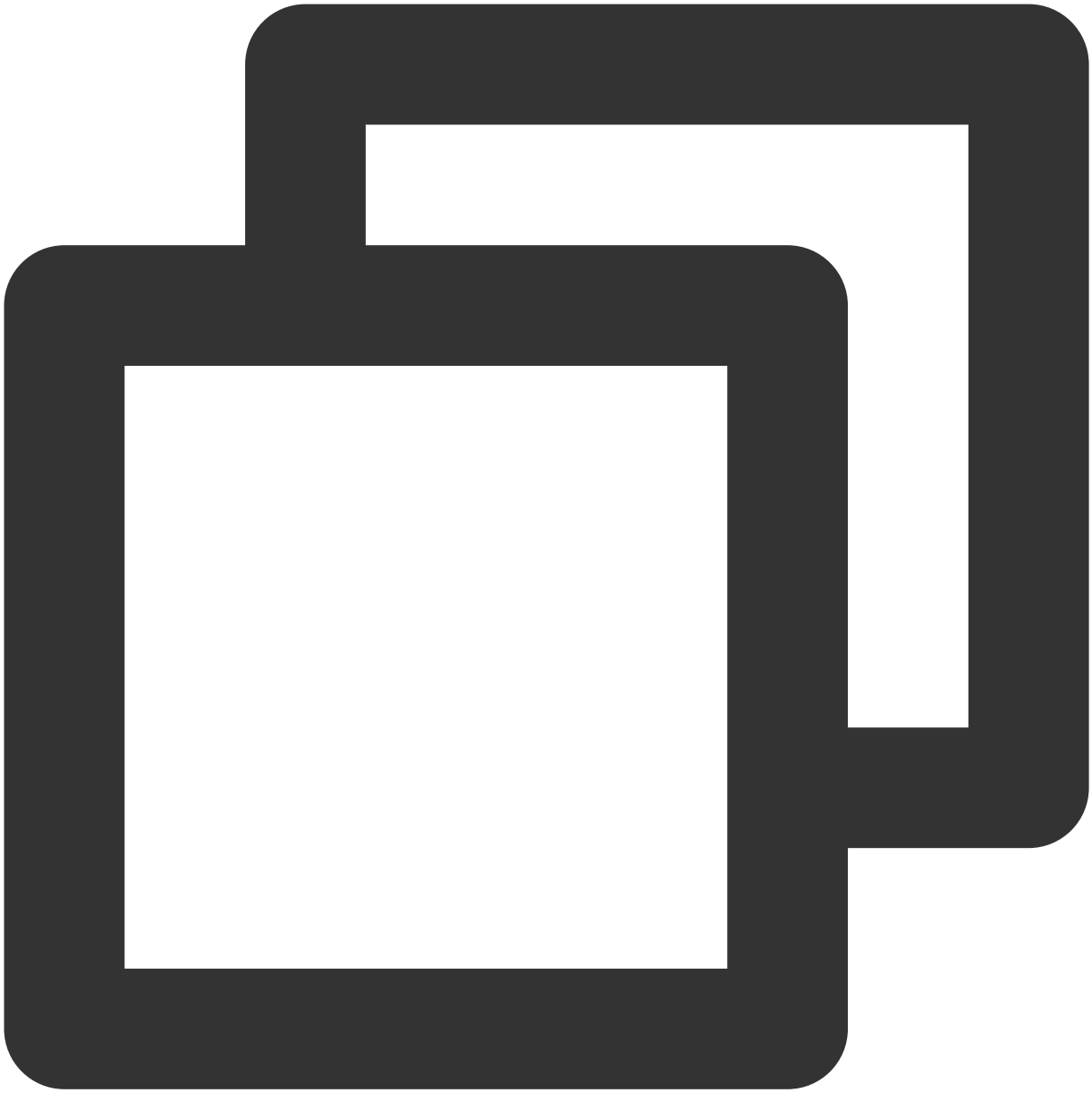
ステップ2：権限の設定および難読化ルール

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



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

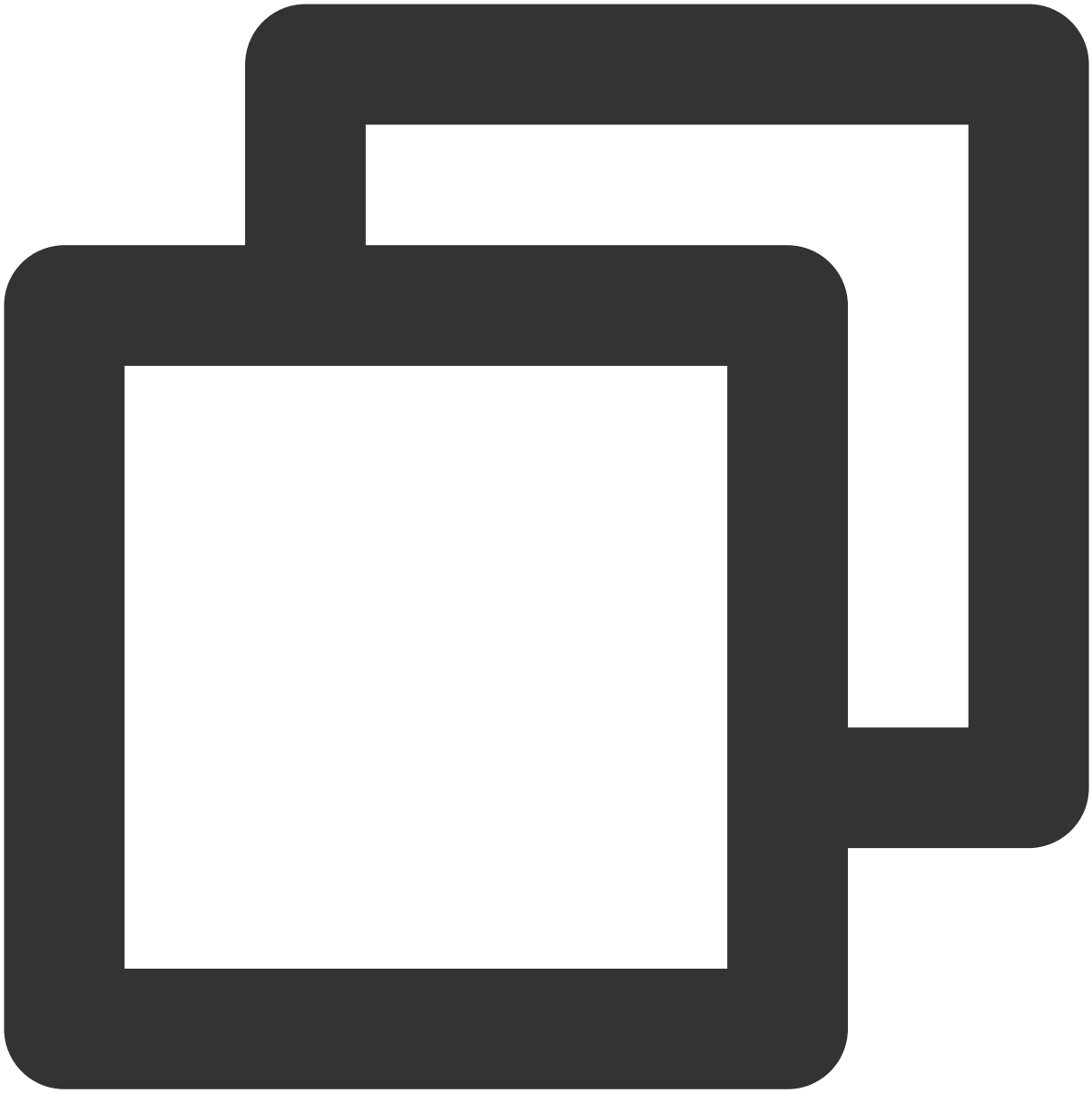
proguard-rules.proファイルでは、SDK関連を非難読化リストに追加します。



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

ステップ3：初期化およびログイン

インターフェースに関する詳細については、TUIKaraokeをご参照ください。

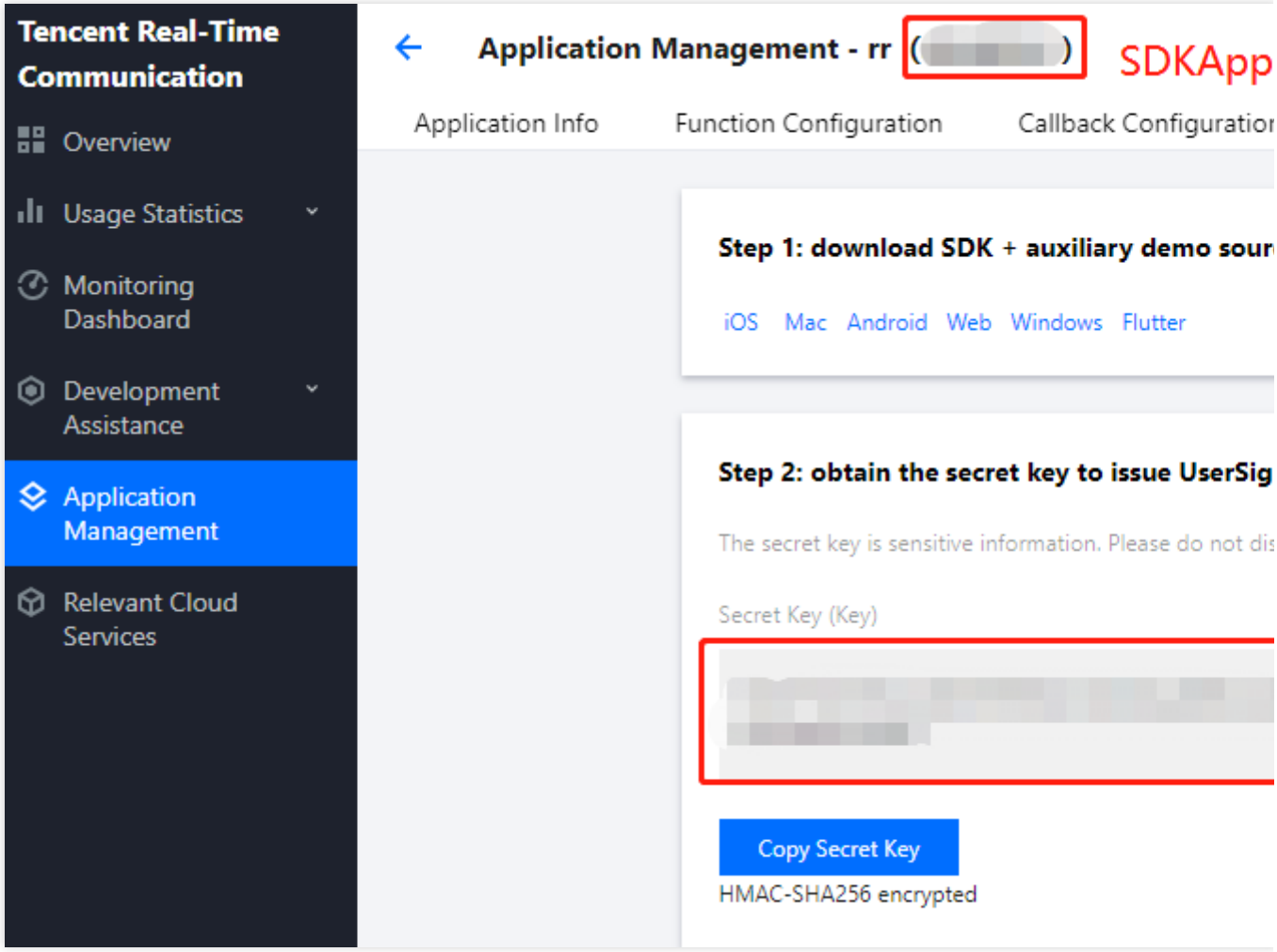


```
// 1.初期化
TRTCKaraokeRoom mTRTCKaraokeRoom = TRTCKaraokeRoom.sharedInstance(this);
mTRTCKaraokeRoom.setDelegate(this);
// 2.ログイン
mTRTCKaraokeRoom.login(SDKAppID, UserID, UserSig, new TRTCKaraokeRoomCallback.Act
    @Override
    public void onCallback(int code, String msg) {
        if (code == 0) {
        //ログイン成功
        }
    }
```

```
    });
```

パラメータの説明：

SDKAppID：**TRTCアプリケーションID**です。Tencent Cloud TRTCサービスをアクティブ化していない場合は、Tencent Cloud TRTCコンソールに進み、新しいTRTCアプリケーションを作成した後、**アプリケーション情報**をクリックすると、SDKAppID情報が次の図のように表示されます。



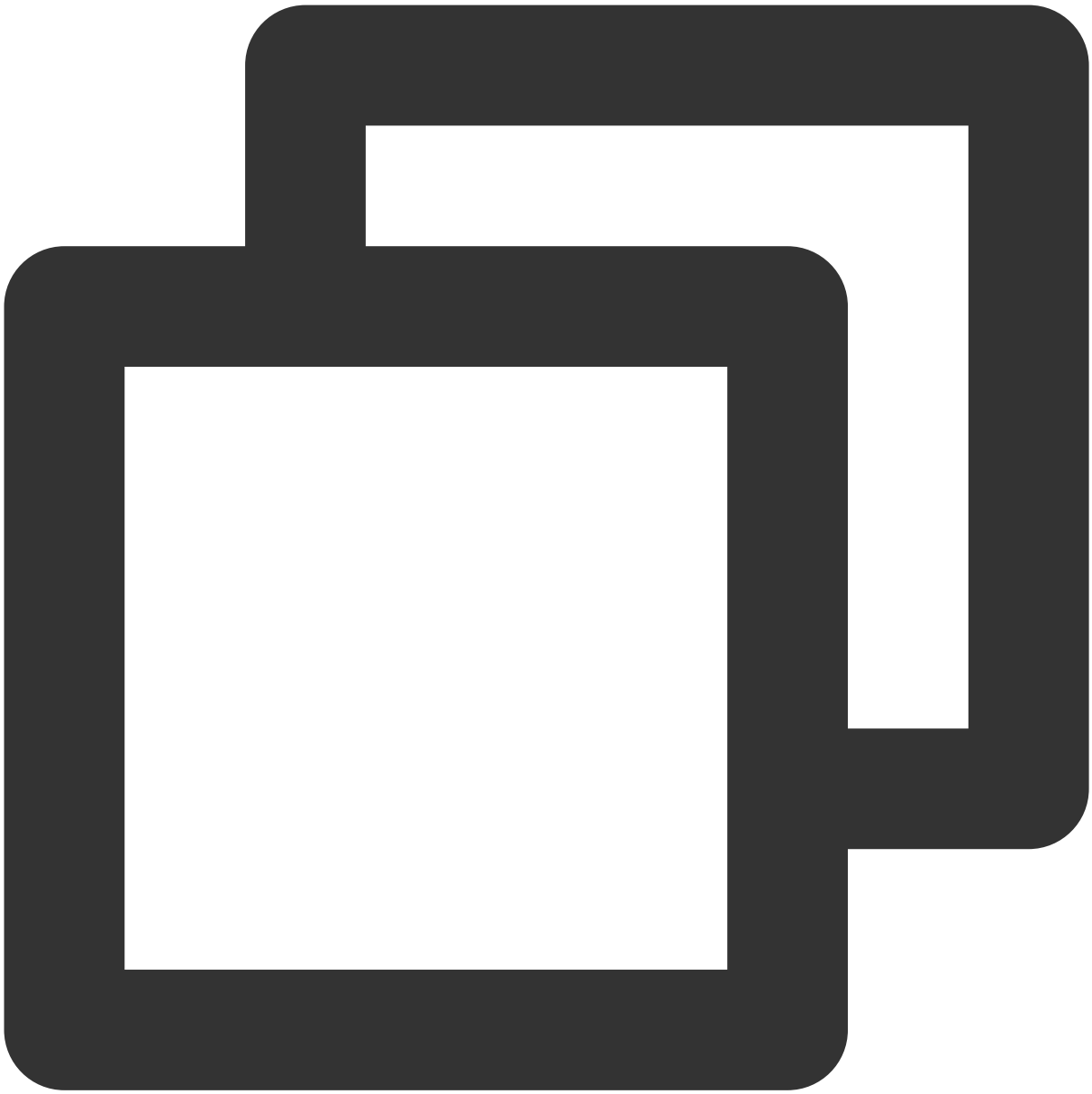
Secretkey：**TRTCアプリケーションキー**であり、SDKAppIdに対応しています。TRTCアプリケーション管理に進むと、SecretKey情報が上の図のように表示されます。

userId：現在のユーザーのIDです。文字列形式で、長さは32バイト以内とし、特殊文字の使用はサポートしていません。英語または数字の使用をお勧めします。業務の実際のアカウントシステムと組み合わせてご自身で設定することができます。

userSig：SDKAppId、userId、Secretkeyなどの情報に基づく計算によって得られるセキュリティ保護署名です。ここをクリックするとデバッグ用のUserSigがオンラインで直接生成されます。UserSigの計算、使用方法をご参照ください。

ステップ4：オンラインKTVシーンの実装

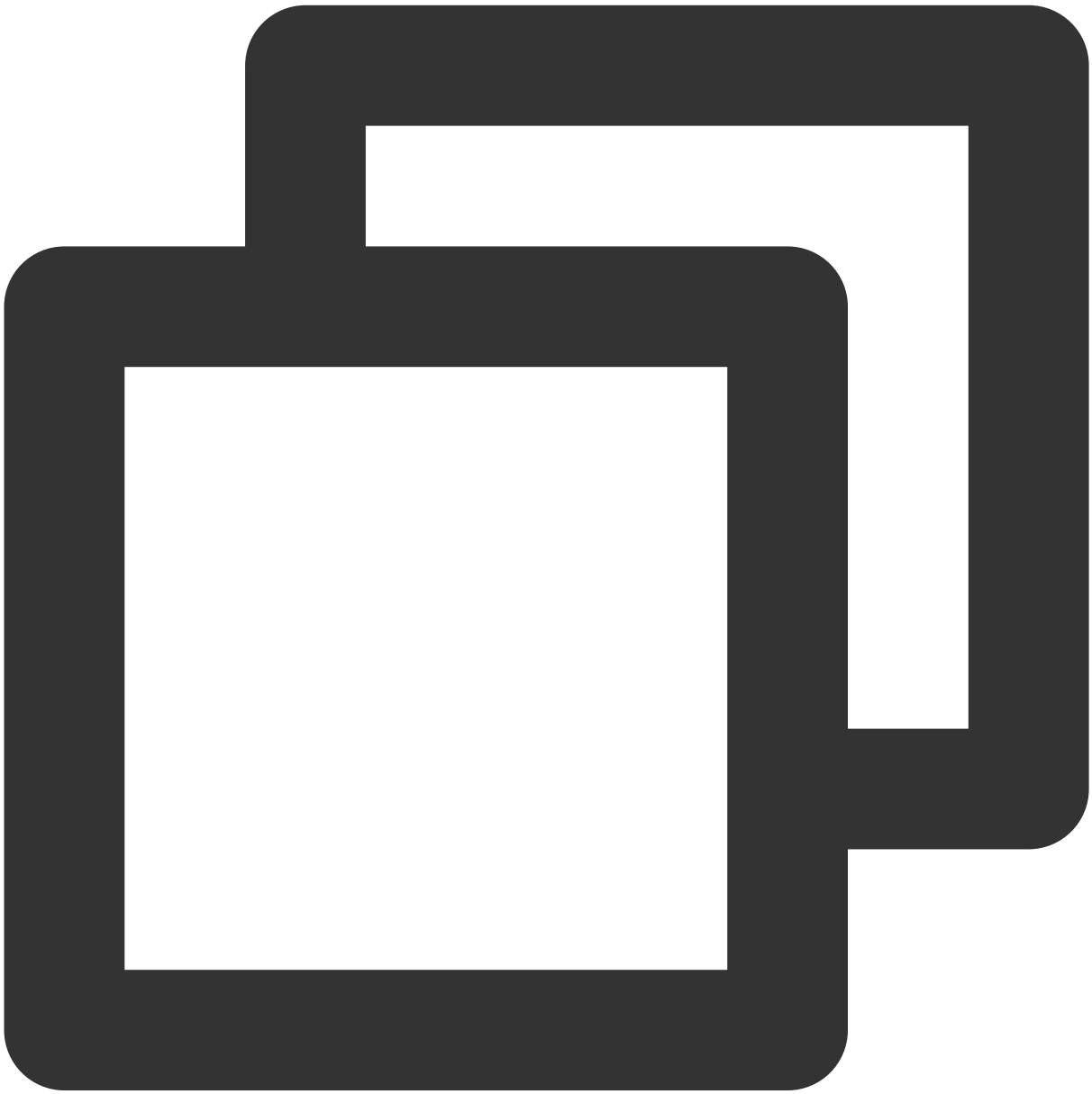
1. キャスターがルームを作成TUIKaraoke.createRoom



```
int roomId = "ルームID";
TRTCKaraokeRoomDef.RoomParam roomParam = new TRTCKaraokeRoomDef.RoomParam();
roomParam.roomName = "ルーム名";
roomParam.needRequest = false; // マイク・オンに対する管理者の確認の要否
roomParam.seatCount = 8;        //ルームの座席数。計8席あります
roomParam.coverUrl = "ルームカバー図のURL";
mTRTCKaraokeRoom.createRoom(roomId, roomParam, new TRTCKaraokeRoomCallback.ActionCa
 @Override
 public void onCallback(int code, String msg) {
```

```
    if (code == 0) {
      //作成に成功
    }
  }
});
```

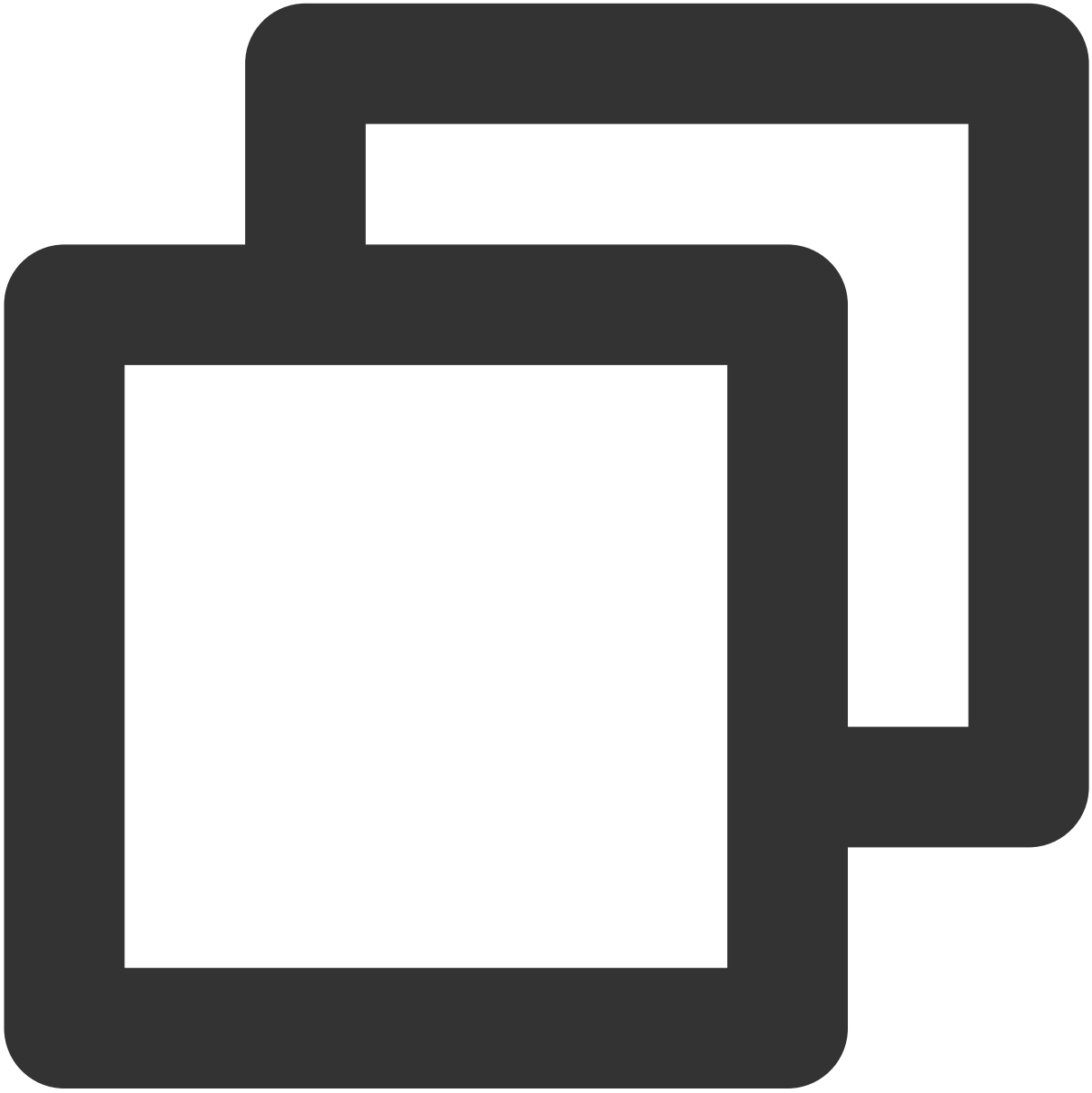
2. リスナーが入室TUIKaraoke.enterRoom



```
mTRTCKaraokeRoom.enterRoom(roomId, new TRTCKaraokeRoomCallback.ActionCallback() {
 @Override
 public void onCallback(int code, String msg) {
     if (code == 0) {
```

```
        //入室に成功
        }
    }
});
```

3. リスナーが自主的にマイク・オン TUIKaraoke.enterSeat



```
// 1.リスナーが呼び出してマイク・オン
int seatIndex = 1;
mTRTCKaraokeRoom.enterSeat(seatIndex, new TRTCKaraokeRoomCallback.ActionCallback()
 @Override
 public void onCallback(int code, String msg) {
```

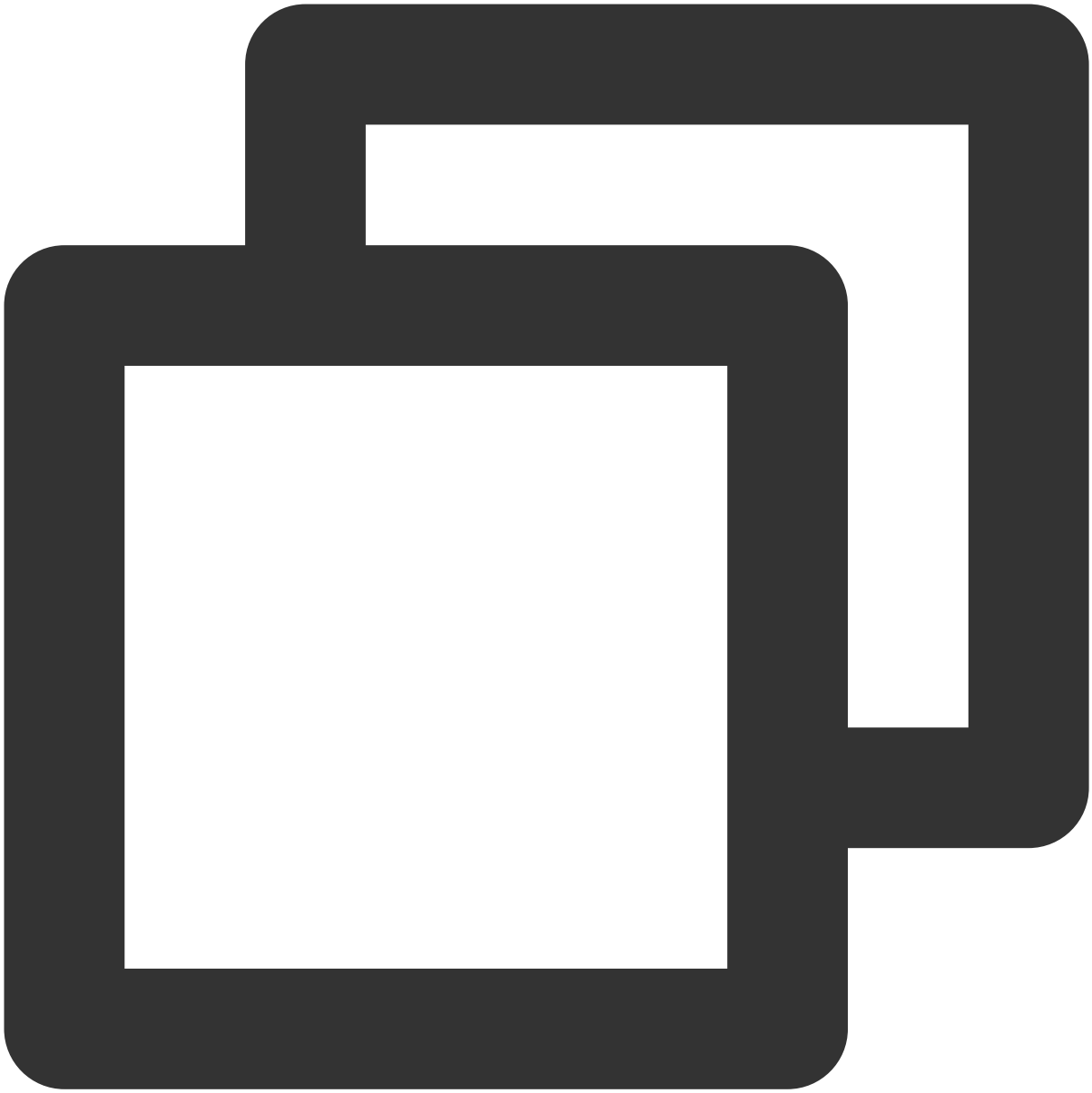
```
        if (code == 0) {
        //マイク・オン成功
        }
    }
});
// 2.onSeatListChangeコールバックを受信し、マイクリストを更新します
@Override
public void onSeatListChange(final List<TRTCKaraokeRoomDef.SeatInfo> seatInfoList)
}
```

説明：

マイク管理に関連するその他の操作については、TUIKaraokeインターフェースドキュメントをご参照の上、必要に応じて実装するか、または当社のTUIKaraokeデモプロジェクトをご参照ください。

4. 音楽再生とKTV体験シーンの実装

ご自身の業務に応じて音楽IDおよびURLリンクを取得し、楽曲を再生できます。インターフェースの詳細についてはTUIKaraoke音楽再生インターフェースをご参照ください。

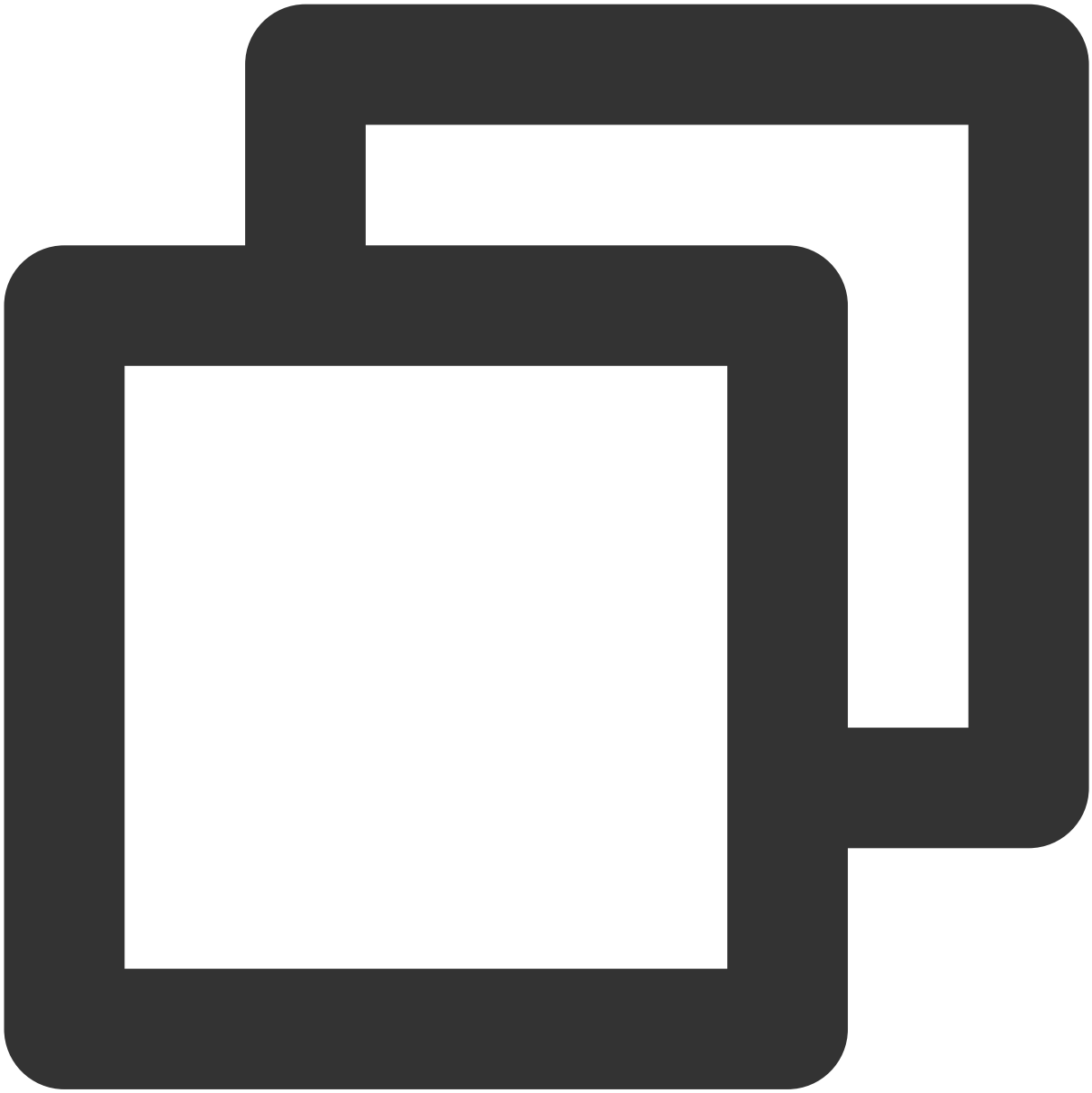


```
//音楽の再生
mTRTCKaraokeRoom.startPlayMusic(musicID,url);
//音楽の停止
mTRTCKaraokeRoom.stopPlayMusic();
```

上記の手順が完了すると、KTVの基本機能を実装できます。業務上、テキストチャット、ギフト送付などの機能も必要な場合、次の機能を統合することができます。

ステップ5：テキストチャット機能（オプション）

各キャスターまたはリスナー間のテキストチャット機能を実装したい場合は、次のメソッドによってチャットメッセージを送受信することができます。
インターフェースに関する詳細については、TRTCKaraokeRoom.sendRoomTextMsgをご参照ください。

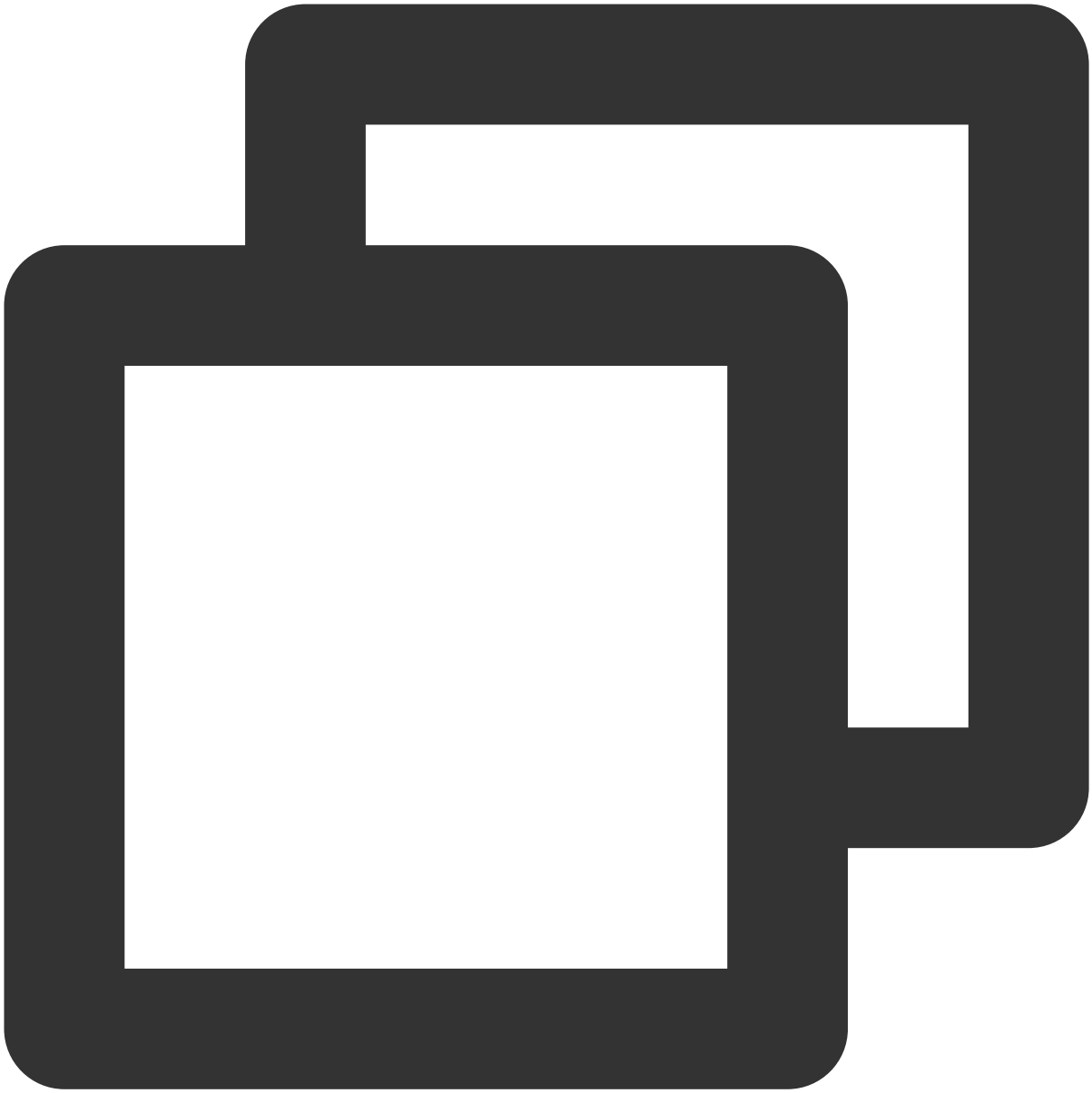


```
// 発信側：テキストメッセージの発信
mTRTCKaraokeRoom.sendRoomTextMsg("Hello Word!", new TRTCKaraokeRoomCallback.ActionC
    @Override
    public void onCallback(int code, String msg) {
        if (code == 0) {
            //送信に成功
        }
```

```
    }
});
// 受信側：テキストメッセージのモニタリング
mTRTCKaraokeRoom.setDelegate(new TRTCKaraokeRoomDelegate() {
  @Override
  public void onRecvRoomTextMsg(String message, TRTCKaraokeRoomDef.UserInfo userInf
      Log.d(TAG,"が" + userInfo.userName + "から受信したメッセージ:" + message);
  }
});
```

ステップ6：ギフト送付機能（オプション）

ギフト送付および受領機能を実装したい場合は、次のメソッドによってギフトを送付または受領し、表示することができます。



```
// 送信側：カスタマイズした「CMD_GIFT」によってギフトメッセージを区別
mTRTCKaraokeRoom.sendRoomCustomMsg("CMD_GIFT",date, new TRTCKaraokeRoomCallback.Act
    @Override
    public void onCallback(int code, String msg) {
        if (code == 0) {
            //送信に成功
        }
    }
});

// 受信側：ギフトメッセージのモニタリング
```

```
mTRTCKaraokeRoom.setDelegate(new TRTCKaraokeRoomDelegate() {
    @Override
    public void onRecvRoomCustomMsg(String cmd, String message, TRTCKaraokeRoomDef.
        if ("CMD_GIFT".equals(cmd)) {
            // ギフトメッセージの受信
            Log.d(TAG, "" + userInfo.userName + "からのギフトを受信:" + message);
        }
    }
});
```

よくあるご質問

TUIKaraokeコンポーネントはボイスチェンジ、キー調整、リバーブなどのオーディオエフェクト機能をサポートしていますか。

サポートしています。

説明：

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

TUIKaraoke APIのクエリー
TRTCKaraoke(iOS)

最終更新日：：2024-07-19 15:32:54

TRTCKaraokeRoomは、Tencent CloudのTRTCおよびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。
管理者が作成した新しいKaraokeルームが配信を開始すると、リスナーはKaraokeルームに入室して聴取/インタラクションを行います。
管理者は、楽曲の順序を管理し、マイク・オンのキャスターをキックアウトすることもできます。
管理者はまた、座席をクローズすることができ、その他のリスナーはマイク・オンを申請することができなくなります。
リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、マイク・オン後は楽曲の選択や歌唱ができるようになります。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
ギフトや各種のテキストメッセージ、カスタムメッセージの送信をサポートします。カスタムメッセージを弾幕、「いいね」などを実装するために使用することができます。
説明：
TUIKitシリーズコンポーネントはTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMサービスの課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブにします。
TRTCKaraokeRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実現プロセスは、 Karaoke（iOS）をご参照ください。
TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用しています。
IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCKaraokeRoom API概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。

setDelegate	イベントコールバックを設定します。
delegateQueue	イベントコールバックが配置されているスレッドを設定します。
login	ログイン。
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。nilの場合は、ルーム内全員の情報を取得します。

音楽再生インターフェース

API	説明
startPlayMusic	音楽の再生を開始します。
stopPlayMusic	音楽の再生を停止します。
pausePlayMusic	音楽の再生を一時停止します。
resumePlayMusic	音楽の再生を再開します。

マイク管理インターフェース

API	説明
enterSeat	ユーザーが発言者になります（リスナー側/管理者ともに呼び出し可）。
leaveSeat	ユーザーが視聴者になります（キャスターが呼び出し）。

pickSeat	視聴者が発言できるように招待（管理者が呼び出し）。
kickSeat	キックアウトしてマイク・オフ（管理者が呼び出し）。
muteSeat	任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
closeSeat	任意のマイクのクローズ/解除（管理者が呼び出し）。

ローカルのオーディオ操作インターフェース

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
setAudioQuality	音質の設定。
muteLocalAudio	ローカルオーディオミュートの開始/停止。
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。
setVoiceEarMonitorEnable	インイヤーモニタリングのオン/オフ。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerを取得します。

メッセージ送信関連インターフェース

API	説明

sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。
sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

招待シグナリング関連インターフェース

API	説明
sendInvitation	ユーザーに招待を送信。
acceptInvitation	招待の同意。
rejectInvitation	招待の辞退。
cancelInvitation	招待の取り消し。

TRTCKaraokeRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが廃棄された時のコールバック。
onRoomInfoChange	ボイスチャットルーム情報変更のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

API	説明
onSeatListChange	全量のマイクリストの変更。

onAnchorEnterSeat	発言者のメンバーがいます（ユーザーが発言者になります/管理者が視聴者を発言できるように招待）。
onAnchorLeaveSeat	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。
onSeatMute	管理者のマイクミュート。
onUserMicrophoneMute	ユーザーのマイクがミュートされているかどうか。
onSeatClose	管理者のマイククローズ。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージを受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

シグナリングイベントのコールバック

API	説明
onReceiveNewInvitation	新規招待リクエストを受信。
onInviteeAccepted	被招待者が招待に同意。
onInviteeRejected	被招待者が招待を拒否。
onInvitationCancelled	招待者が招待を取り消し。

楽曲イベントコールバック

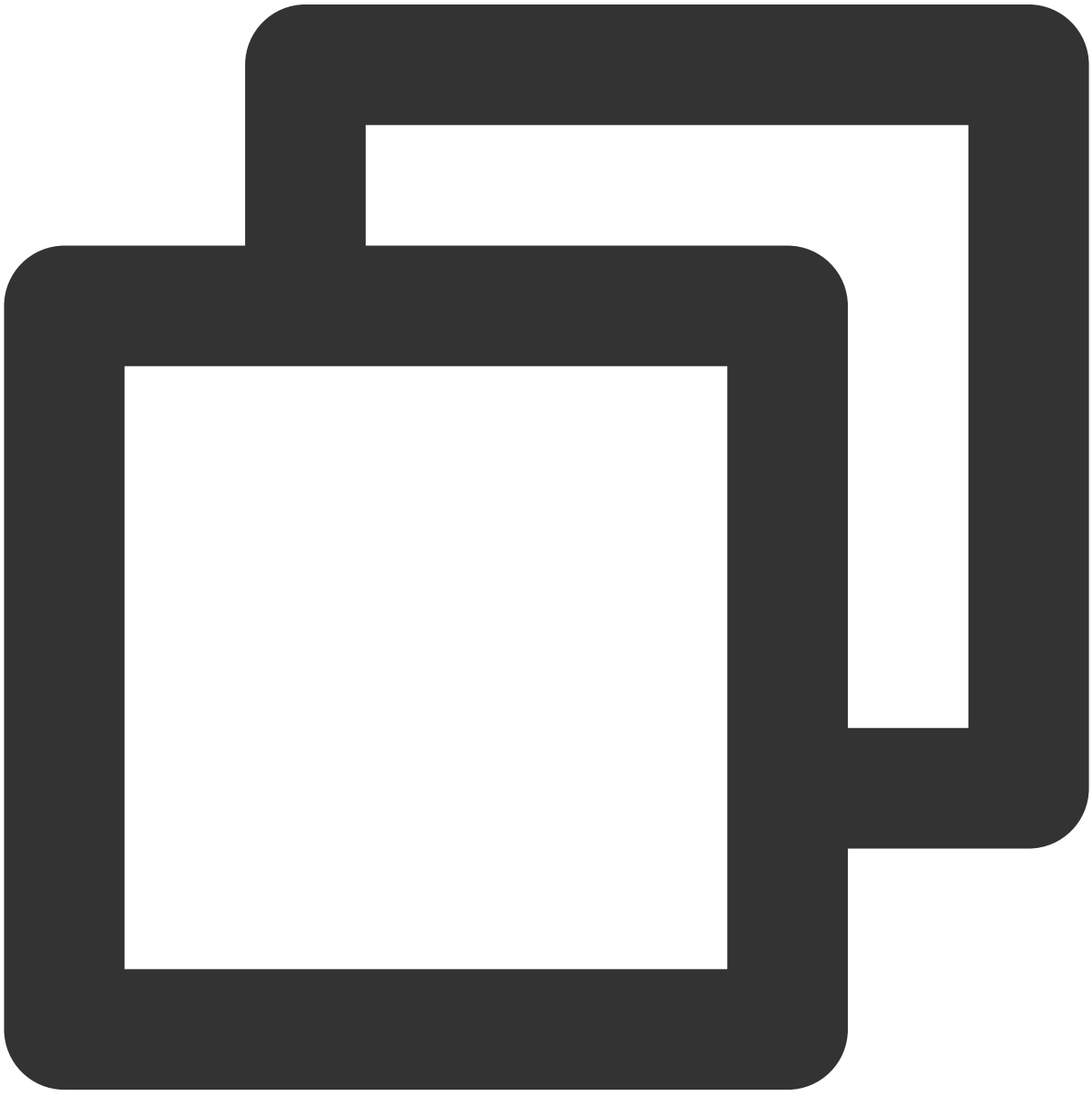
API	説明
onMusicProgressUpdate	楽曲再生進捗度のコールバック。

onMusicPrepareToPlay	音楽再生準備のコールバック。
onMusicCompletePlaying	音楽再生完了のコールバック。

SDK基本関数

sharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを取得します。



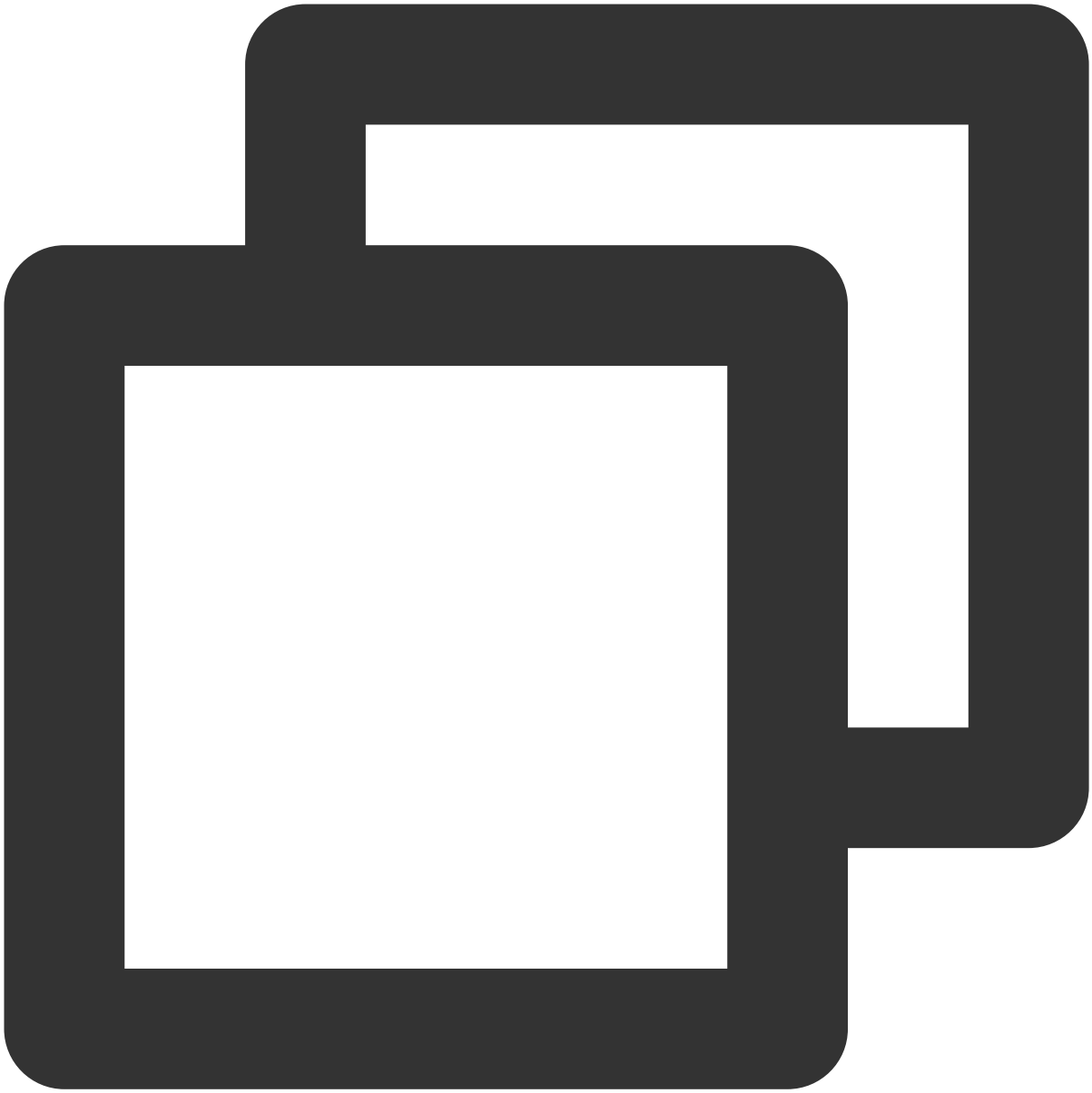
```
/**
* TRTCKaraokeRoomシングルトンオブジェクトの取得
*
* - returns: TRTCKaraokeRoomインスタンス
* - note: {@link TRTCKaraokeRoom#destroySharedInstance()}を呼び出してシングルトンオブジェ
*/
+ (instancetype)sharedInstance NS_SWIFT_NAME(shared());
```

destroySharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを破棄します。

説明：

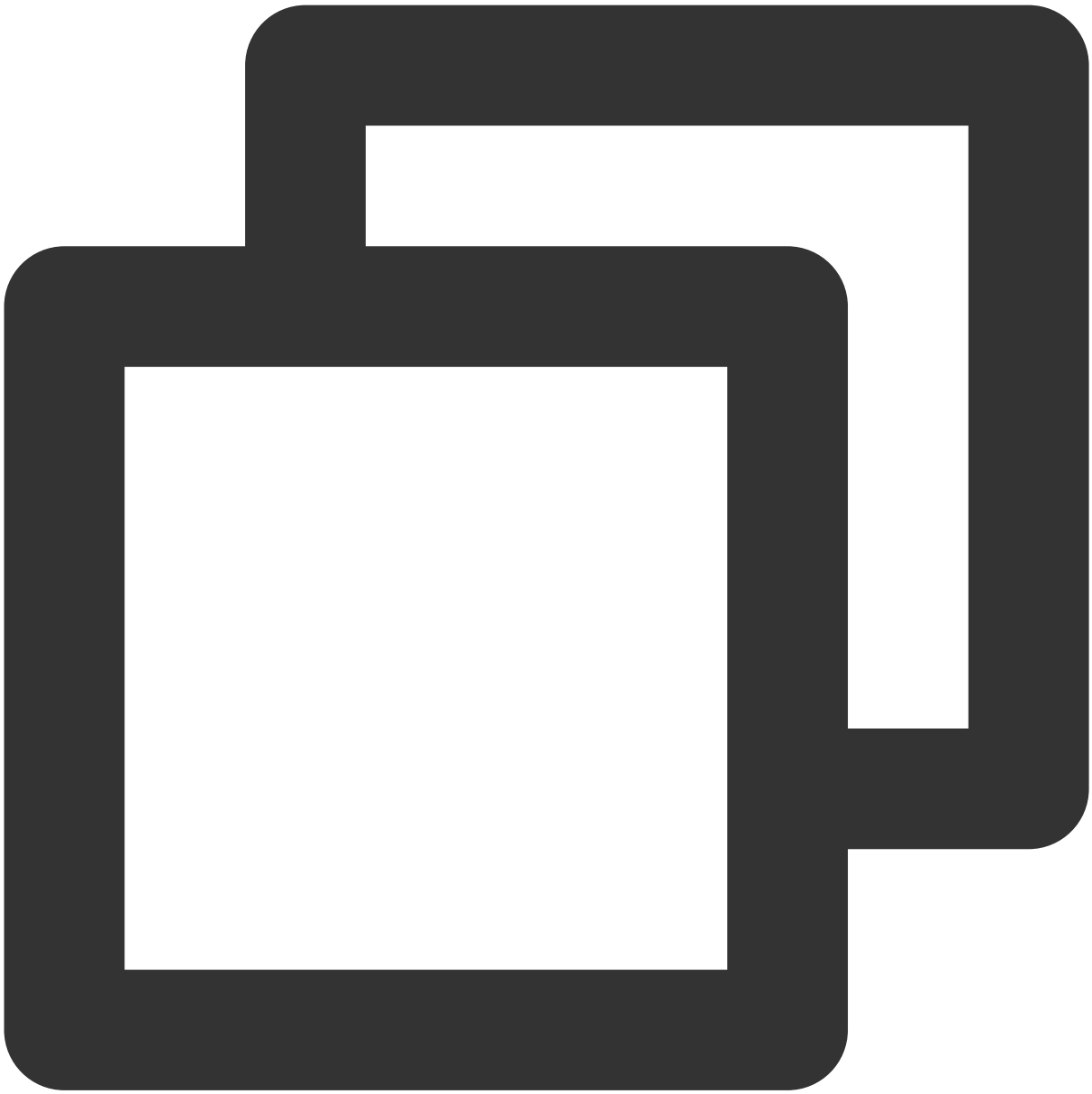
インスタンスの破棄後は、外部にキャッシュされたTRTCKaraokeRoomインスタンスの再使用ができません。改めてsharedInstanceを呼び出し、インスタンスを新規取得する必要があります。



```
/**
* TRTCKaraokeRoomシングルトンオブジェクトの破棄
*
* - note:インスタンスの破棄後は、外部にキャッシュされたTRTCKaraokeRoomインスタンスは再使用ができ
*/
+ (void)destroySharedInstance NS_SWIFT_NAME(destroyShared());
```

setDelegate

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



```
/**
* コンポーネントコールバックインターフェースの設定
*
* TRTCKaraokeRoomDelegateによってTRTCKaraokeRoomの各種ステータス通知を取得することができます
*
* - parameter delegateコールバックインターフェース
* - note: TRTCKaraokeRoomのコールバックイベントです。デフォルトではMain Queueでコールバックさ
*/
```

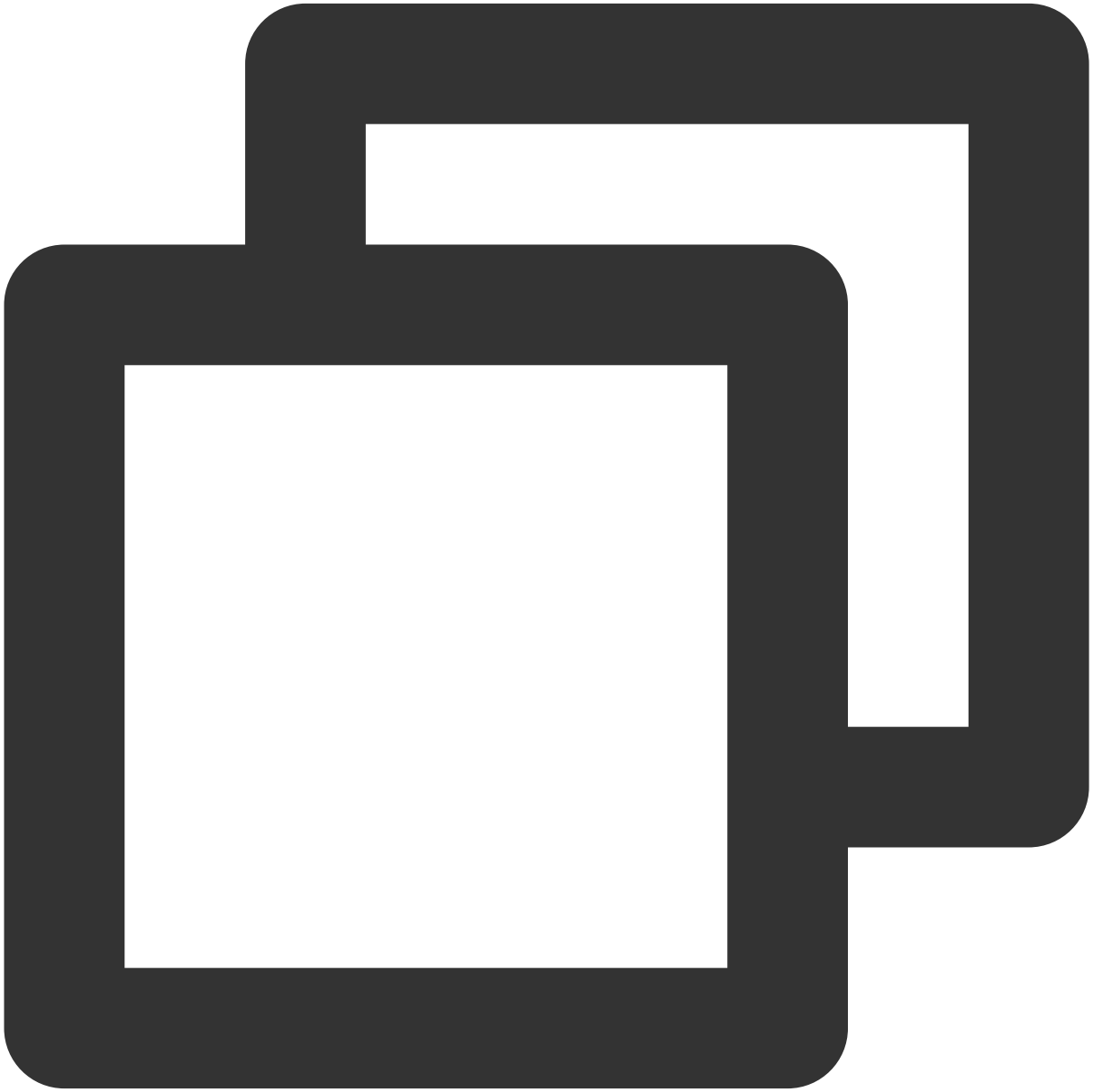
```
- (void)setDelegate:(id<TRTCKaraokeRoomDelegate>)delegate NS_SWIFT_NAME(setDelegate
```

説明：

setDelegateはTRTCKaraokeRoomのプロキシコールバックです。

setDelegateQueue

イベントコールバックが所在するスレッドキューを設定し、デフォルトはメインスレッド MainQueue に送信します。



```
/**
```

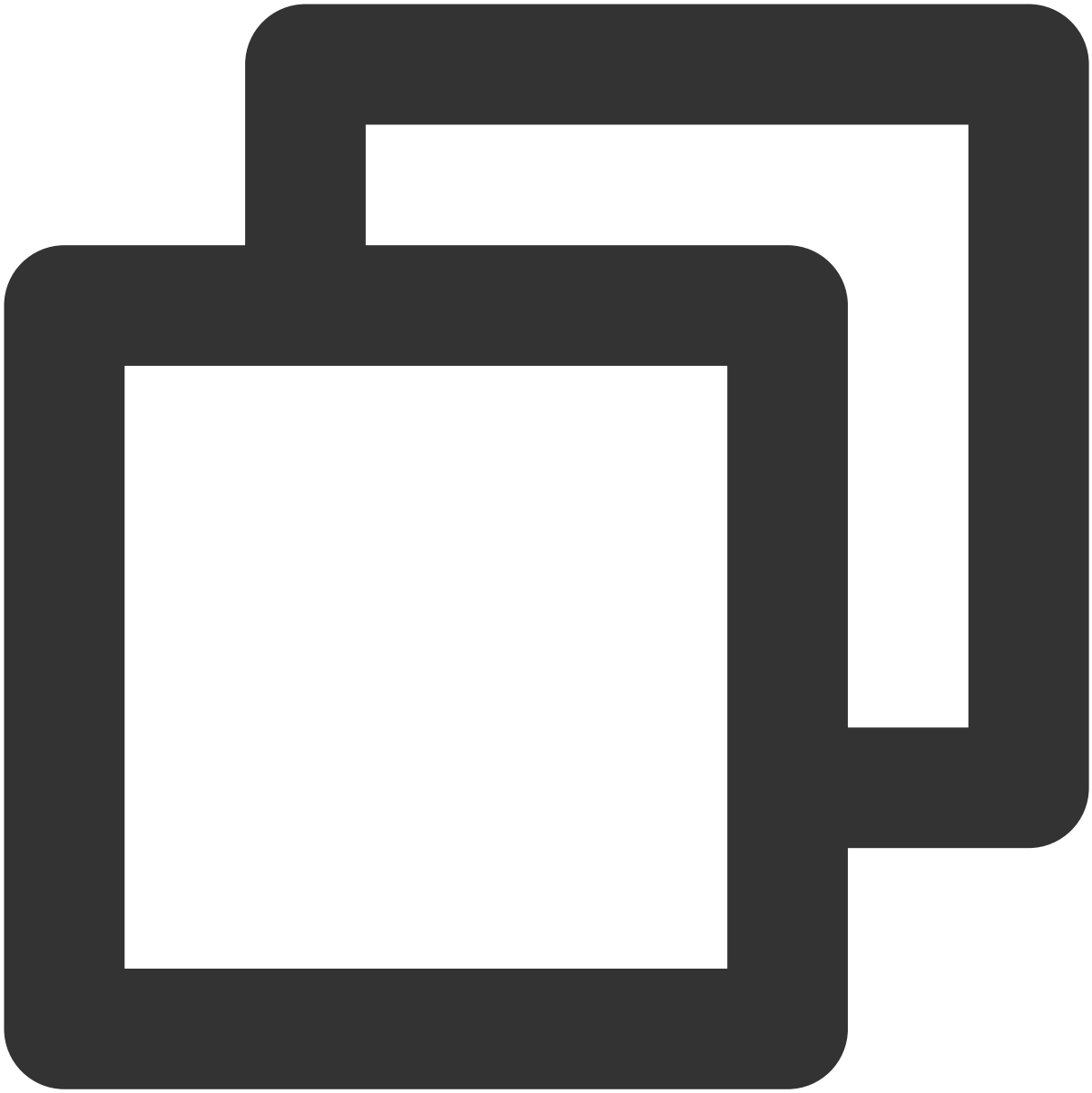
```
 * イベントコールバックが配置されているキューの設定
 *
 * - parameter queueキューです。TRTCKaraokeRoomの各種ステータス通知コールバックは、指定したque
 */
- (void)setDelegateQueue:(dispatch_queue_t)queue NS_SWIFT_NAME(setDelegateQueue(que
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
queue	dispatch_queue_t	TRTCKaraokeRoomの各種ステータス通知は、指定したスレッドキューに送信されます。

login

ログイン。



```
- (void)login:(int)sdkAppID
       userId:(NSString *)userId
      userSig:(NSString *)userSig
     callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(login(sdkAppID:userI
```

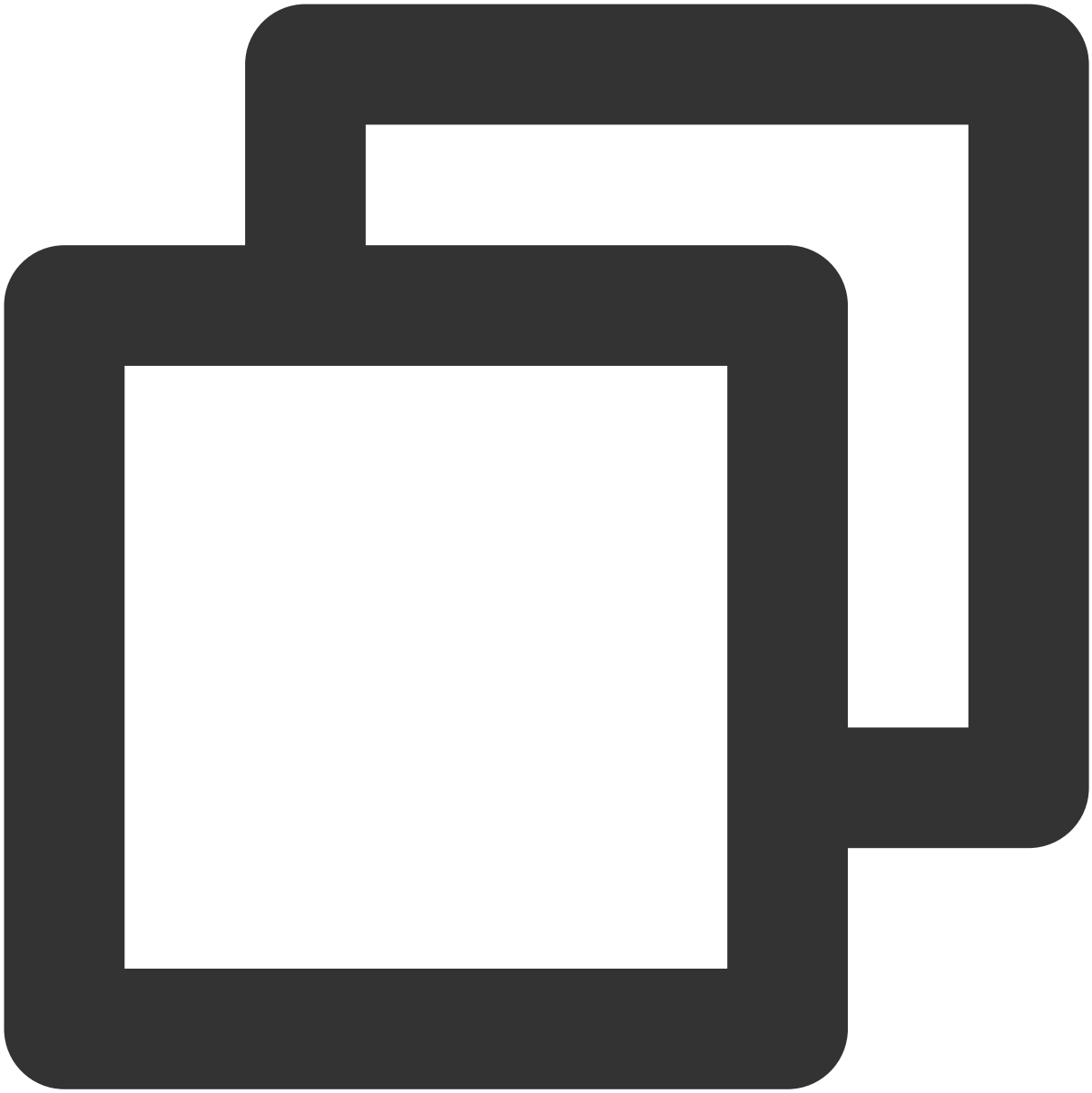
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

sdkAppId	int	TRTCコンソール> 【アプリケーション管理】>アプリケーション情報の中でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプであり、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、 UserSigの計算、使用方法をご参照ください。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

logout

ログアウト。



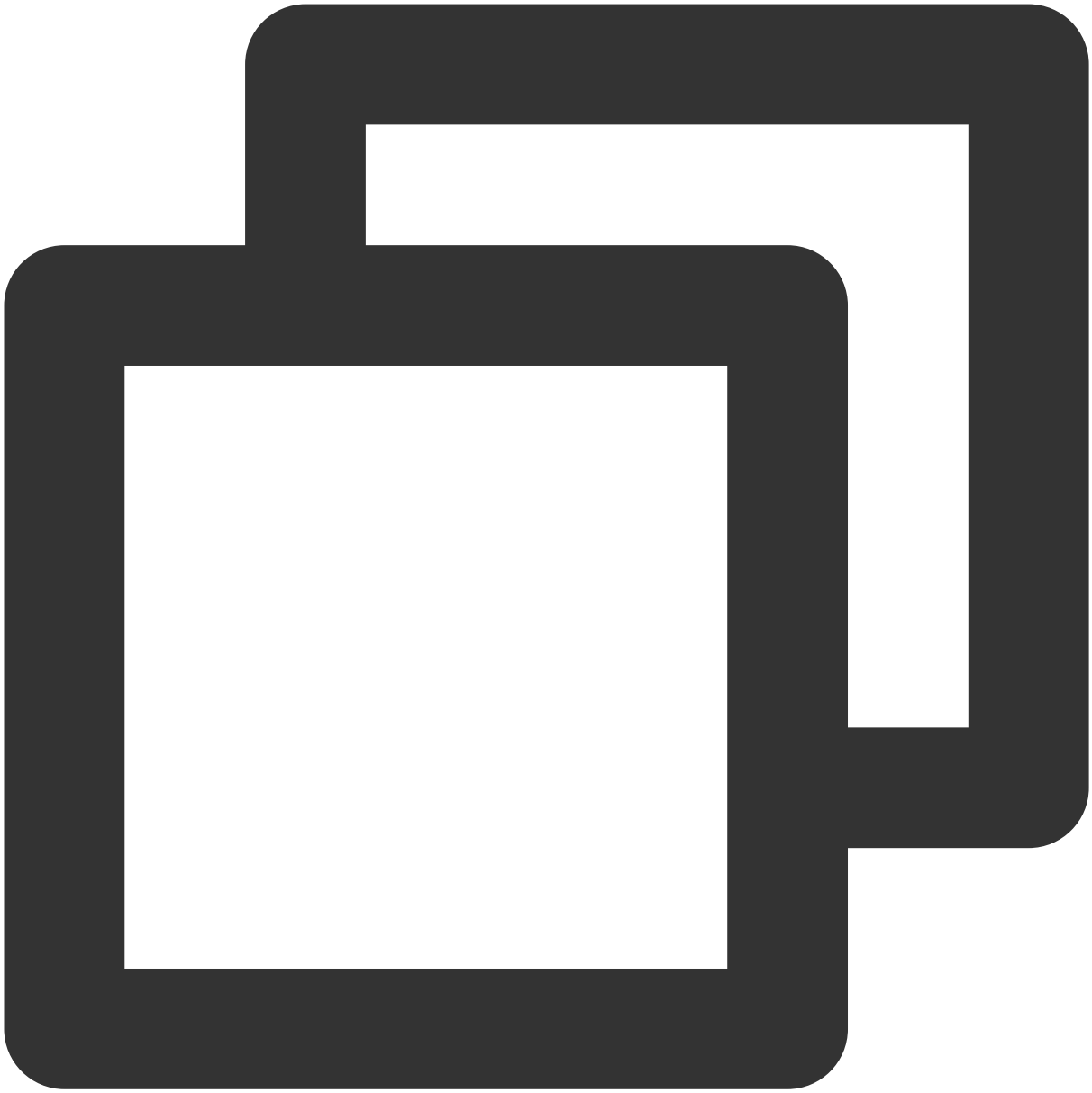
```
- (void)logout:(ActionCallback _Nullable)callback NS_SWIFT_NAME(logout(callback:));
```

パラメータは下表に示すとおりです：

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

setSelfProfile

個人情報の修正。



```
- (void)setSelfProfile:(NSString *)userName avatarURL:(NSString *)avatarURL callbac
```

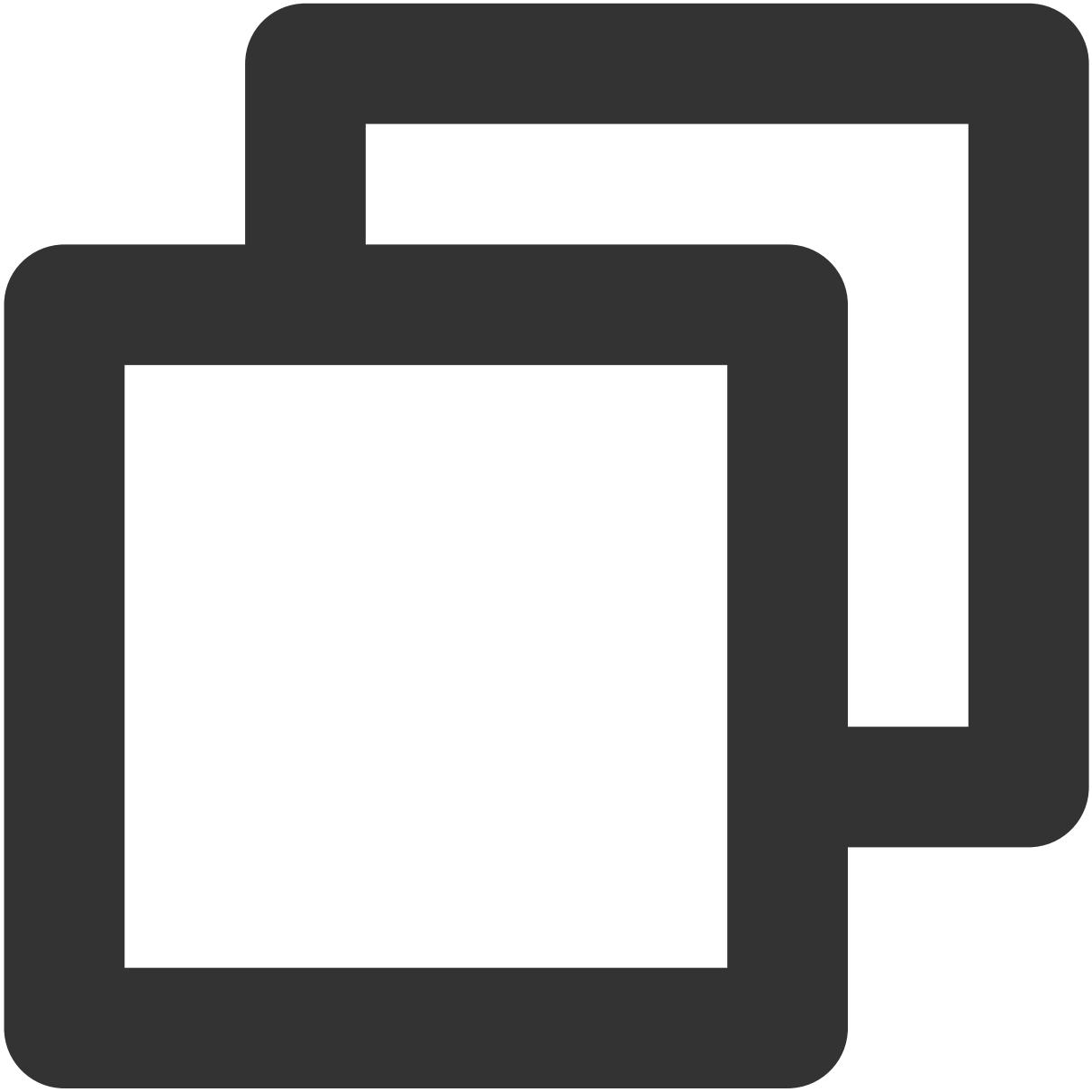
パラメータは下表に示すとおりです：

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

ルーム関連インターフェース関数

createRoom

ルームの作成（管理者が呼び出し）。



```
- (void)createRoom:(int)roomID roomParam:(RoomParam *)roomParam callback:(ActionCal
```

パラメータは下表に示すとおりです：

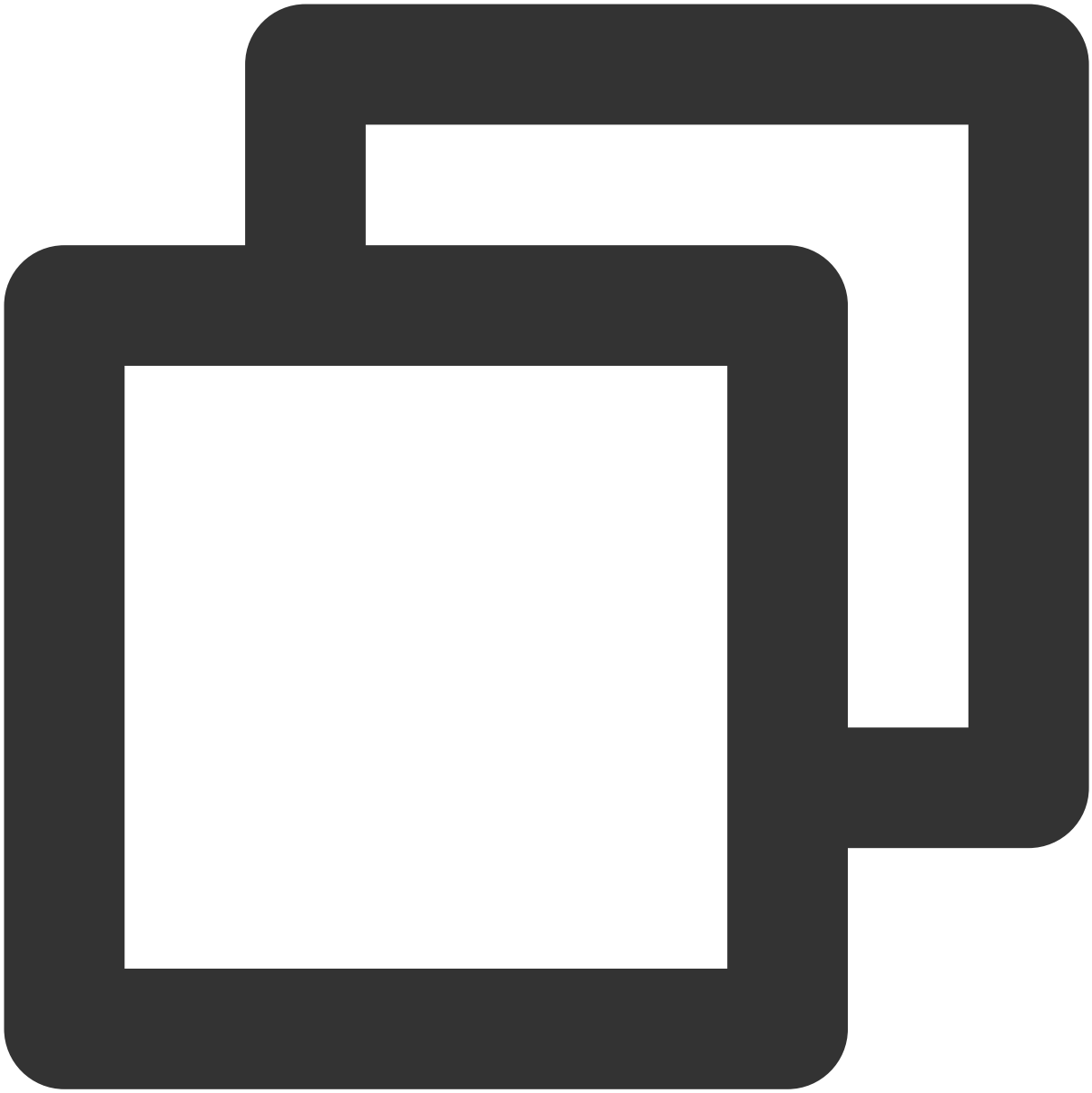
パラメータ	タイプ	意味

roomId	int	ルームIDです。ご自身でアサインし、一元管理する必要があります。複数のroomIDを、1つのボイスチャットルームリストにまとめることができます。Tencent Cloudでは現在、ボイスチャットルームリストの管理サービスを行っていませんので、ご自身でボイスチャットルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報です。ルーム名、マイク情報、カバー情報など、ルームを説明するために用いる情報に使用します。マイク管理が必要な場合は、ルームのマイク数を記入する必要があります。
callback	ActionCallback	ルームの新規作成結果のコールバック。成功時にcodeは0になります。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです：
1. 管理者は、 `createRoom` を呼び出して新しいKaraokeルームを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否、ルームタイプなどルームの属性情報を渡します。
2. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
3. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリスト変更をUI上に更新することができます。
4. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。



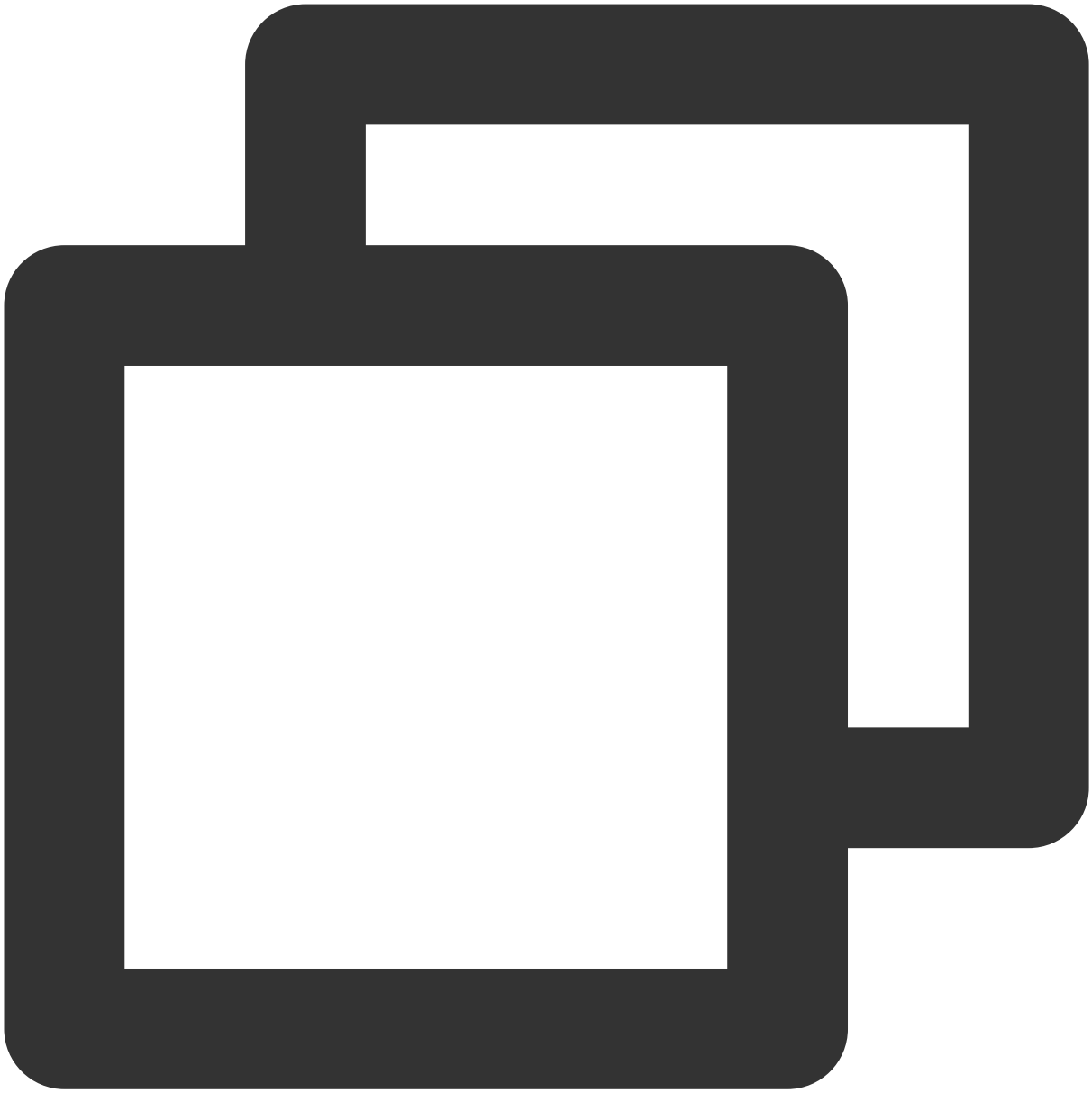
```
- (void)destroyRoom:(ActionCallback _Nullable)callback NS_SWIFT_NAME(destroyRoom(ca
```

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
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

enterRoom

入室（リスナーが呼び出し）。



```
- (void)enterRoom:(NSInteger)roomID callback:(ActionCallback _Nullable)callback NS_
```

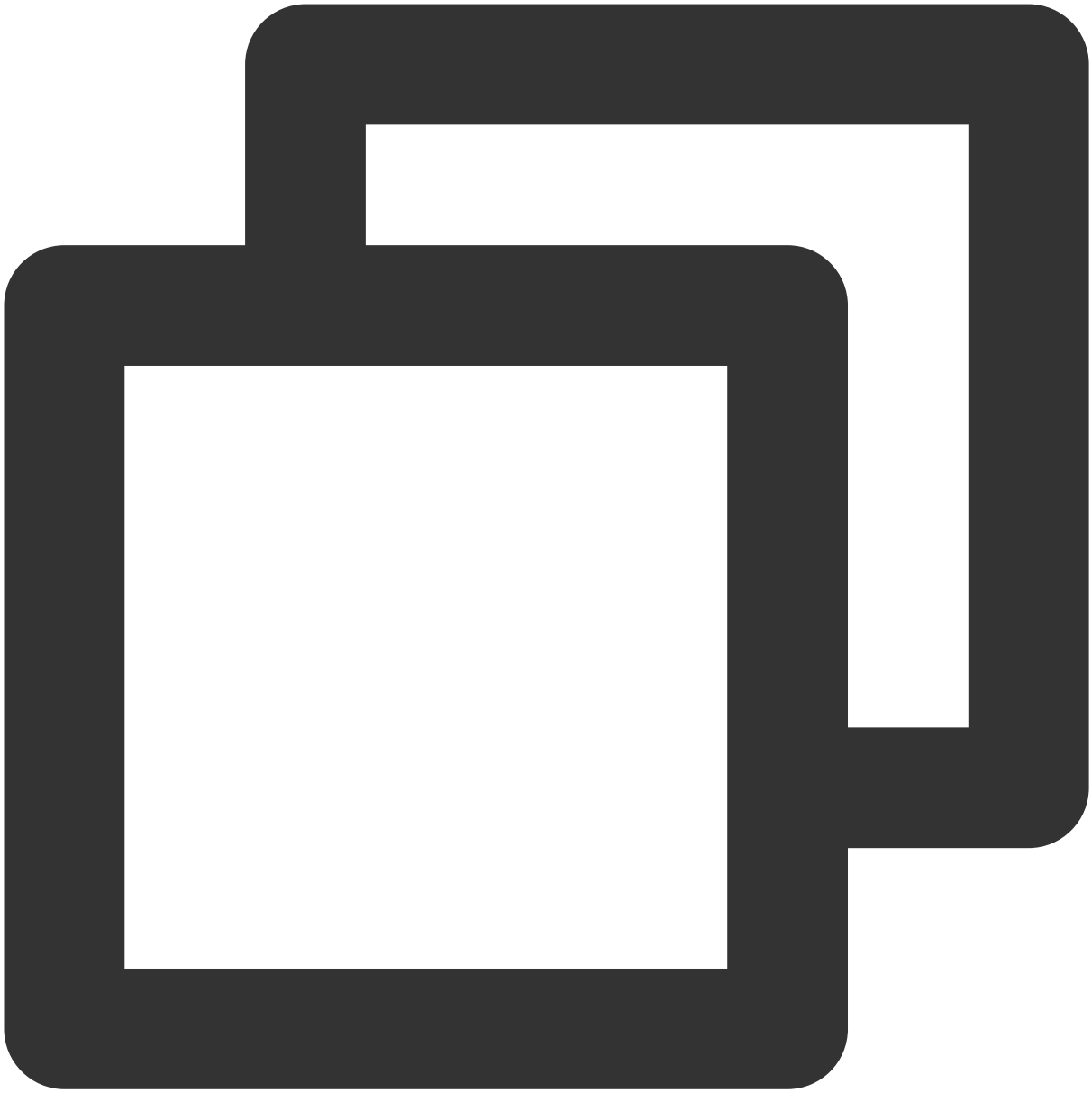
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
roomId	int	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

リスナーが入室し聴取する際の通常の呼び出しプロセスは次のとおりです：
1. リスナーがサーバーから最新のKaraokeルームリストを取得します。その中には、多くのボイスチャットルームのroomIdおよびルーム情報が含まれる場合があります。
2. リスナーは1つのKaraokeルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
3. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、発言者にする際の管理者への同意リクエストの要否の記録など。
4. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時にマイクリストの変更をUI上に更新することができます。
5. 入室後にマイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。

exitRoom

ルームから退出します。



```
- (void)exitRoom:(ActionCallback _Nullable)callback NS_SWIFT_NAME(exitRoom(callback
```

パラメータは下表に示すとおりです：

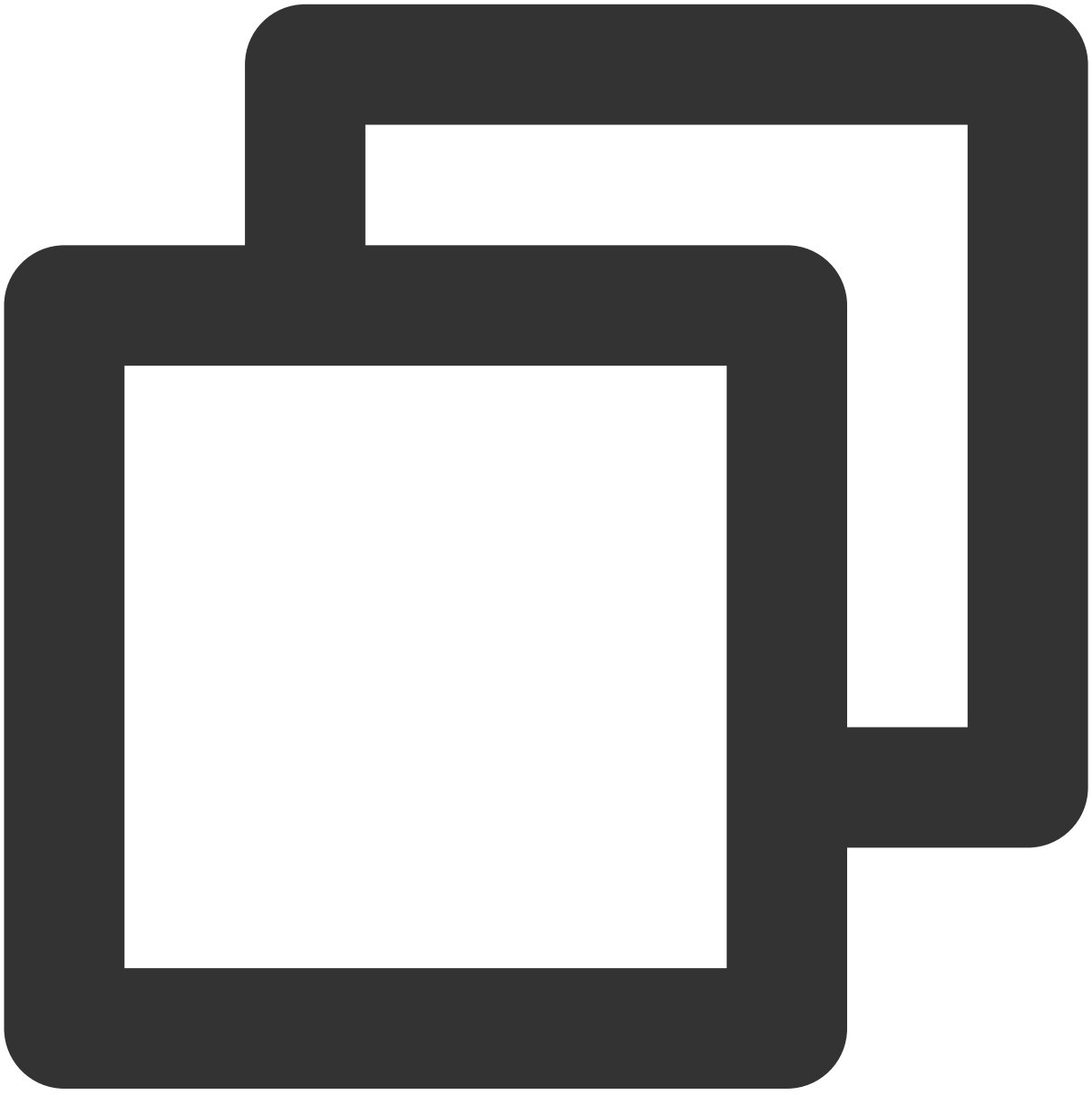
パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

説明：

ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視できます。



```
- (void)getRoomInfoList:(NSArray<NSNumber *> *)roomIdList callback:(KaraokeInfoCall
```

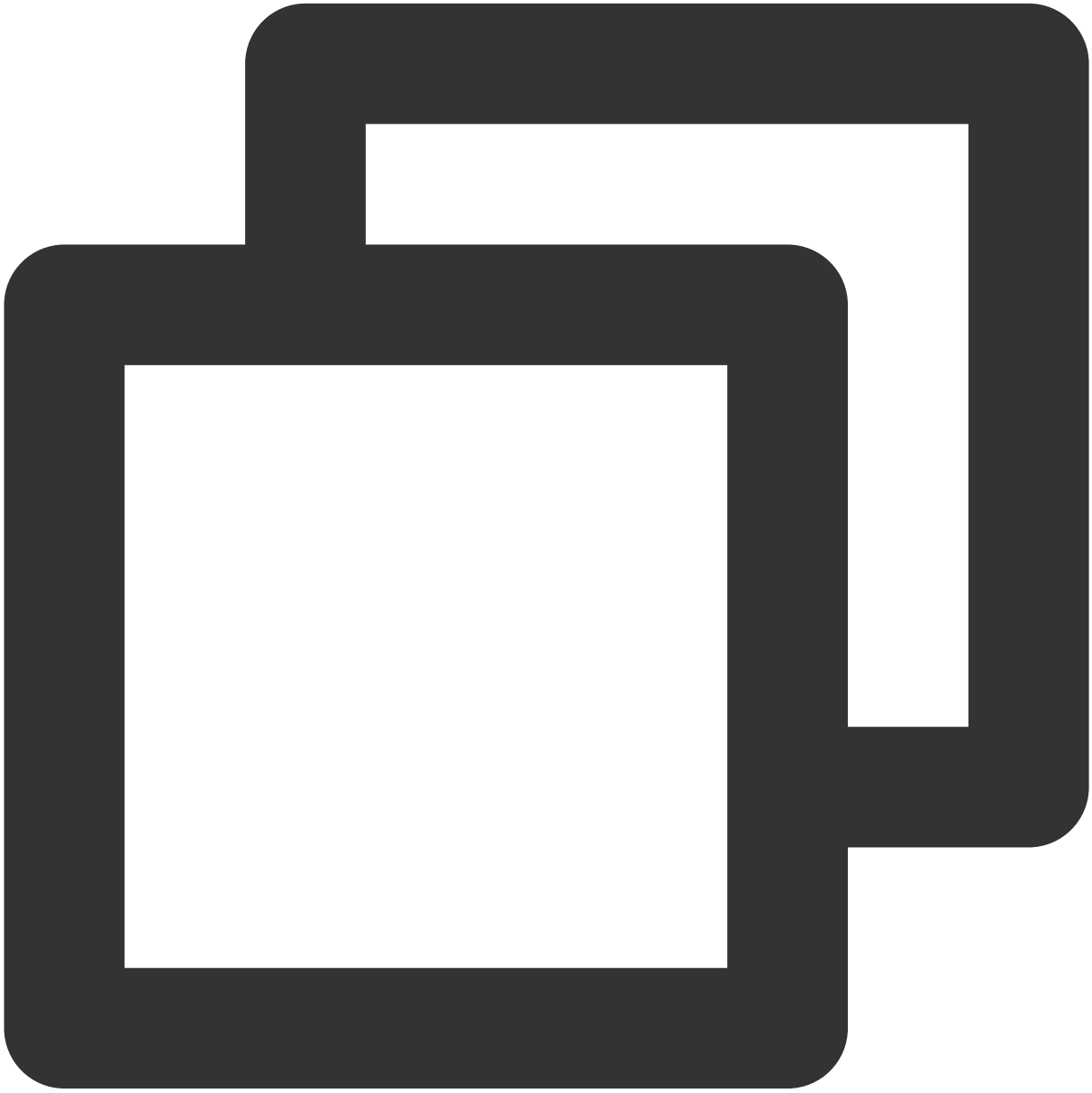
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

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

getUserInfoList

指定されたuserIdのユーザー情報を取得します。



```
- (void)getUserInfoList:(NSArray<NSString *> * _Nullable)userIDList callback:(Karao
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userIdList	List<String>	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。
userlistcallback	UserListCallback	ユーザーの詳細情報のコールバック。

音楽再生インターフェース

startPlayMusic

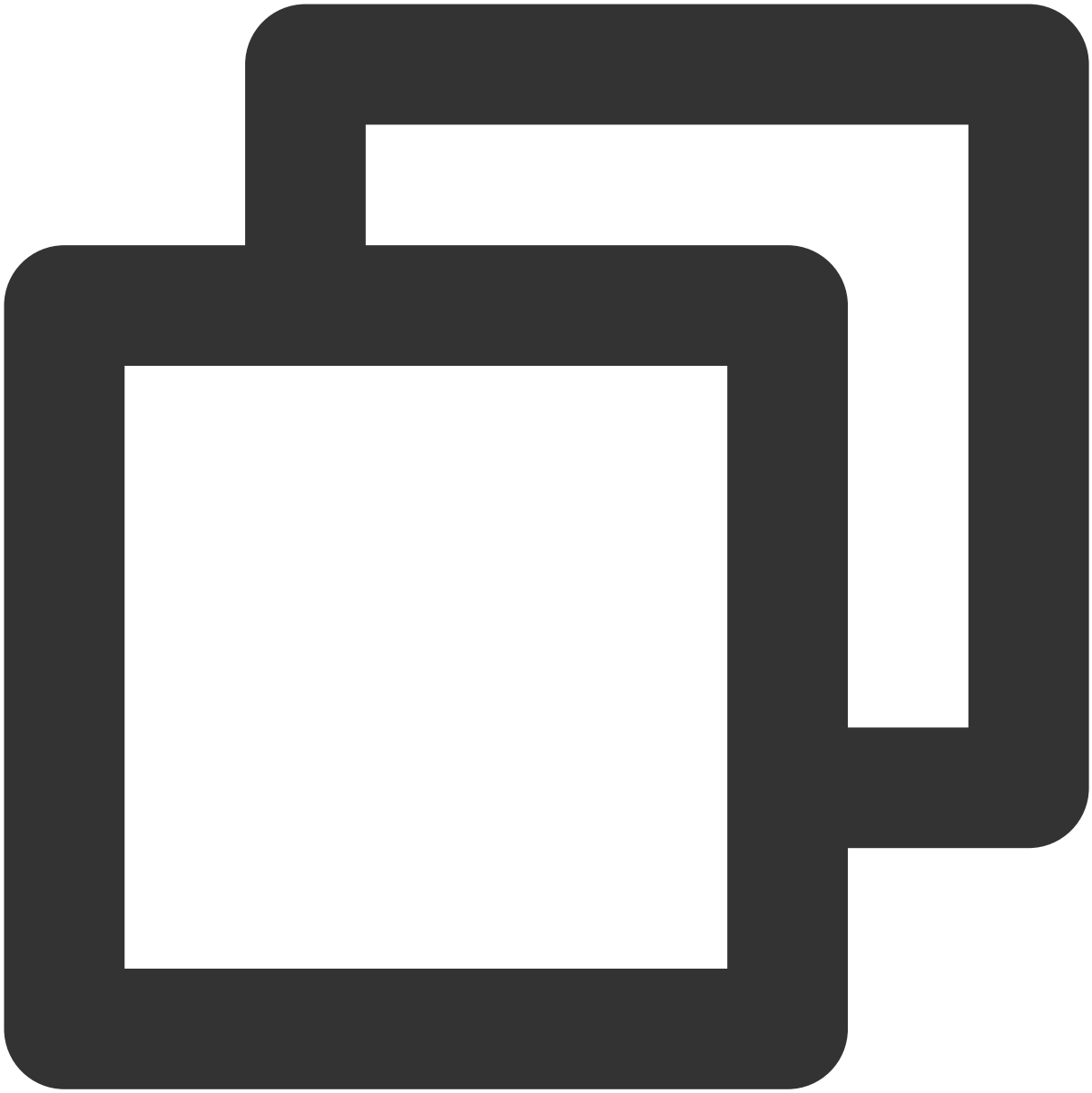
音楽を再生します（マイク・オン後に呼び出し）。

説明：

音楽を再生すると、 `onMusicPrepareToPlay` というイベント通知を受信します。

音楽の再生中、ルーム内の全メンバーは、 `onMusicProgressUpdate` というイベント通知を継続して受け取ります。

音楽の再生が完了すると、 `onMusicCompletePlaying` というイベント通知を受信します。



```
- (void)startPlayMusic:(int32_t)musicID originalUrl:(NSString *)originalUrl accompa
```

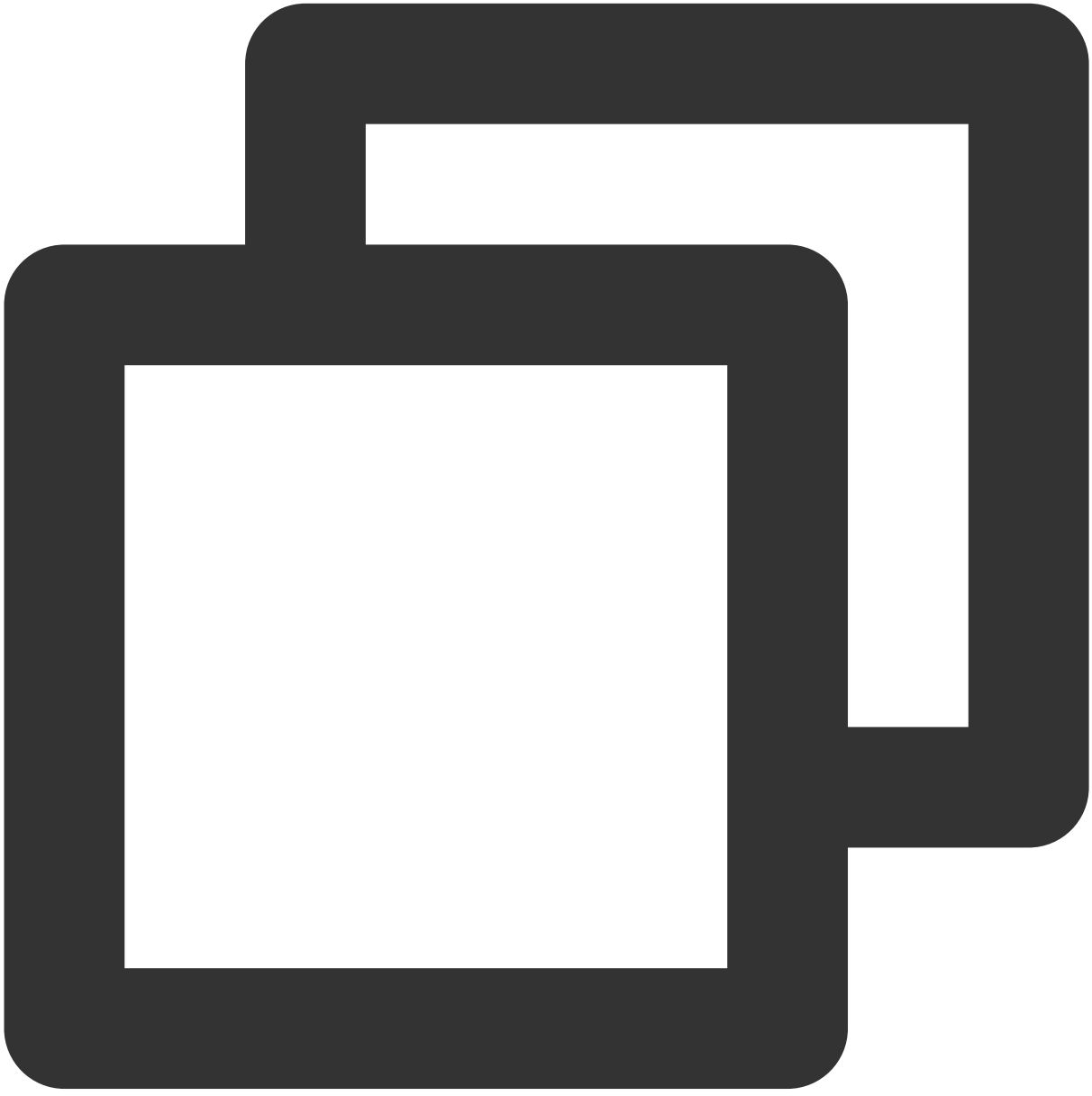
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int32_t	音楽のID。
originalUrl	String	原曲の絶対パス。
accompanyUrl	String	伴奏の絶対パス。

このインターフェースを呼び出すと、最後に再生されていた楽曲が停止します。

stopPlayMusic

音楽の再生を停止します（音楽を再生するときに呼び出します）。
説明：
再生が停止すると、 `onMusicCompletePlaying` というイベント通知を受信します。



```
- (void)stopPlayMusic NS_SWIFT_NAME(stopPlayMusic());
```

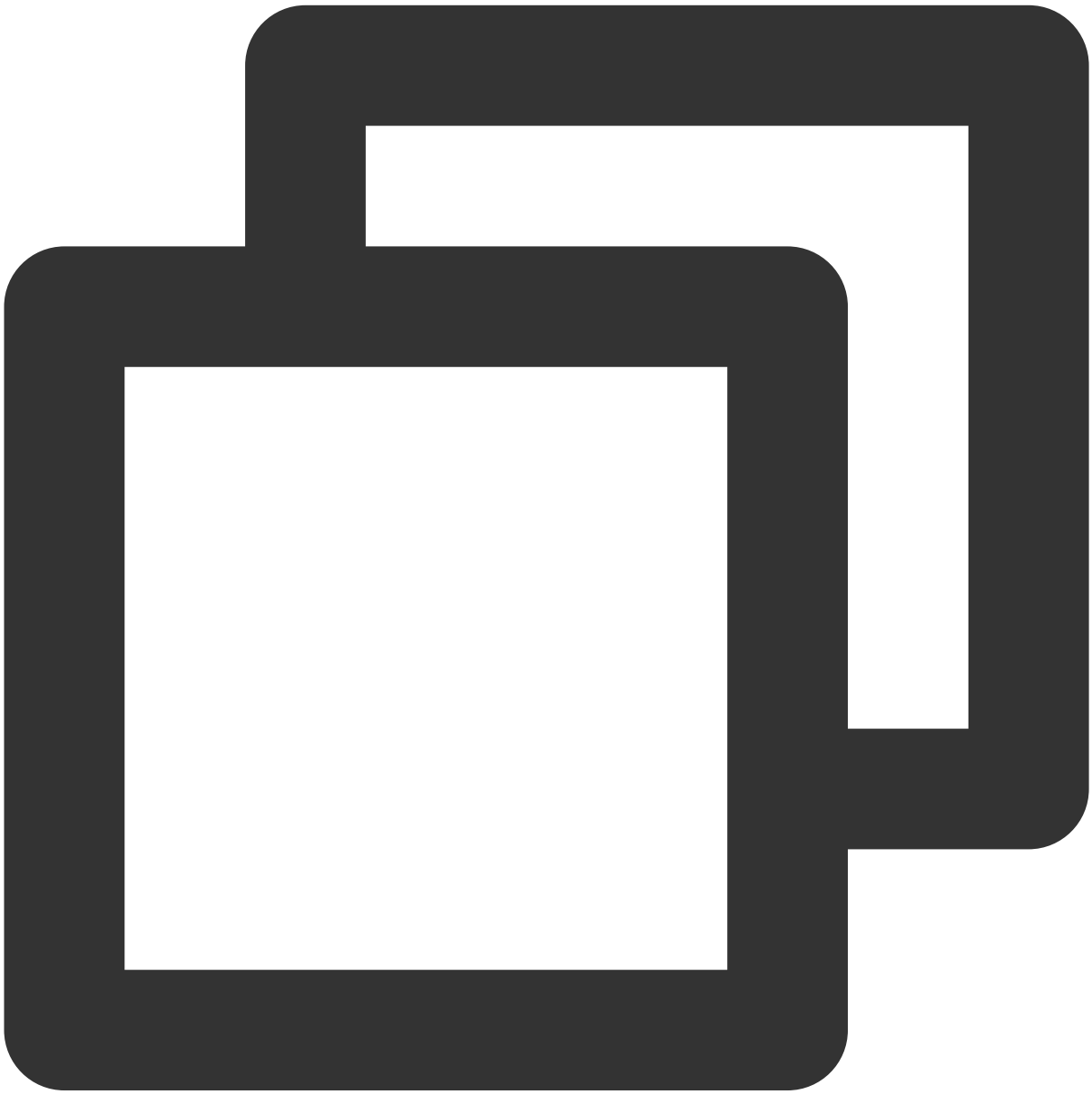
pausePlayMusic

音楽の再生を一時停止します（音楽を再生するときに呼び出します）。

説明：

`onMusicProgressUpdate` というイベント通知が一時停止されます

`onMusicCompletePlaying` というイベント通知を受信しません。



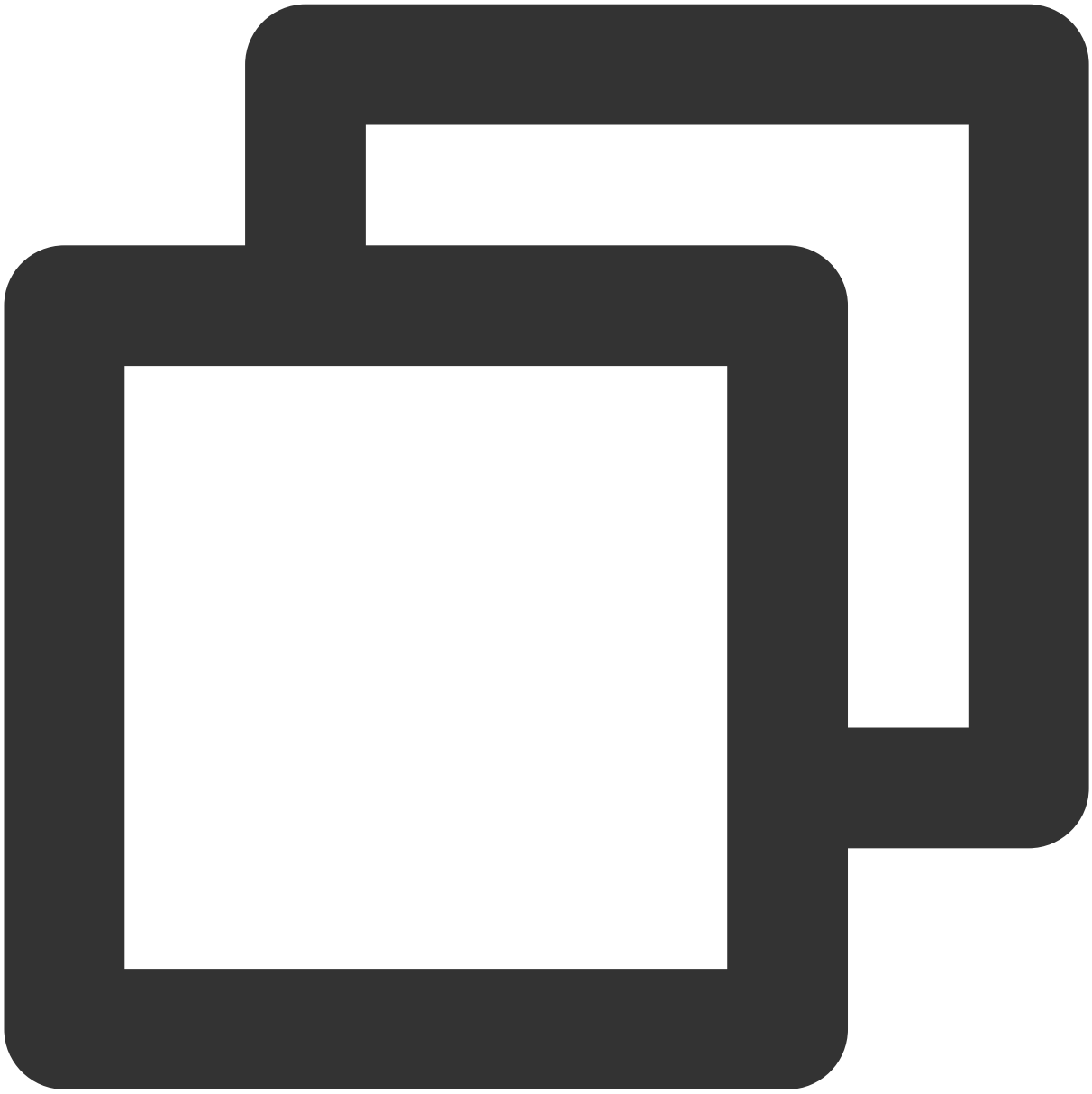
```
- (void)pausePlayMusic NS_SWIFT_NAME(pausePlayMusic());
```

resumePlayMusic

一時停止した音楽を再開します（一時停止後に呼び出します）。

説明：

`onMusicPrepareToPlay` というイベント通知を受信しません。



```
- (void)resumePlayMusic NS_SWIFT_NAME(resumePlayMusic());
```

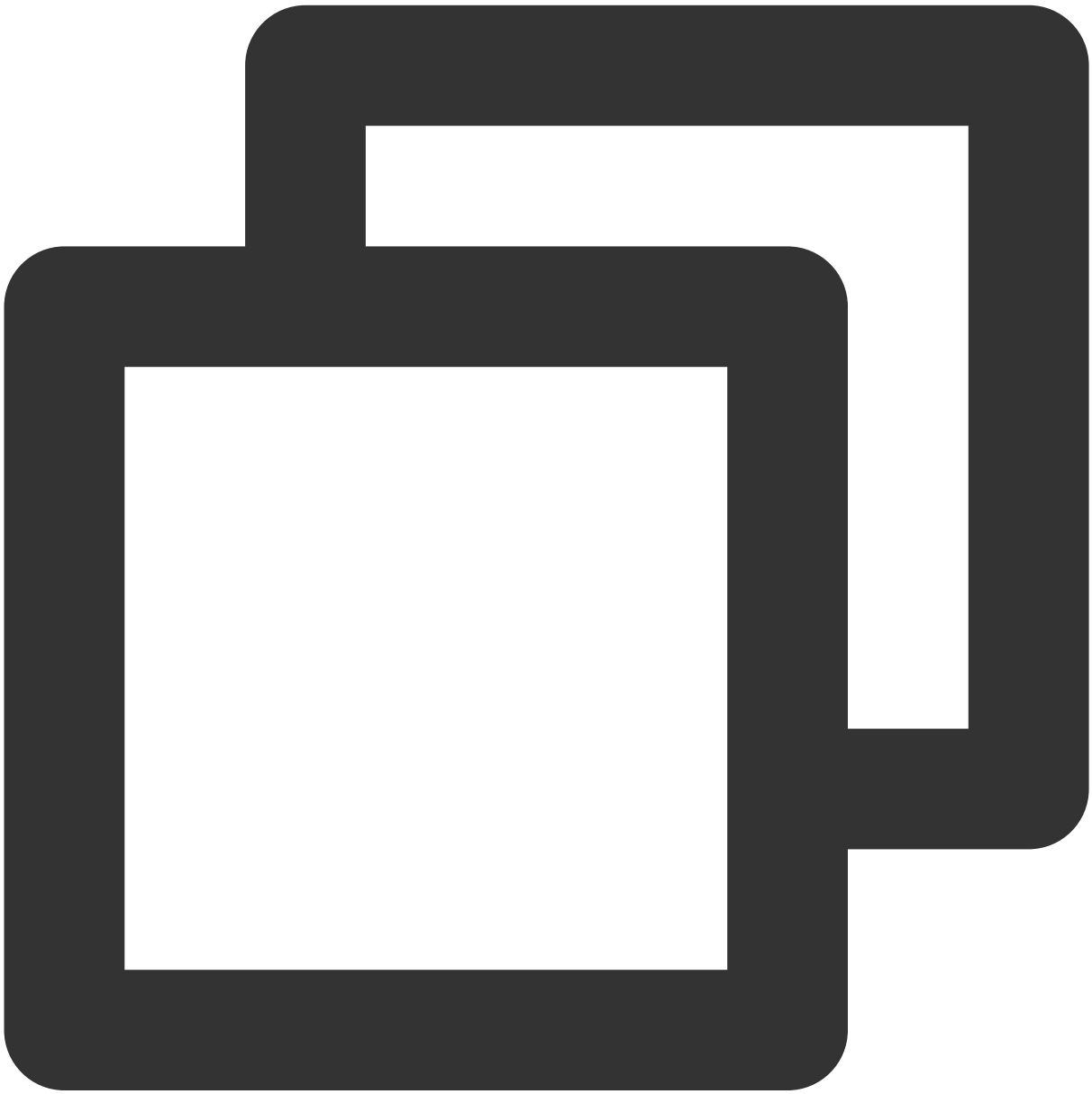
マイク管理インターフェース

enterSeat

ユーザーが発言者になります（リスナー側/管理者ともに呼び出し可）。

説明：

マイク・オンの成功後、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。



```
- (void)enterSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callback
```

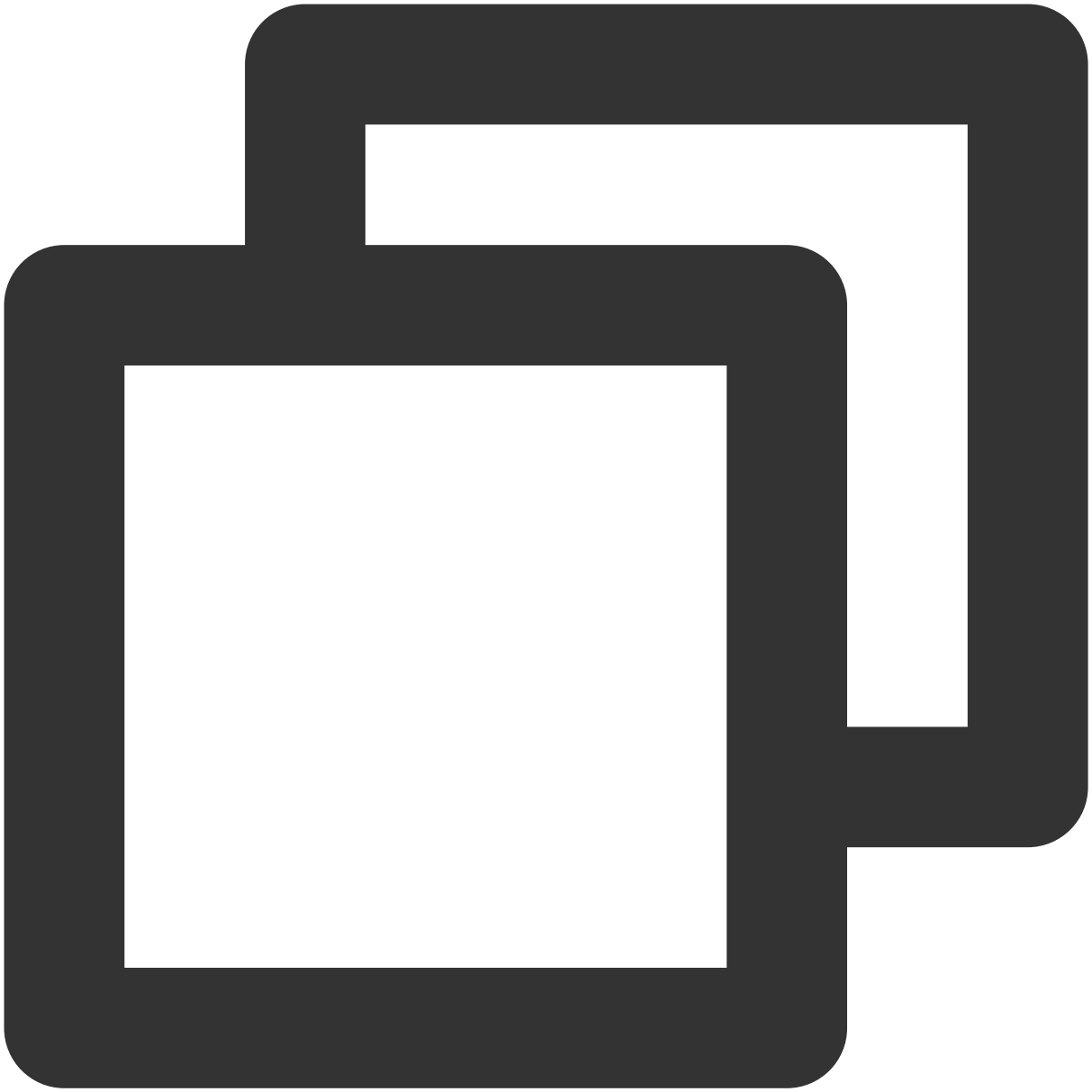
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	マイク・オンの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、`onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

leaveSeat

ユーザーが視聴者になります（キャスターが呼び出し）。
説明：
マイク・オフの成功後、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。



```
- (void)leaveSeat:(ActionCallback _Nullable)callback NS_SWIFT_NAME(leaveSeat(callba
```

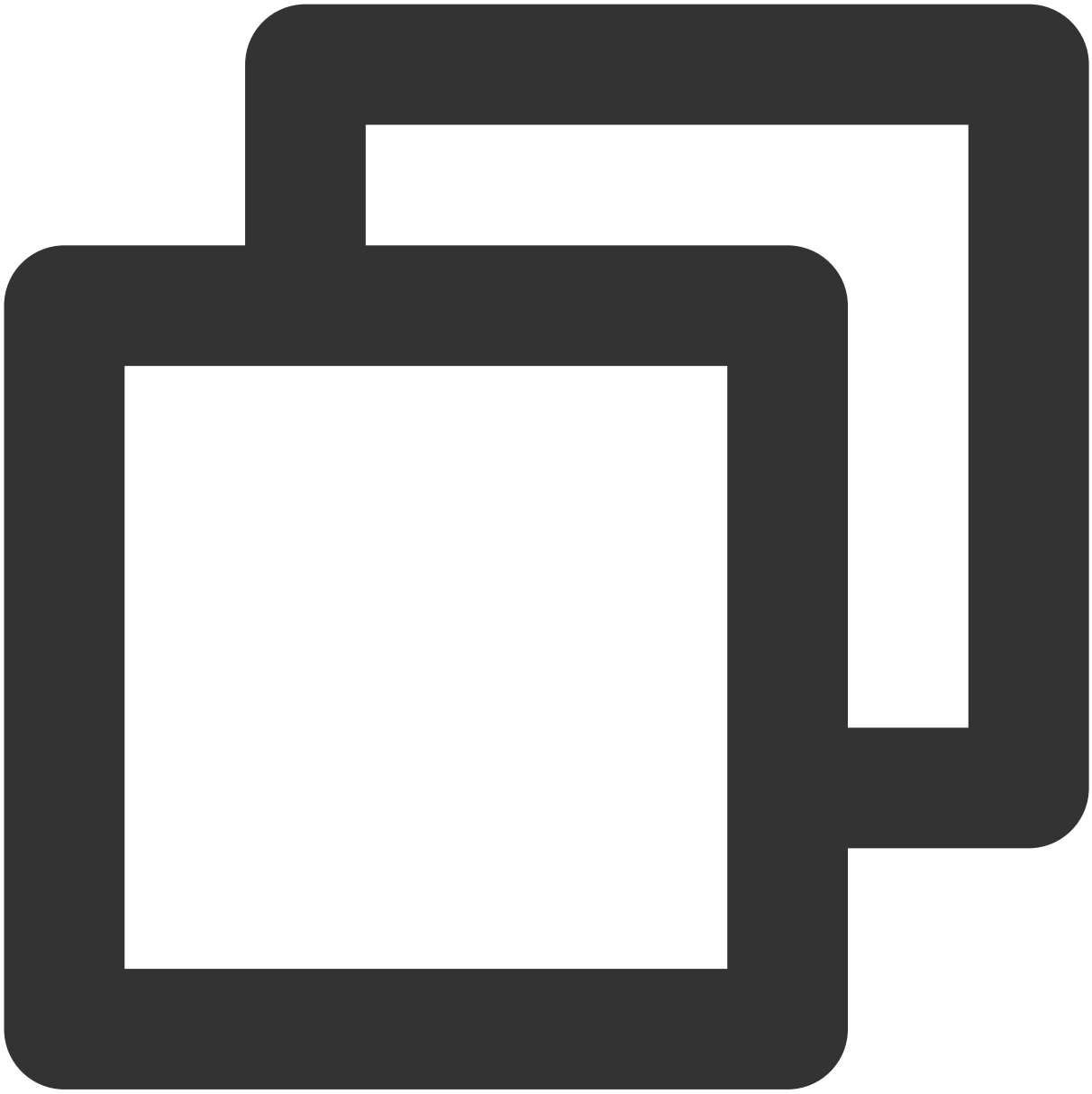
パラメータは下表に示すとおりです：

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

pickSeat

視聴者が発言できるように招待（管理者が呼び出し）。

説明：
管理者が視聴者を発言できるように招待すると、ルーム内の全メンバーは、`onSeatListChange` と `onAnchorEnterSeat` というイベント通知を受信します。



```
- (void)pickSeat:(NSInteger)seatIndex userId:(NSString *)userId callback:(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
seatIndex	int	視聴者が発言できるように招待する必要があるマイク番号。

userId	String	ユーザーID。
callback	ActionCallback	操作コールバック。

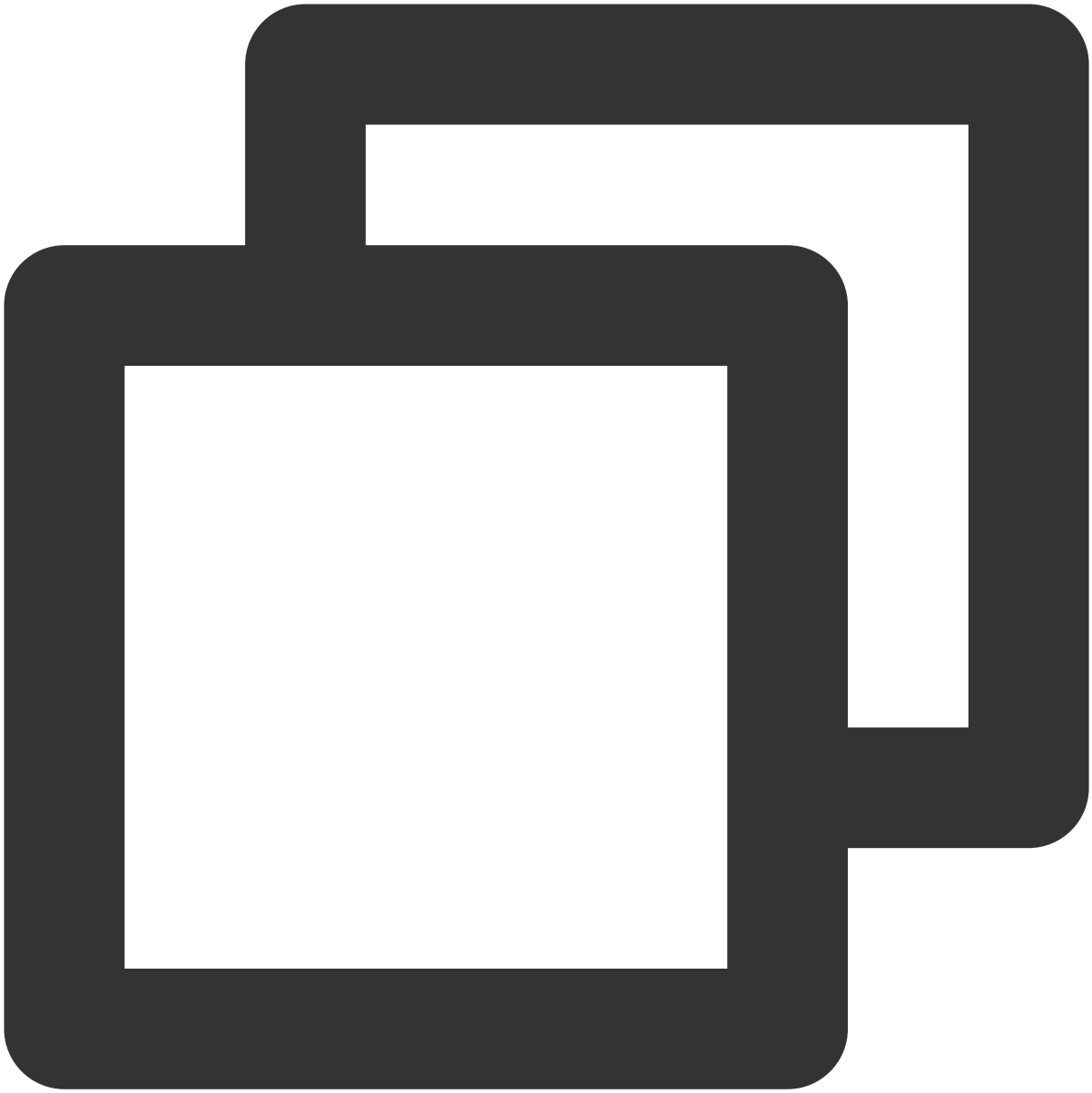
このインターフェースを呼び出すと、すぐにマイクリストが修正されます。管理者がリスナーの同意がなければマイク・オンできないケースの場合は、まず `sendInvitation` を呼び出してからリスナーに申請し、`onInvitationAccept` を受信すると、この関数を呼び出せるようになります。

kickSeat

キックアウトしてマイク・オフ（管理者が呼び出し）。

説明：

管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。



```
- (void)kickSeat:(NSInteger)seatIndex callback:(ActionCallback _Nullable)callback N
```

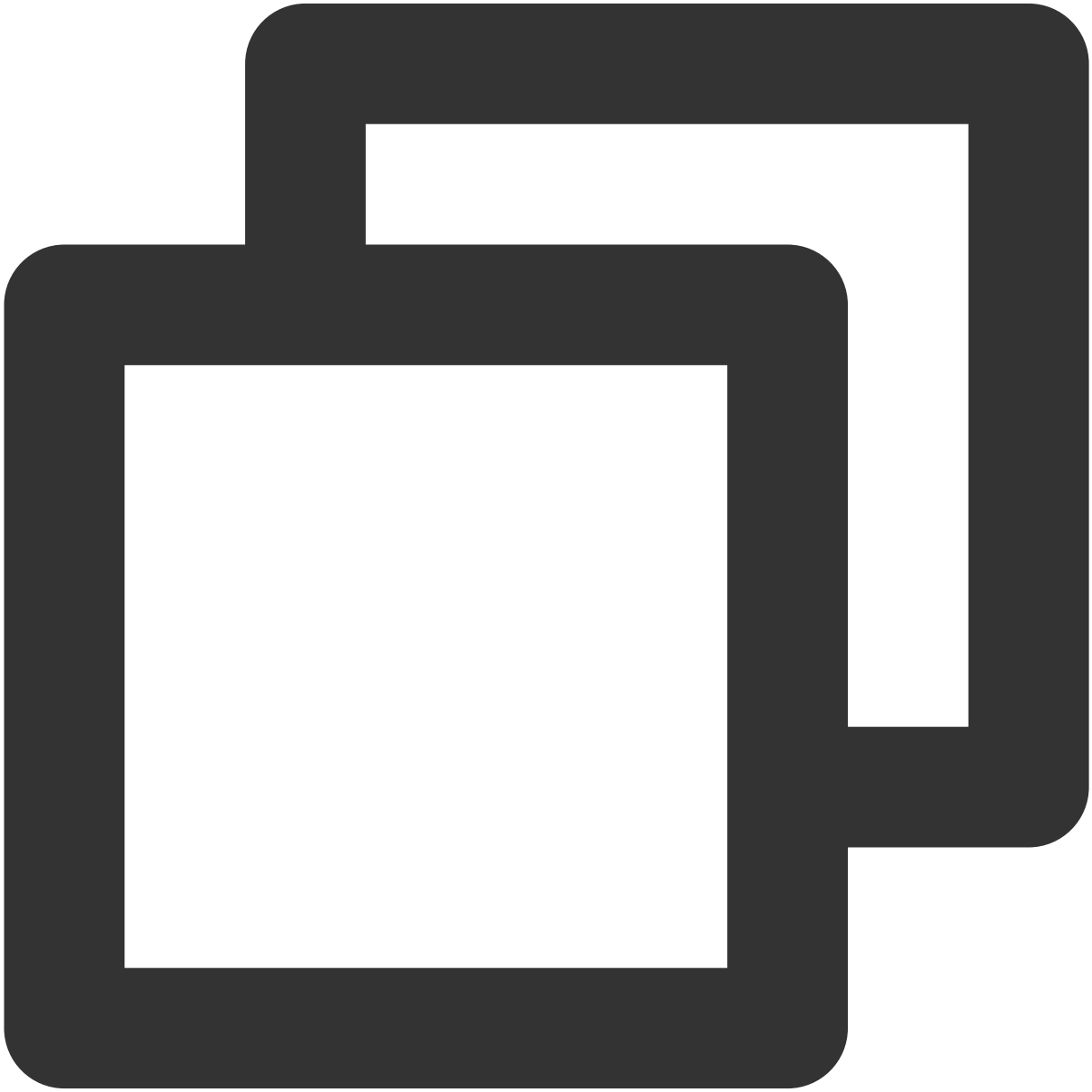
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	キックアウトしてマイク・オフの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

muteSeat

任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
説明：
任意のマイクをミュート/ミュート解除します。ルーム内の全メンバーが `onSeatListChange` および `onSeatMute` というイベント通知を受信します。



```
- (void)muteSeat:(NSInteger)seatIndex isMute:(BOOL)isMute callback:(ActionCallback
```

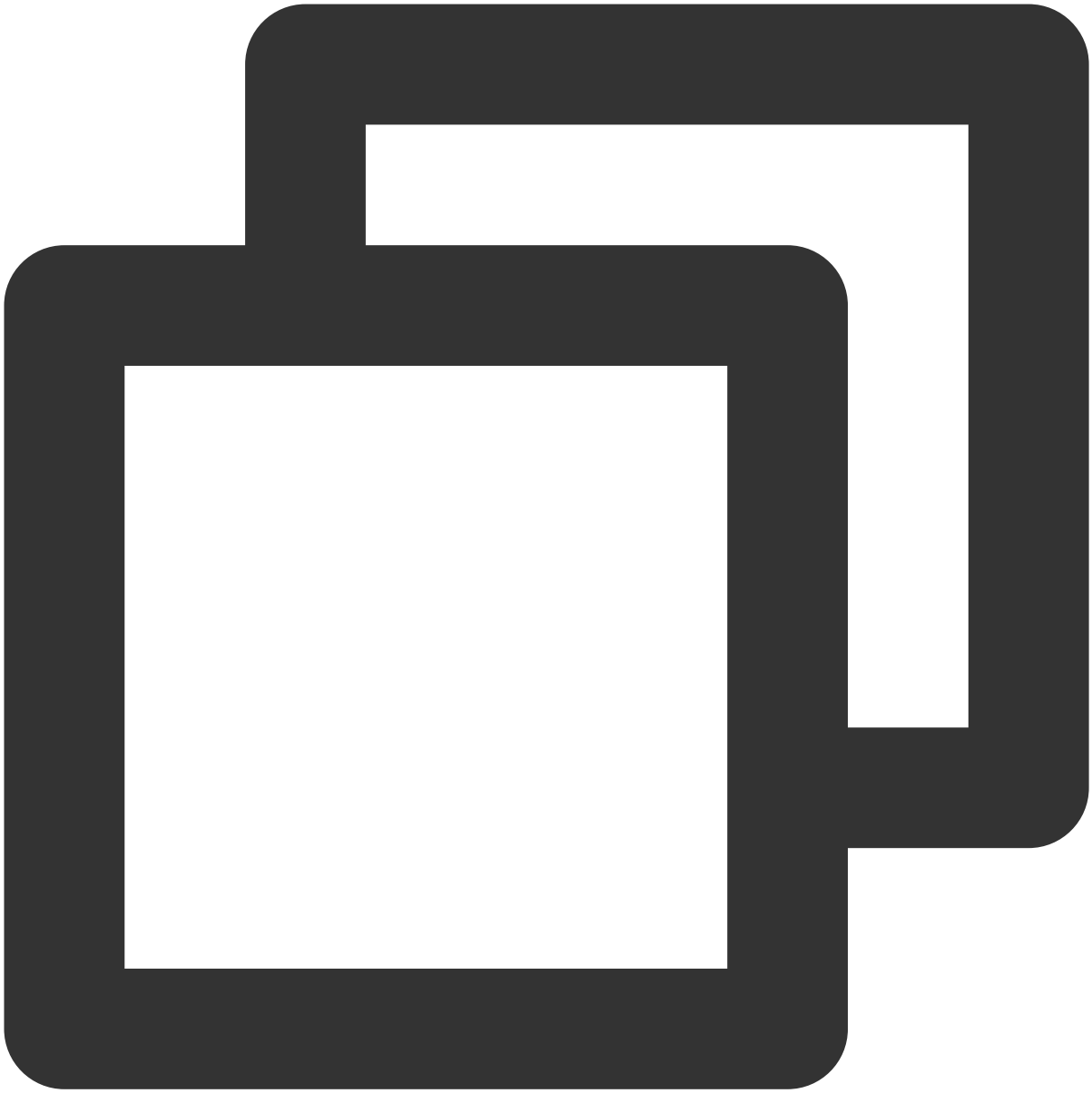
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isMute	boolean	true：該当するマイクをミュートします；false：該当するマイクをミュート解除します。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。seatIndexの座席に該当するキャスターは、muteAudioを自動的に呼び出してミュート/ミュート解除にします。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。
説明：
管理者は、該当するマイクをクローズ/解除し、ルーム内の全メンバーは `onSeatListChange` および `onSeatClose` というイベント通知を受信します。



```
- (void)closeSeat:(NSInteger)seatIndex isClose:(BOOL)isClose callback:(ActionCallba
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isClose	boolean	true：該当するマイクをクローズします； false：該当するマイクをクローズ解除します。

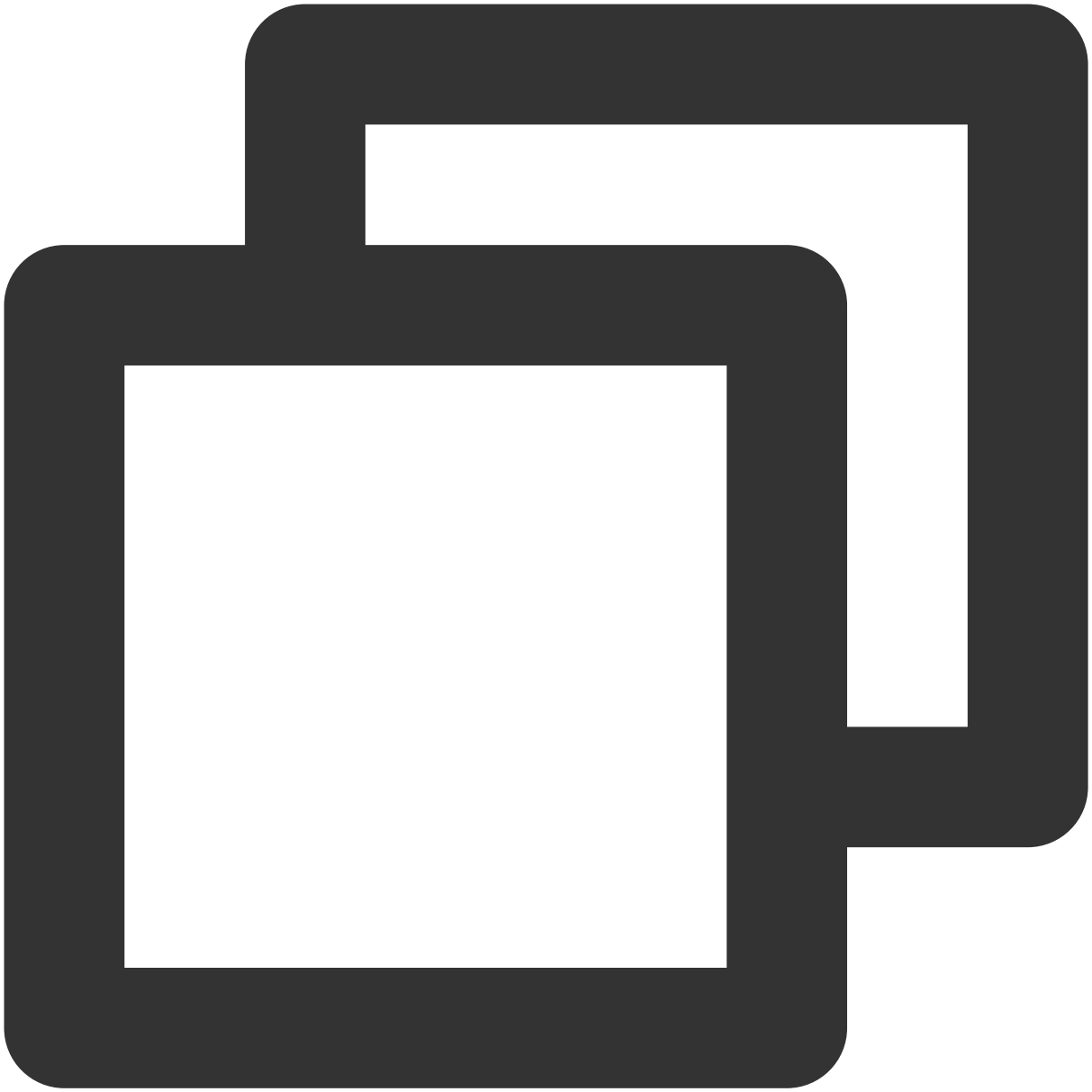
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。該当するseatIndexの座席上のキャスターはクローズされ、自動的にマイク・オフになります。

ローカル音声操作のインターフェース

startMicrophone

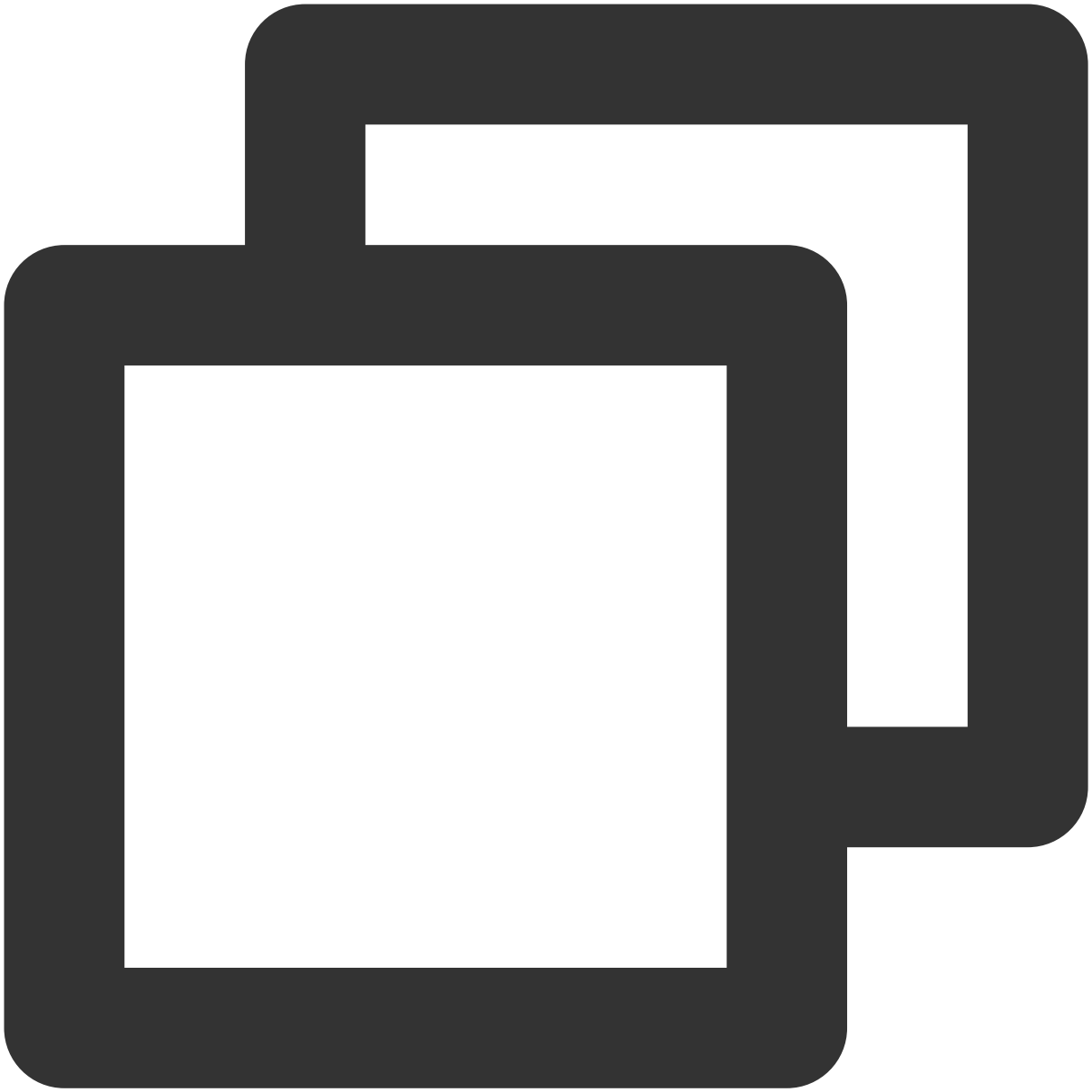
マイクの集音開始。



```
- (void)startMicrophone;
```

stopMicrophone

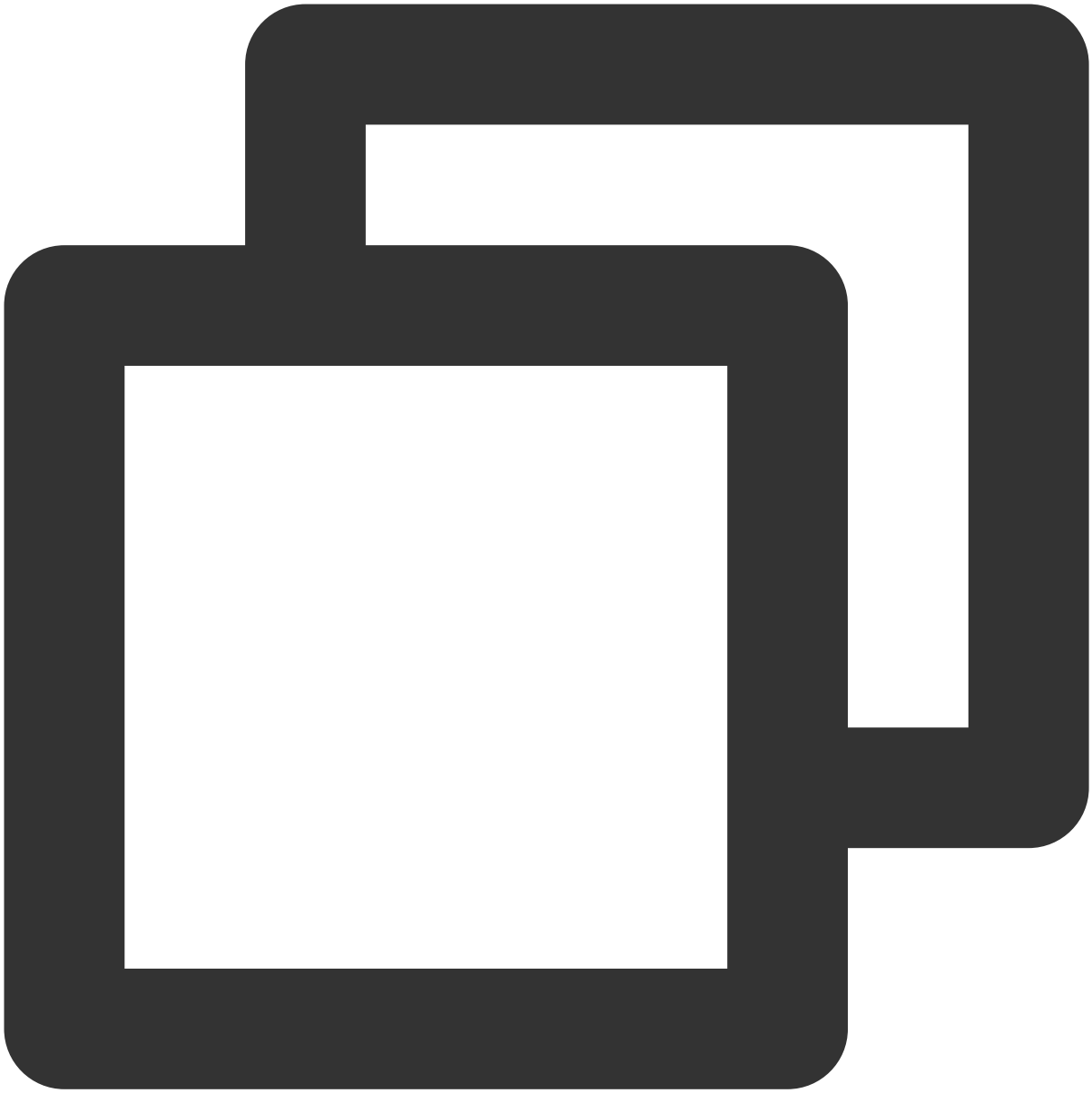
マイクの集音停止。



```
- (void)stopMicrophone;
```

setAudioQuality

音質の設定。



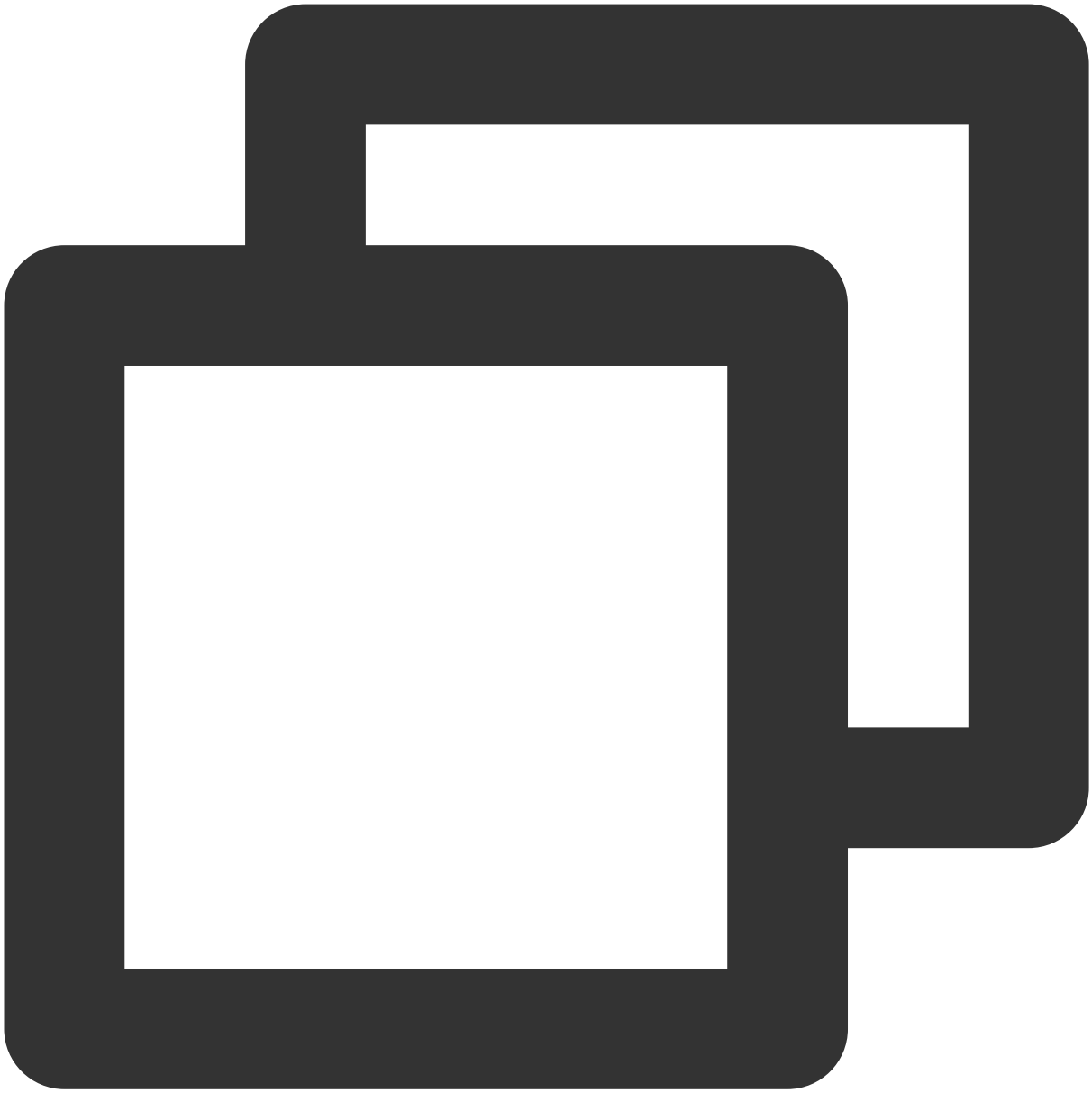
```
- (void)setAuidoQuality:(NSInteger)quality NS_SWIFT_NAME(setAuidoQuality(quality:))
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
quality	int	音声品質。詳細は TRTC SDKをご参照ください。

muteLocalAudio

ローカルの音声のミュート/ミュート取り消し。



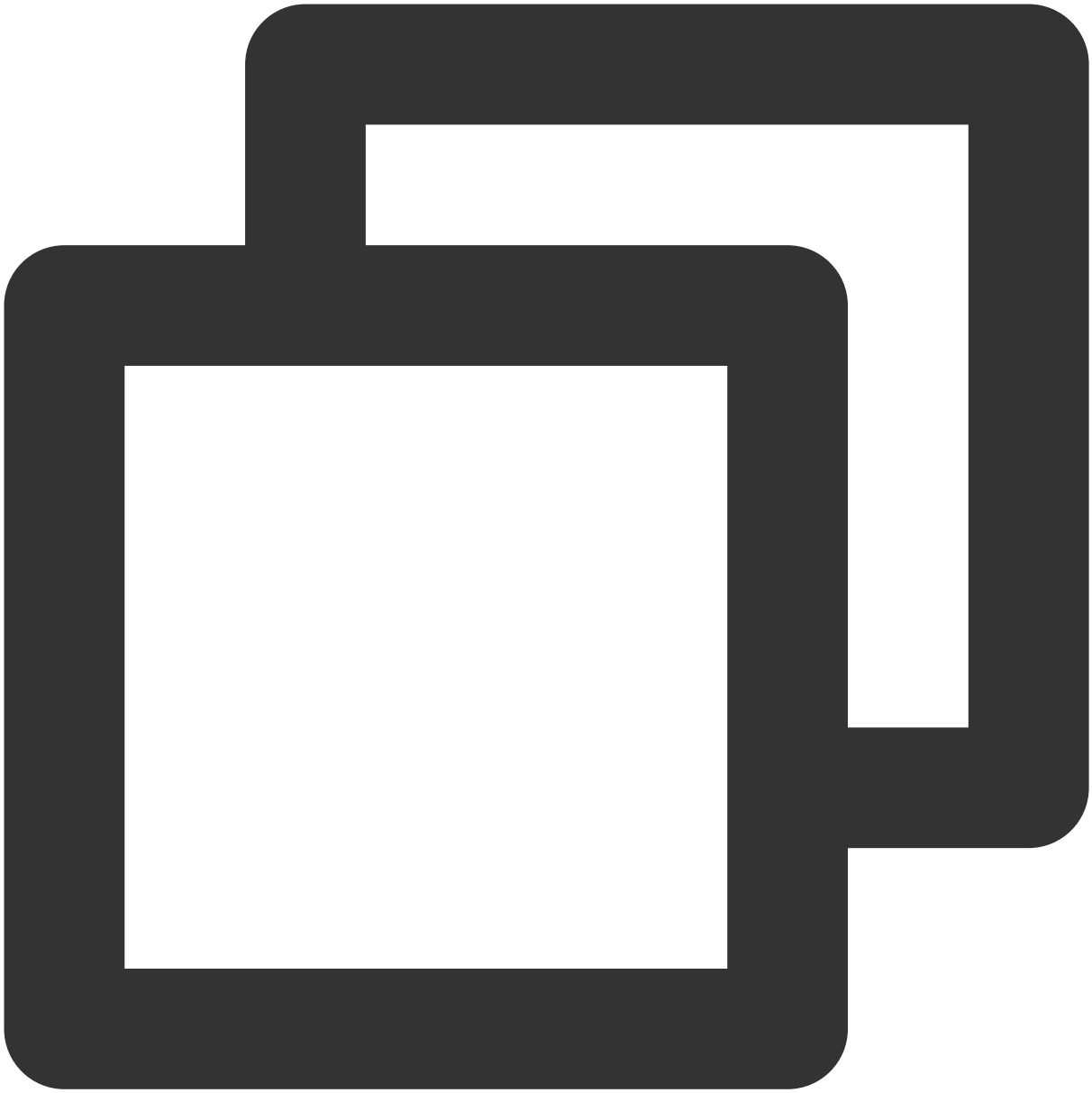
```
- (void)muteLocalAudio:(BOOL)mute NS_SWIFT_NAME(muteLocalAudio(mute:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
mute	boolean	ミュート/ミュート取り消し。詳細は TRTC SDKをご参照ください。

setSpeaker

スピーカーの起動設定。



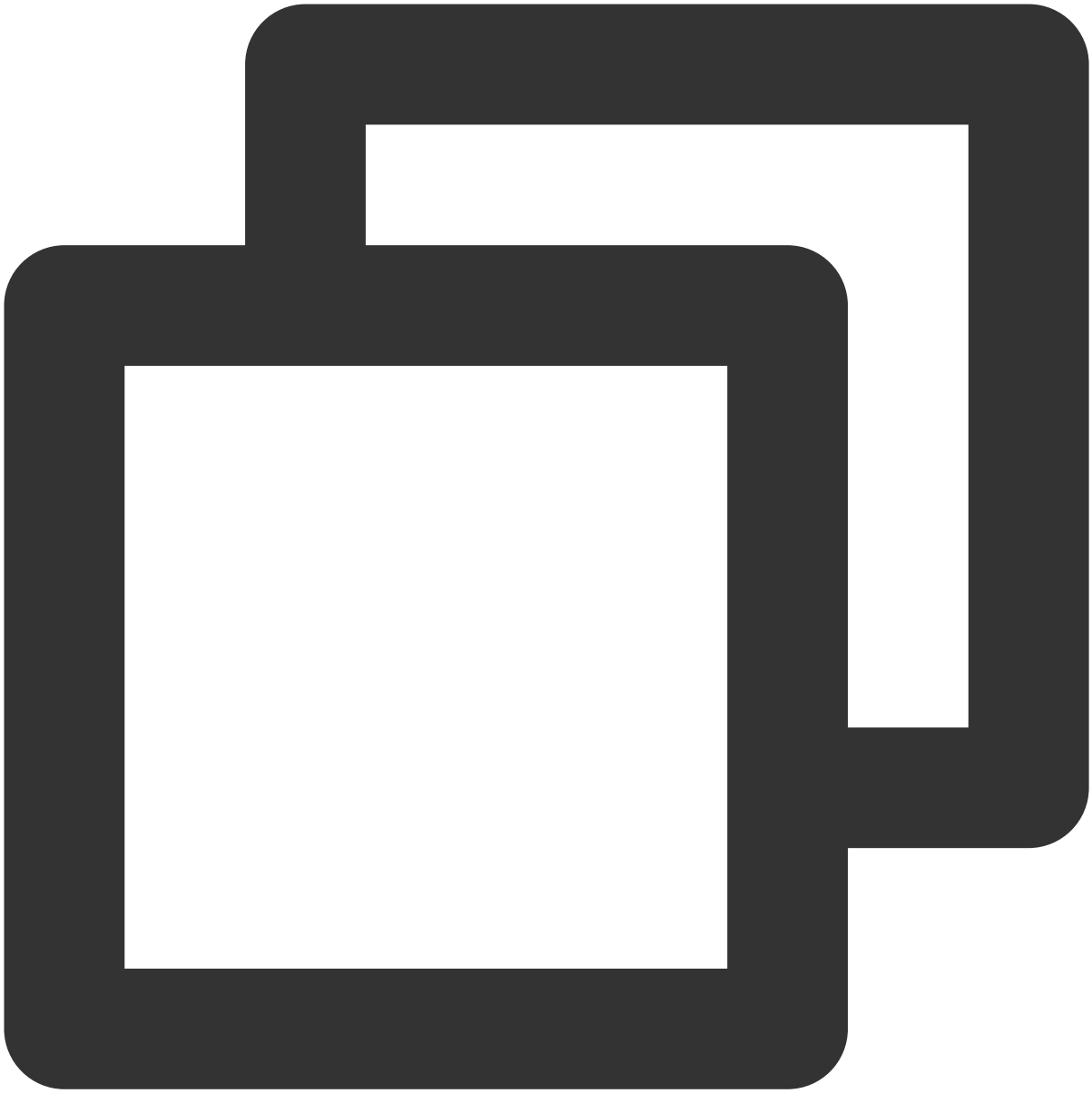
```
- (void)setSpeaker:(BOOL)userSpeaker NS_SWIFT_NAME(setSpeaker(userSpeaker:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
useSpeaker	boolean	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。



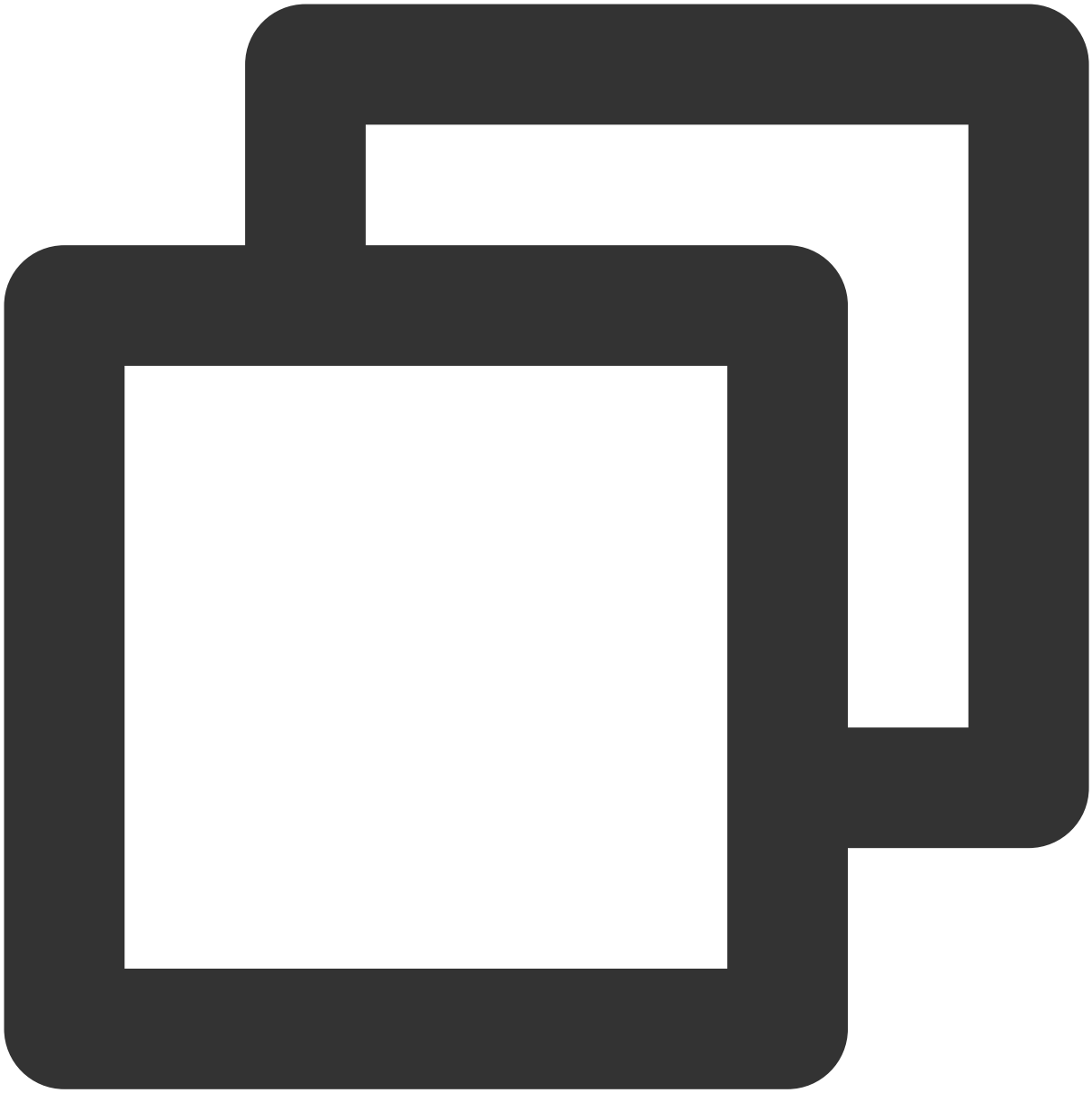
```
- (void)setAudioCaptureVolume:(NSInteger)volume NS_SWIFT_NAME(setAudioCaptureVolume
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
volume	int	集音音量、0 - 100、 デフォルト100。

setAudioPlayoutVolume

再生音量の設定。



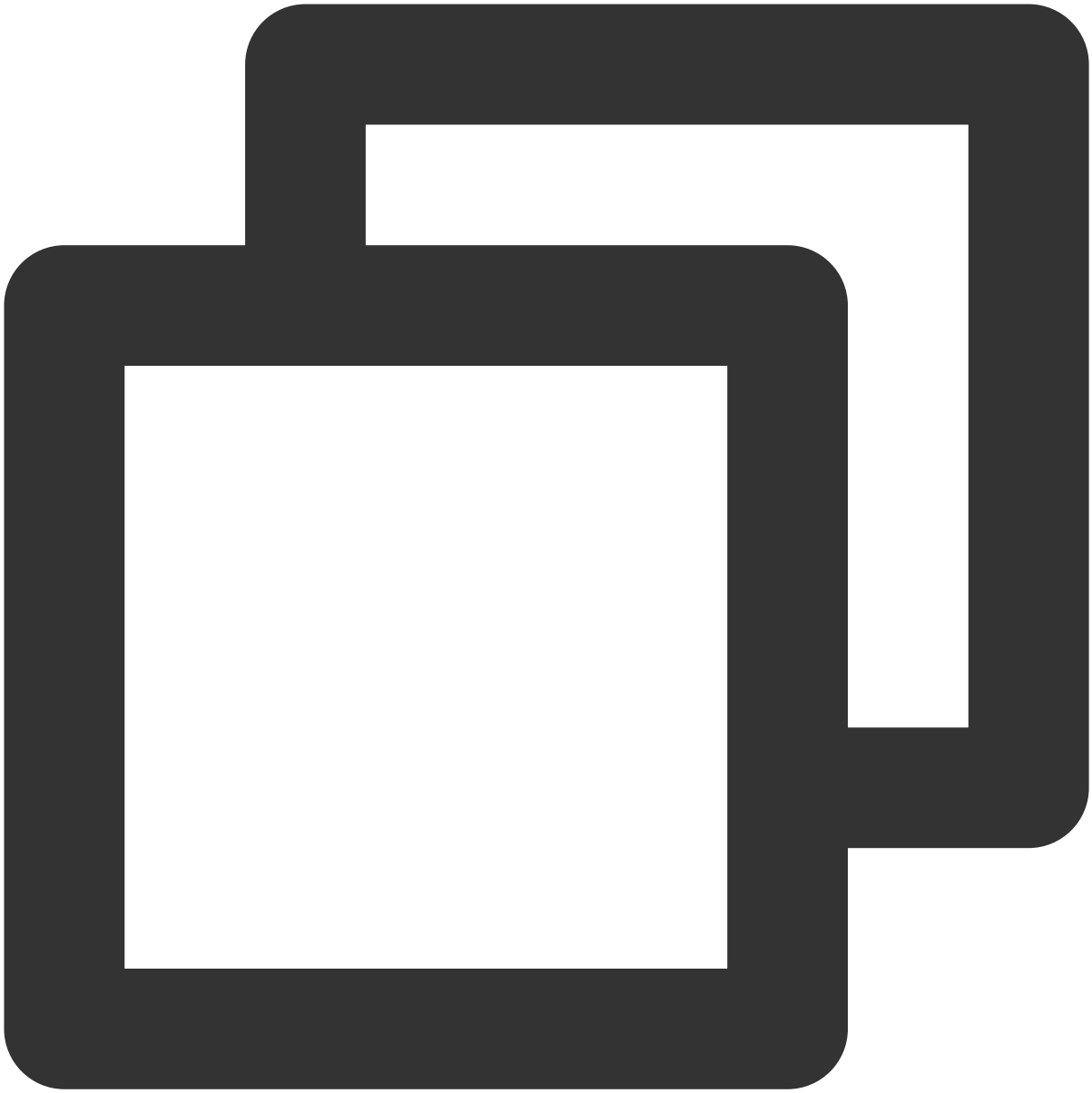
```
- (void)setAudioPlayoutVolume:(NSInteger)volume NS_SWIFT_NAME(setAudioPlayoutVolume
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
volume	int	再生音量、0 - 100、 デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。



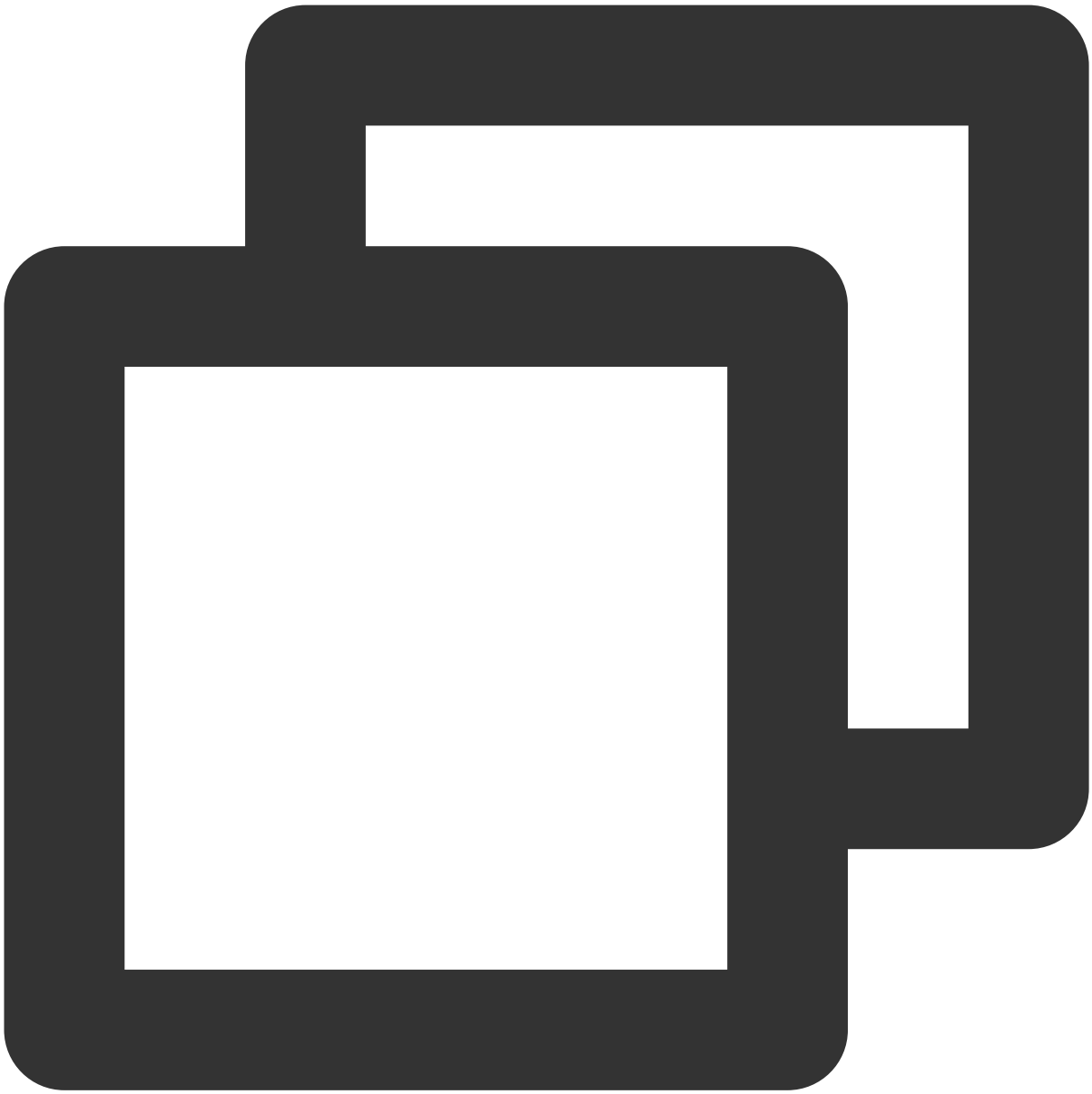
```
- (void)muteRemoteAudio:(NSString *)userId mute:(BOOL)mute NS_SWIFT_NAME(muteRemote
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userId	String	指定ユーザーID。
mute	boolean	true：ミュートをオンにします；false：ミュートをオフにします。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。



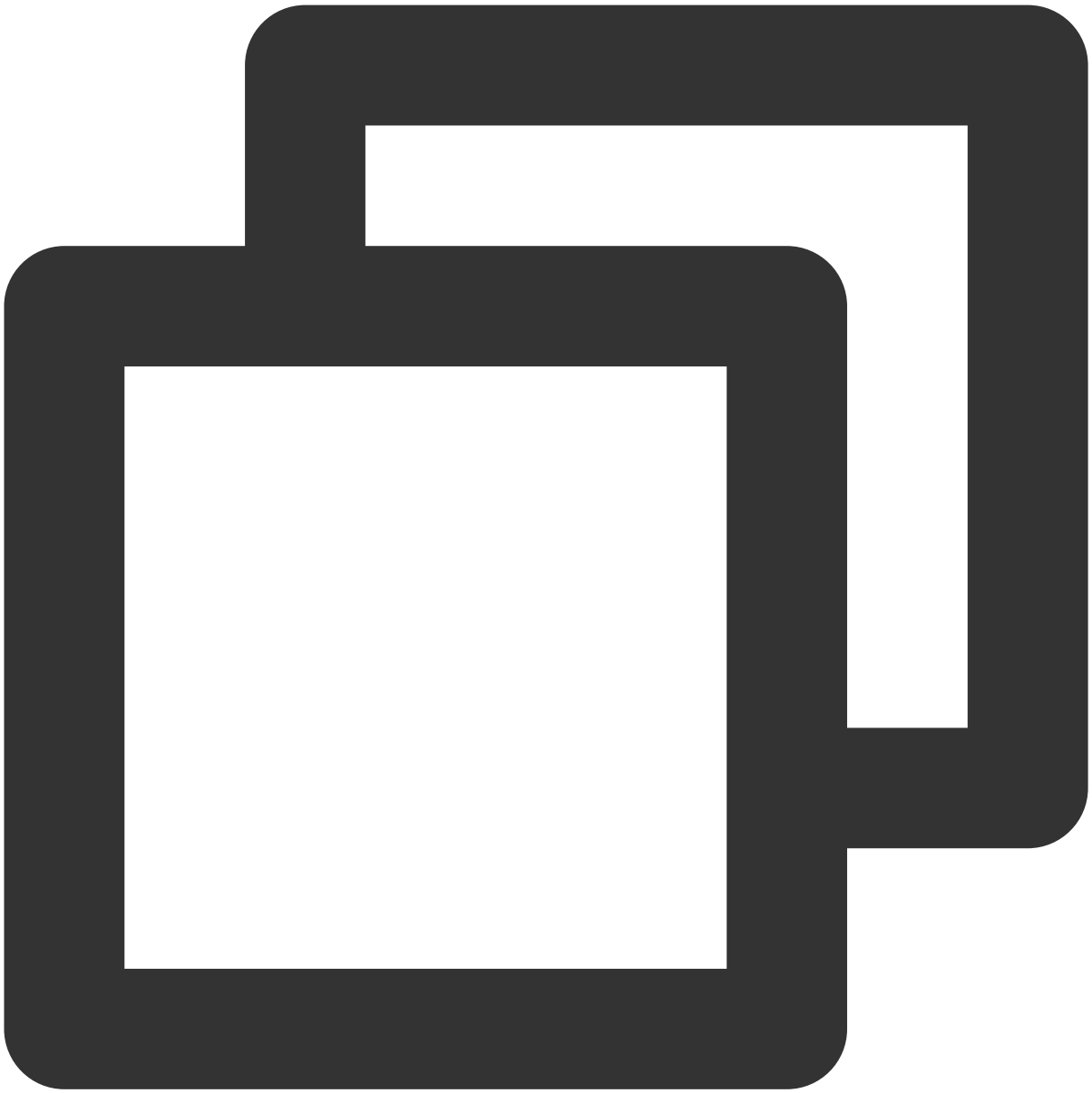
```
- (void)muteAllRemoteAudio:(BOOL)isMute NS_SWIFT_NAME(muteAllRemoteAudio(isMute:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
isMute	boolean	true：ミュートをオンにします；false：ミュートをオフにします。

setVoiceEarMonitorEnable

インイヤーモニタリングのオン/オフ。



```
- (void)setVoiceEarMonitorEnable:(BOOL)enable NS_SWIFT_NAME(setVoiceEarMonitor(enab
```

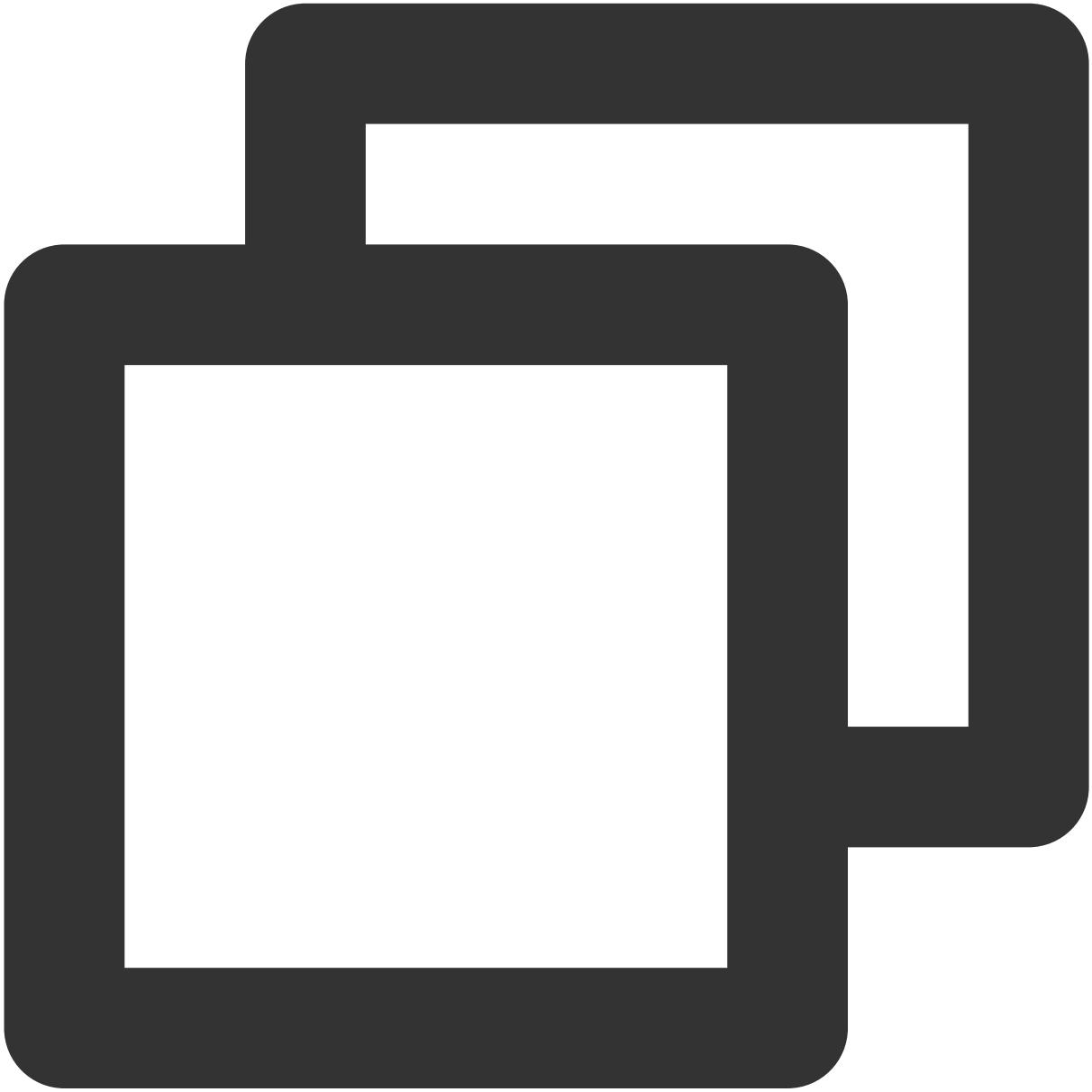
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
enable	boolean	true：インイヤーモニタリングをオン。false：インイヤーモニタリングをオフ。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

BGMサウンドエフェクト管理オブジェクト TXAudioEffectManagerを取得します。

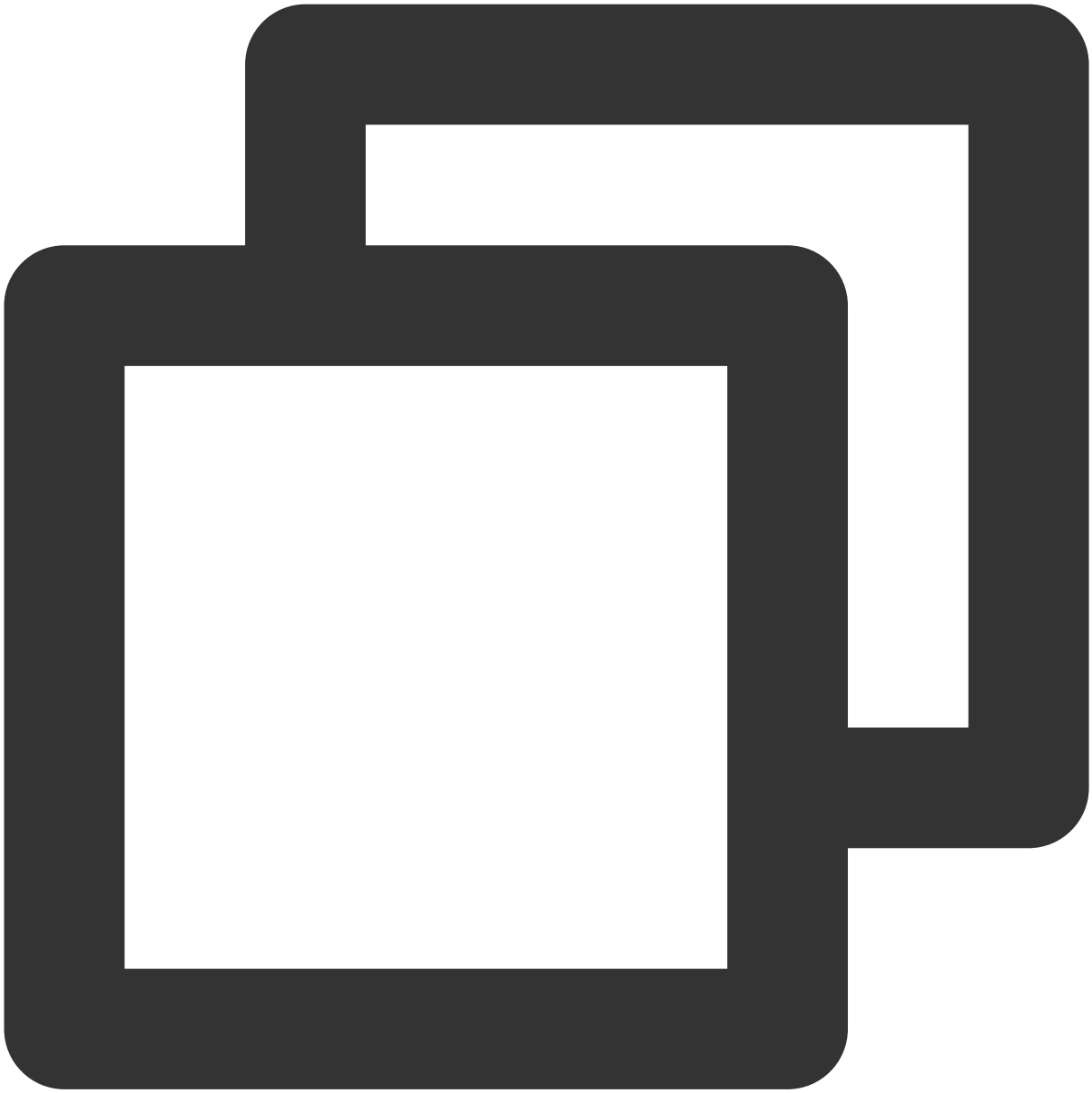


```
- (TXAudioEffectManager * _Nullable)getAudioEffectManager;
```

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。



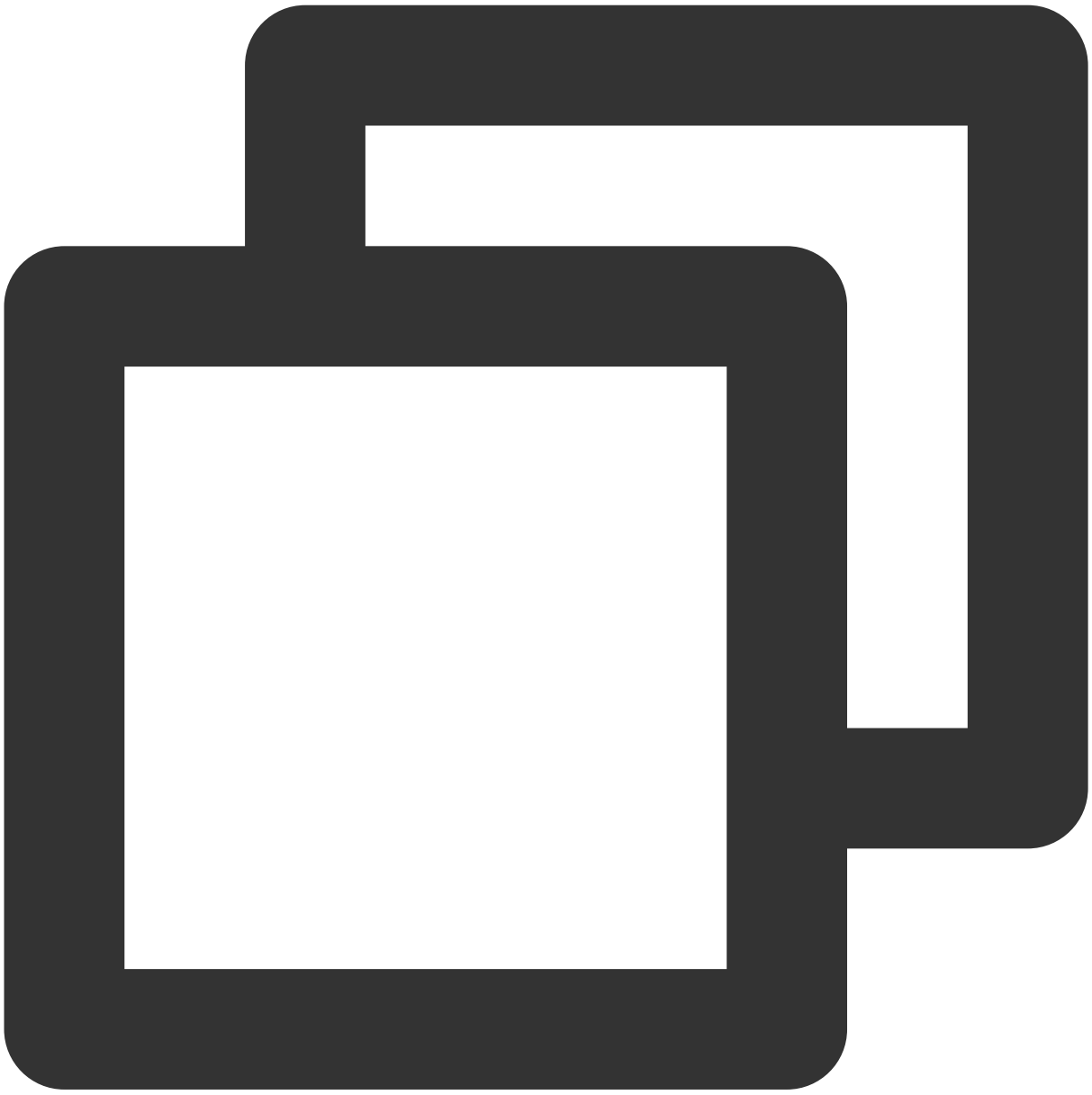
```
- (void)sendRoomTextMsg:(NSString *)message callback:(ActionCallback _Nullable)call
```

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

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。



```
- (void)sendRoomCustomMsg:(NSString *)cmd message:(NSString *)message callback:(Act
```

パラメータは下表に示すとおりです：

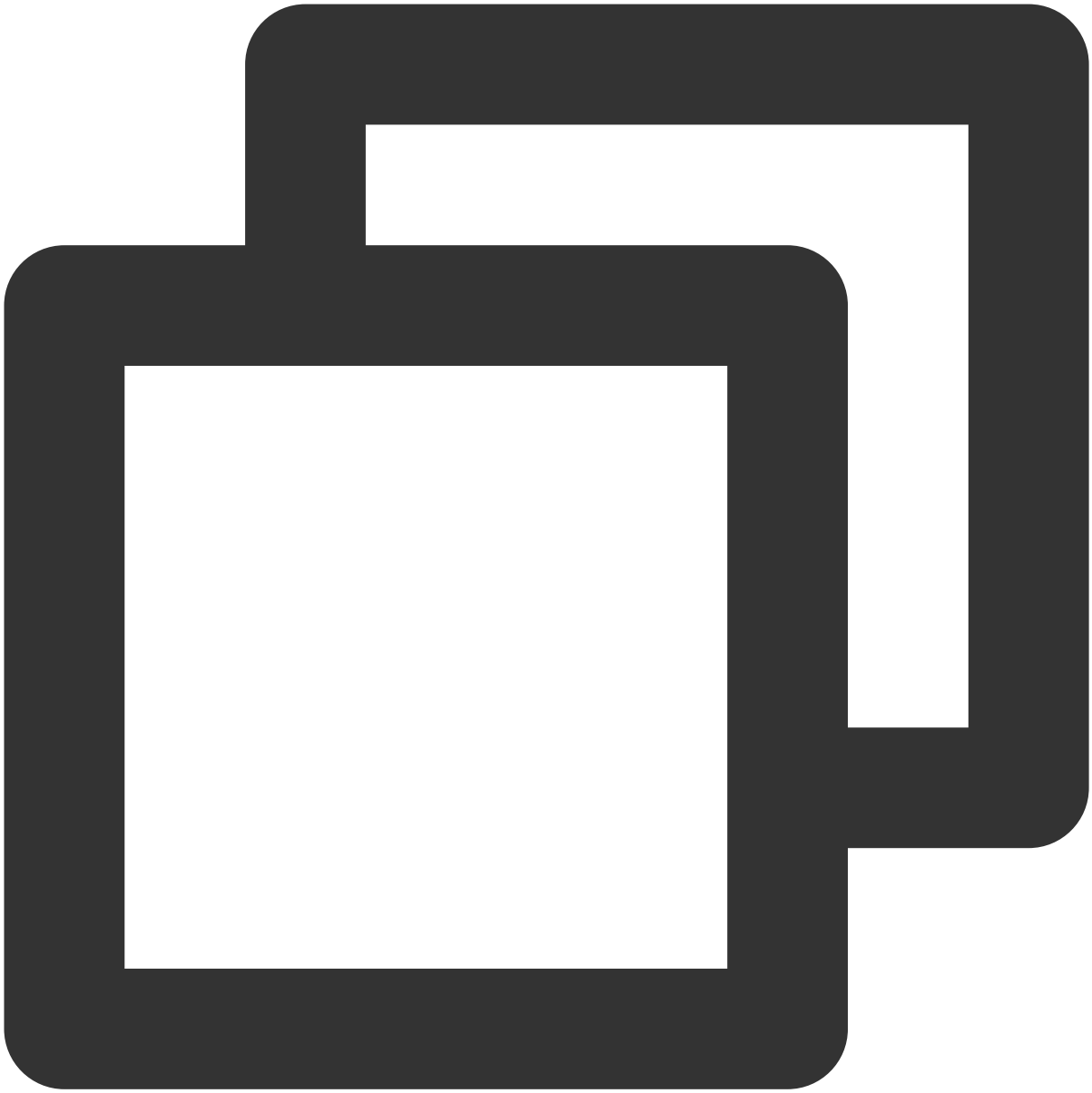
パラメータ	タイプ	意味

cmd	String	コマンドワードです。開発者がカスタマイズするものであり、主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

招待シグナリング関連インターフェース

sendInvitation

ユーザーに招待を送信。



```
- (NSString *)sendInvitation:(NSString *)cmd
                      userId:(NSString *)userId
                     content:(NSString *)content
                    callback:(ActionCallback _Nullable)callback NS_SWIFT_NAME(sendI
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
cmd	String	業務カスタマイズコマンド。

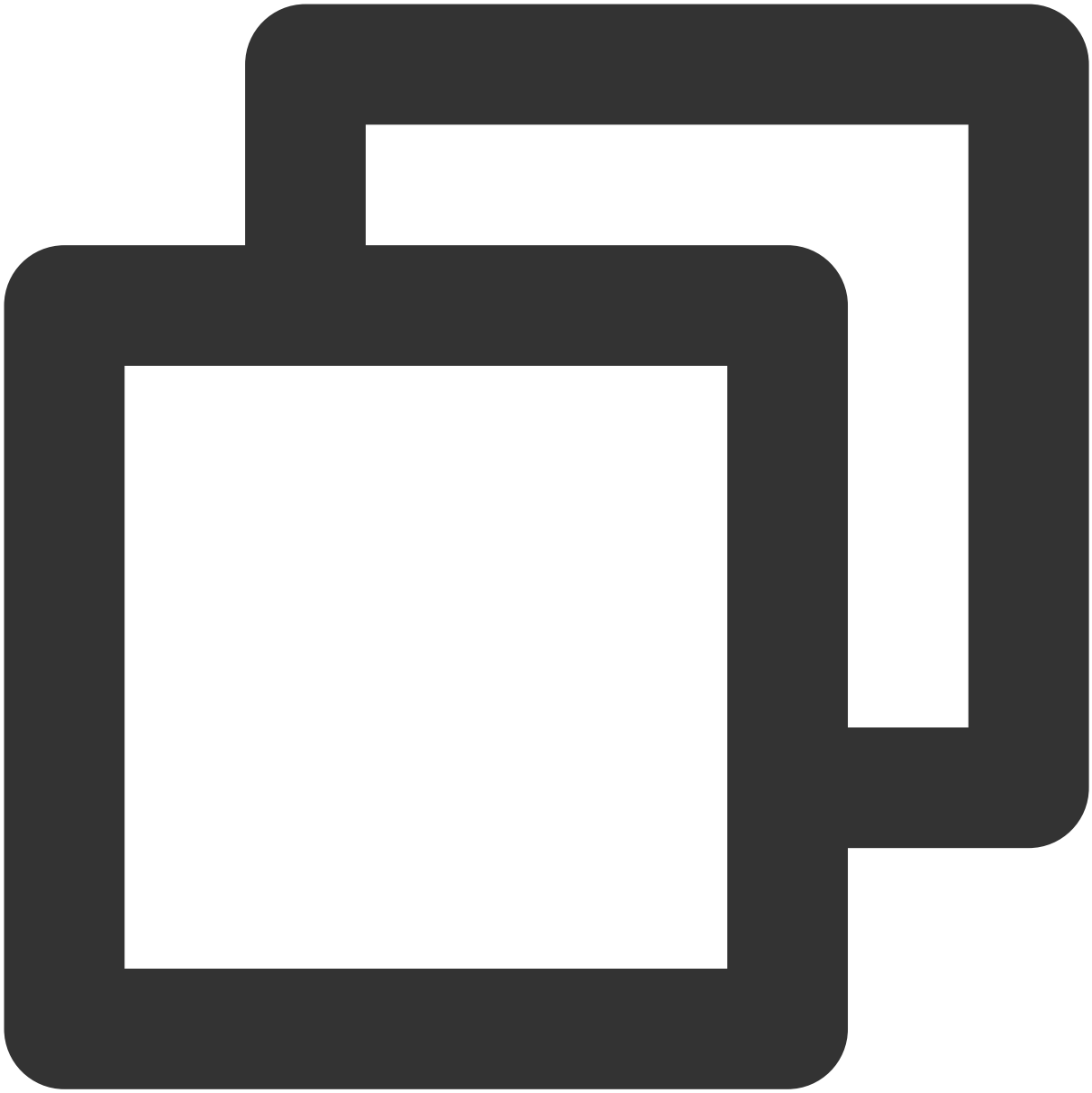
userId	String	招待ユーザーID。
content	String	招待コンテンツ。
callback	ActionCallback	送信結果のコールバック。

戻り値：

戻り値	タイプ	意味
inviteId	String	今回の招待IDの識別に使用。

acceptInvitation

招待の同意。



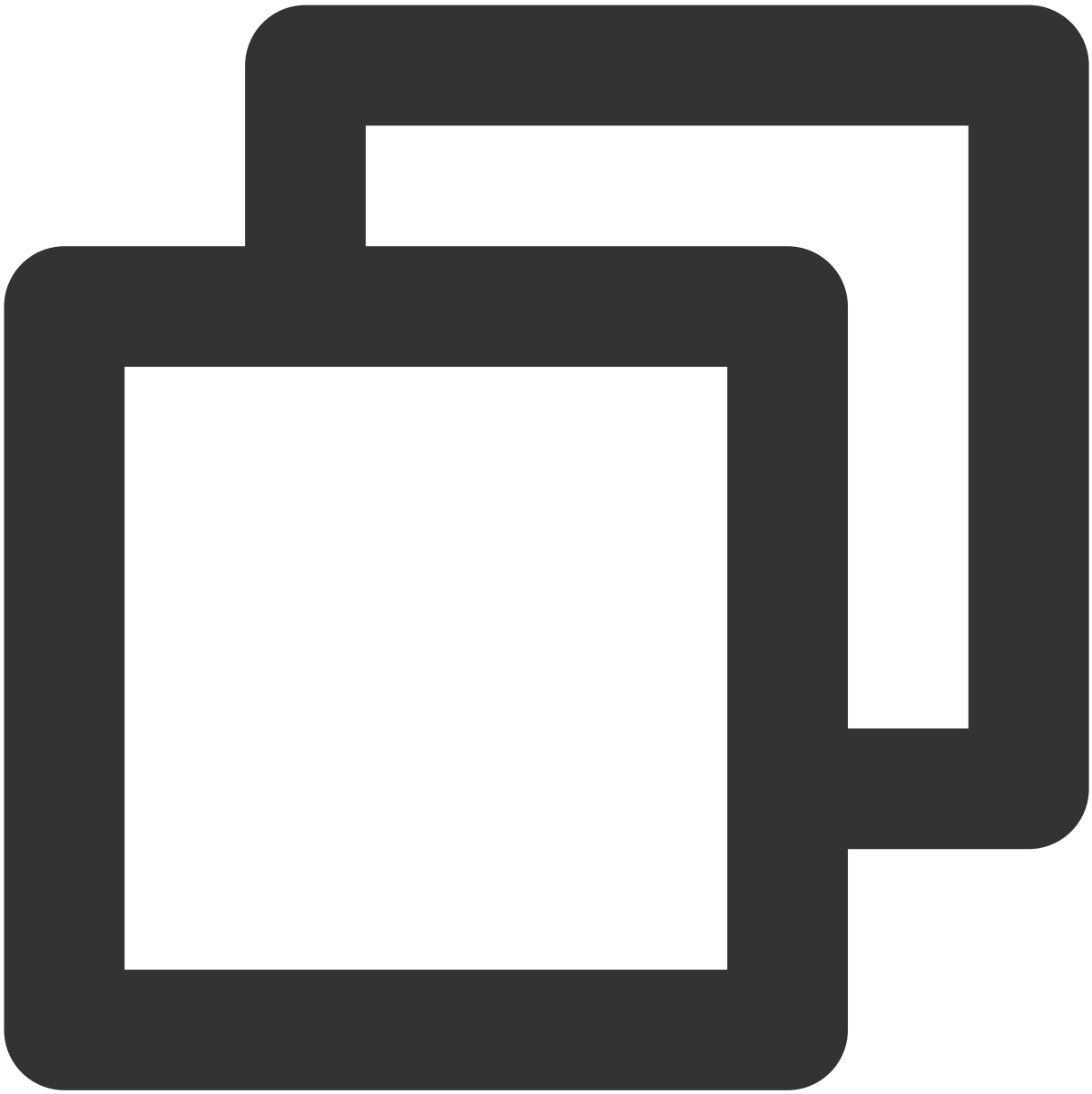
```
- (void)acceptInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

rejectInvitation

招待の拒否。



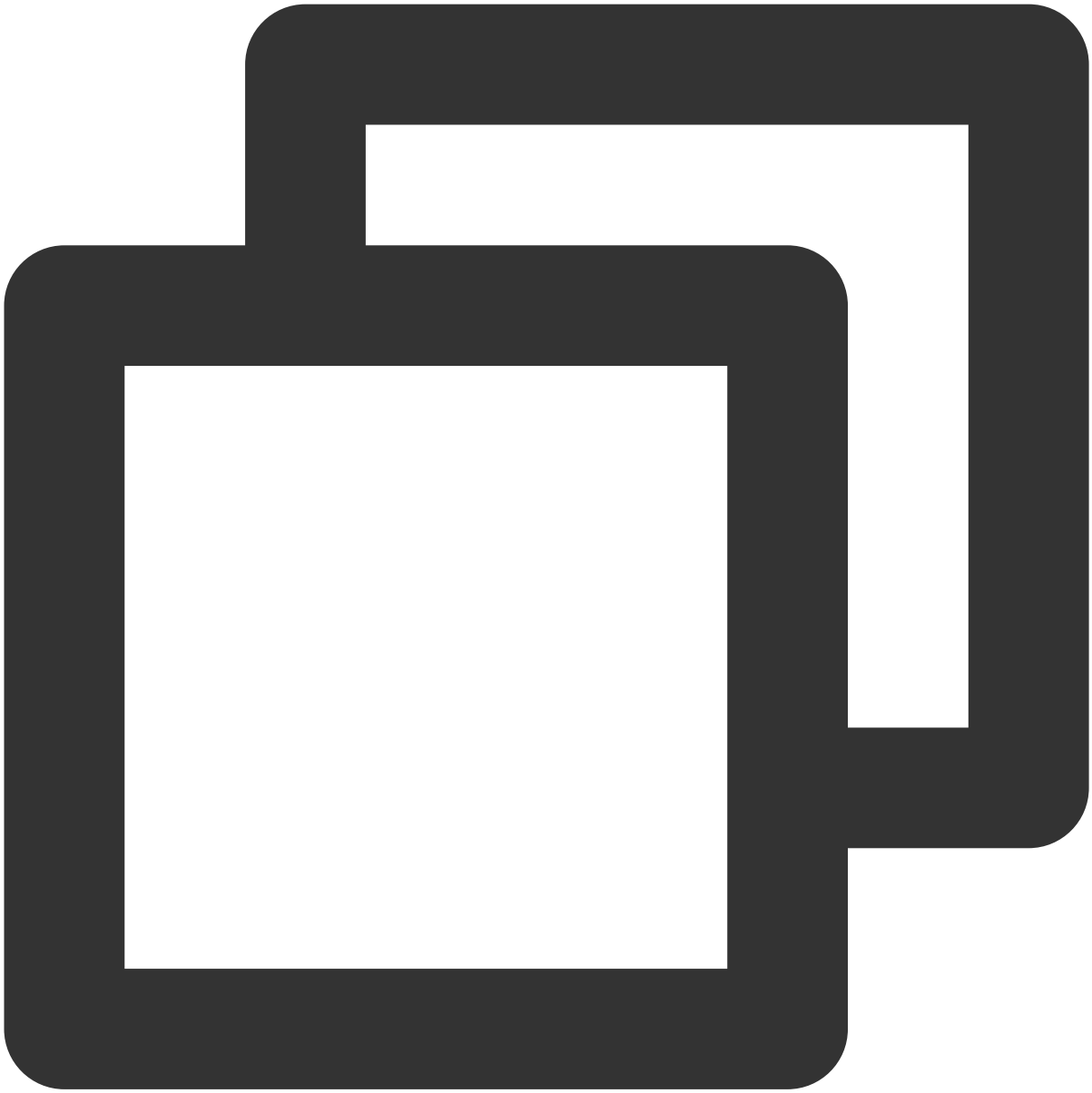
```
- (void)rejectInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

cancelInvitation

招待の取り消し。



```
- (void)cancelInvitation:(NSString *)identifier callback:(ActionCallback _Nullable)
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

TRTCKaraokeRoomDelegateイベントコールバック

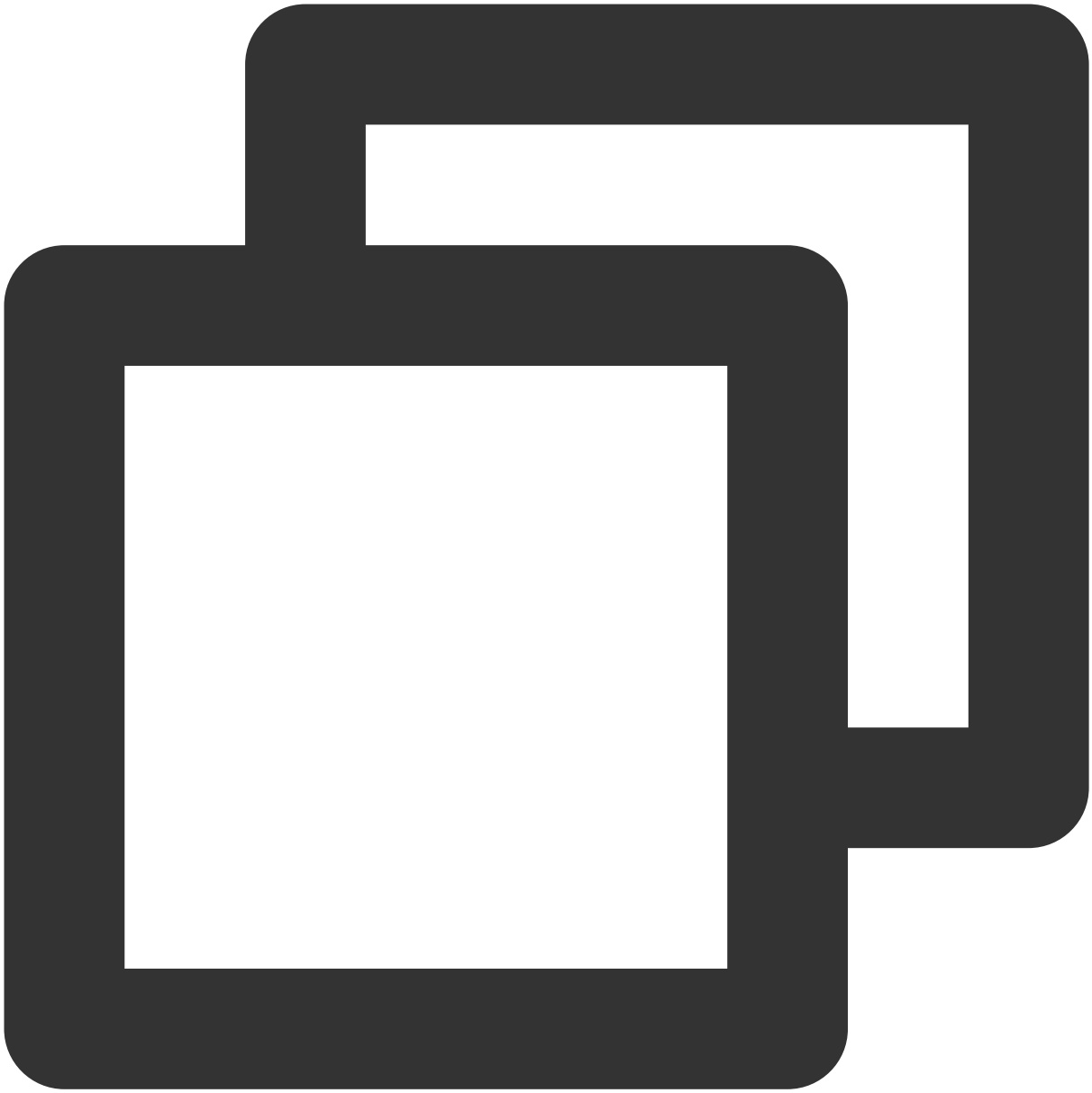
一般的なイベントコールバック

onError

エラーのコールバック。

説明：

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



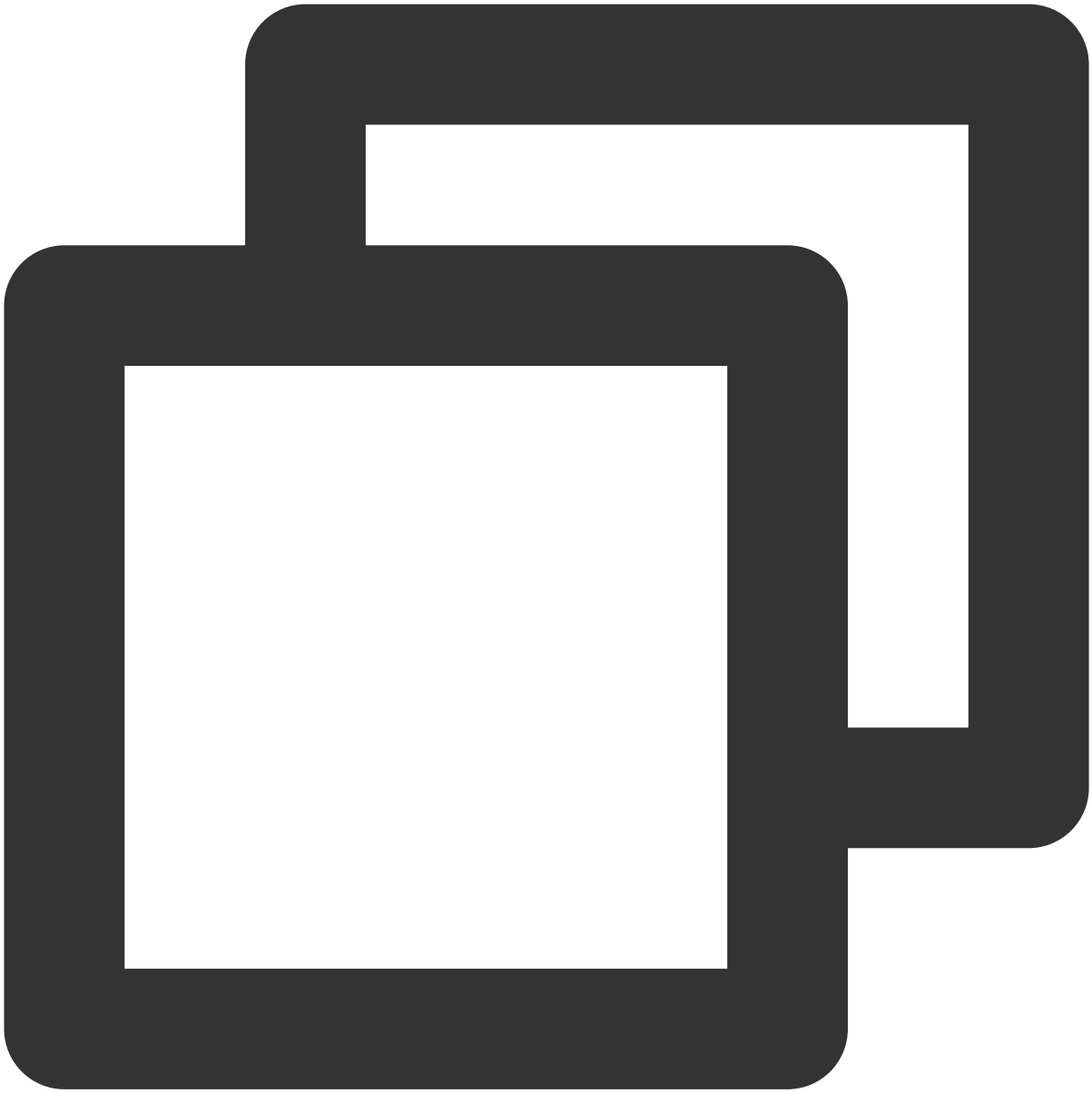
```
- (void)onError:(int)code
                message:(NSString*)message
NS_SWIFT_NAME(onError(code:message:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラーメッセージ。

onWarning

警告のコールバック。



```
- (void)onWarning:(int)code
                  message:(NSString  *)message
NS_SWIFT_NAME(onWarning(code:message:));
```

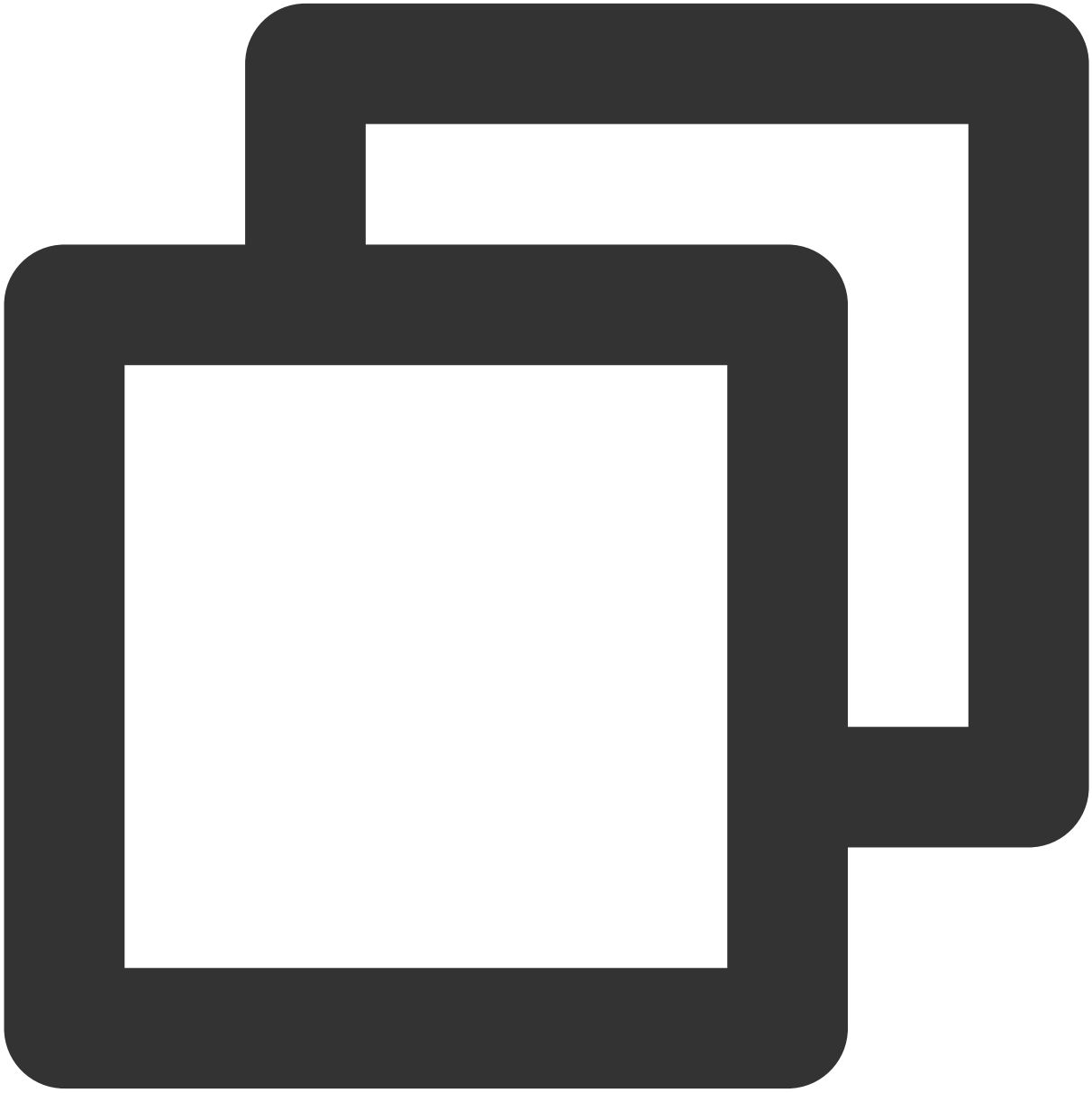
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

code	int	エラーコード。
message	String	警告メッセージ。

onDebugLog

Logコールバック。



```
- (void)onDebugLog:(NSString *)message
NS_SWIFT_NAME(onDebugLog(message:));
```

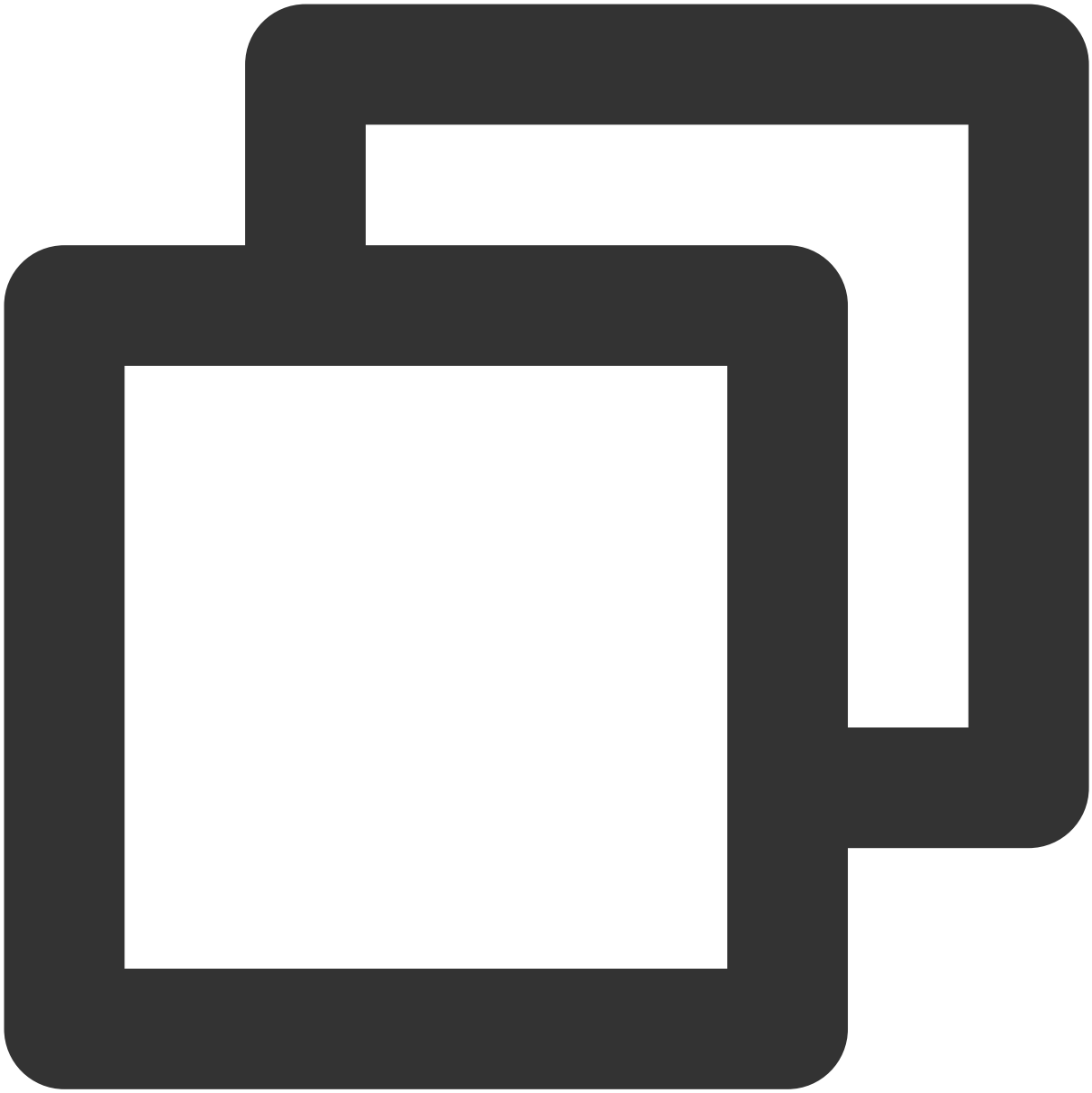
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
message	String	ログ情報。

ルームイベントのコールバック

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。



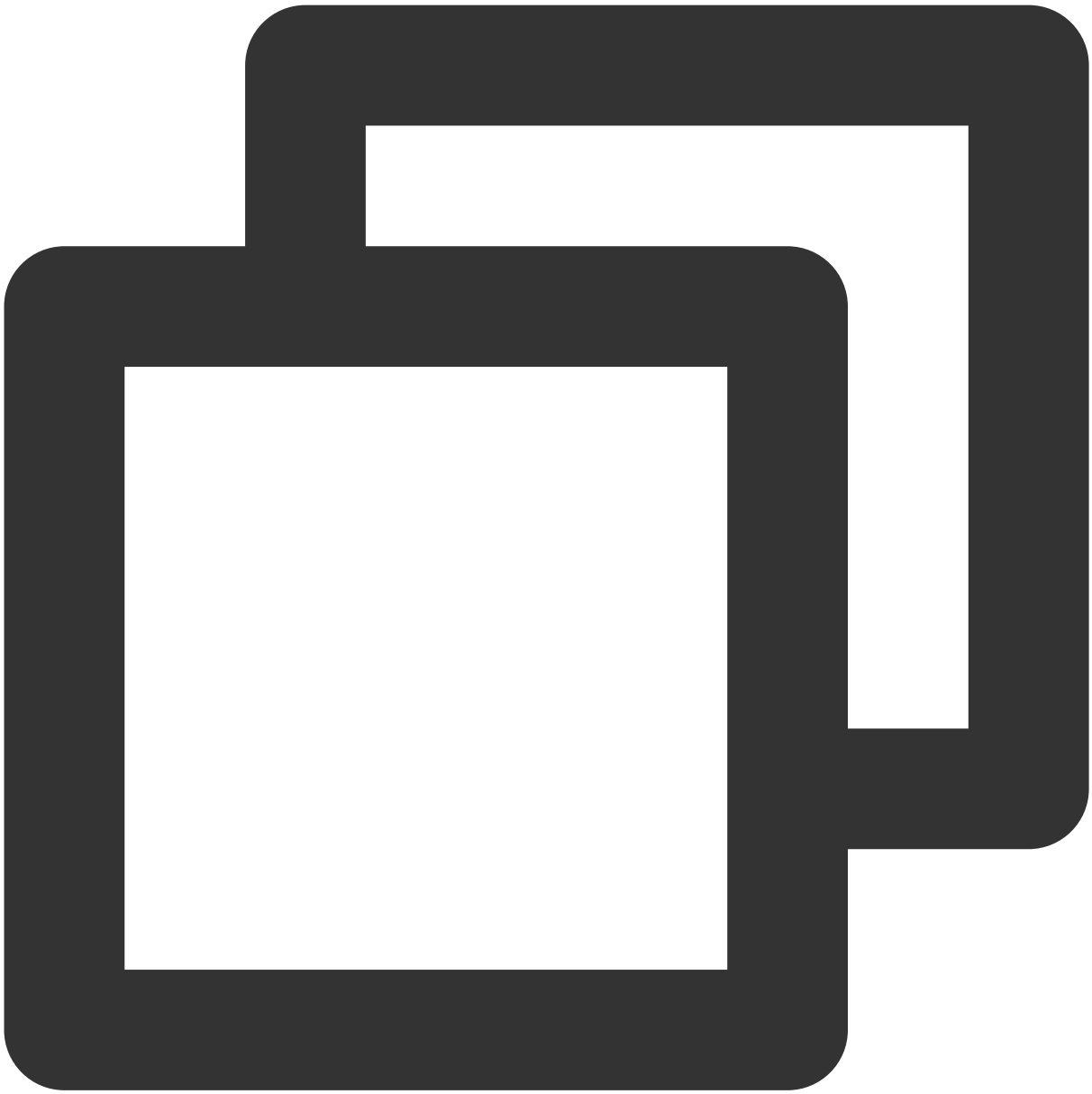
```
- (void)onRoomDestroy:(NSString *)message
NS_SWIFT_NAME(onRoomDestroy(message:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
message	String	コールバック情報。

onRoomInfoChange

入室に成功後、このインターフェースをコールバックします。roomInfoの情報は、管理者がルームを作成するときに渡されます。



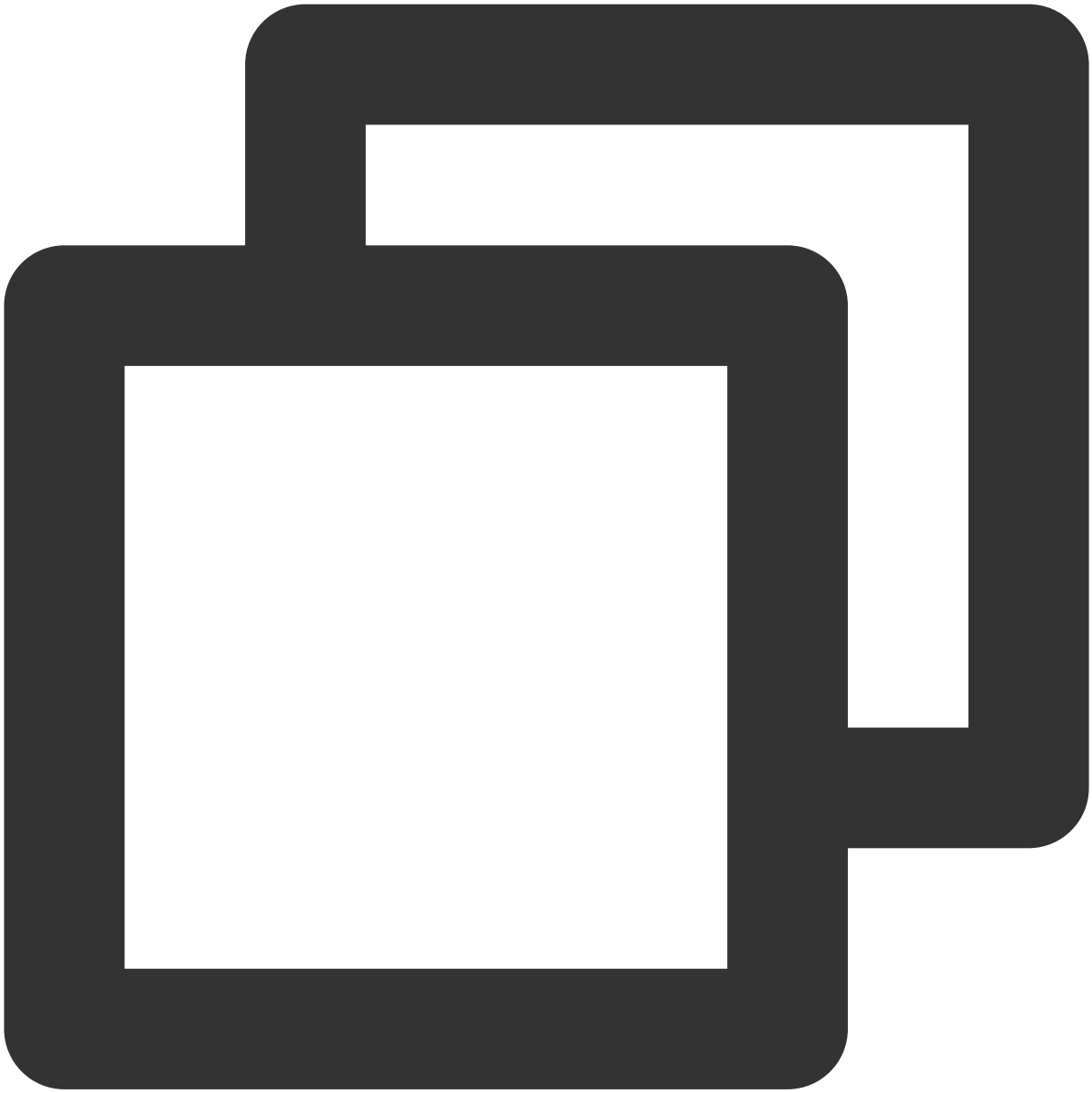
```
- (void)onRoomInfoChange:(KaraokeInfo *)roomInfo
NS_SWIFT_NAME(onRoomInfoChange(roomInfo:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
roomInfo	RoomInfo	ルーム情報。

onUserMicrophoneMute

ユーザーのマイクがミュートになっているかどうかのコールバックです。ユーザーがmuteLocalAudioを呼び出すと、ルームの他のユーザーがこの通知を受信します。



```
- (void)onUserMicrophoneMute:(NSString *)userId mute:(BOOL)mute
NS_SWIFT_NAME(onUserMicrophoneMute(userId:mute:));
```

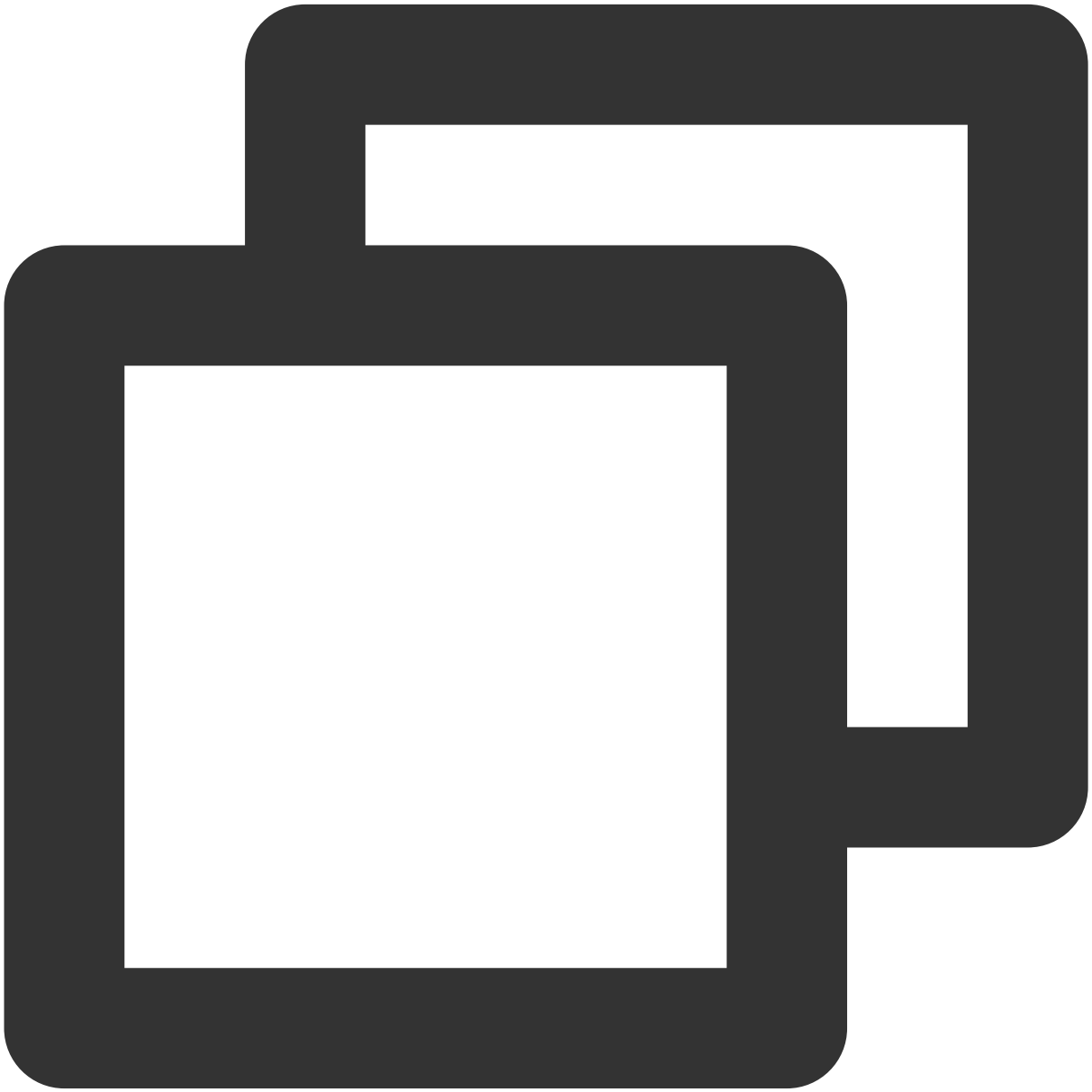
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

userId	String	ユーザーID。
mute	boolean	音量の大きさ。値：0～100。

onUserVolumeUpdate

音量レベルリマインダを有効にすると、各メンバーの音量を通知します。



```
- (void)onUserVolumeUpdate:(NSArray<TRTCVolumeInfo *> *)userVolumes totalVolume:(NS
NS_SWIFT_NAME(onUserVolumeUpdate(userVolumes:totalVolume:));
```

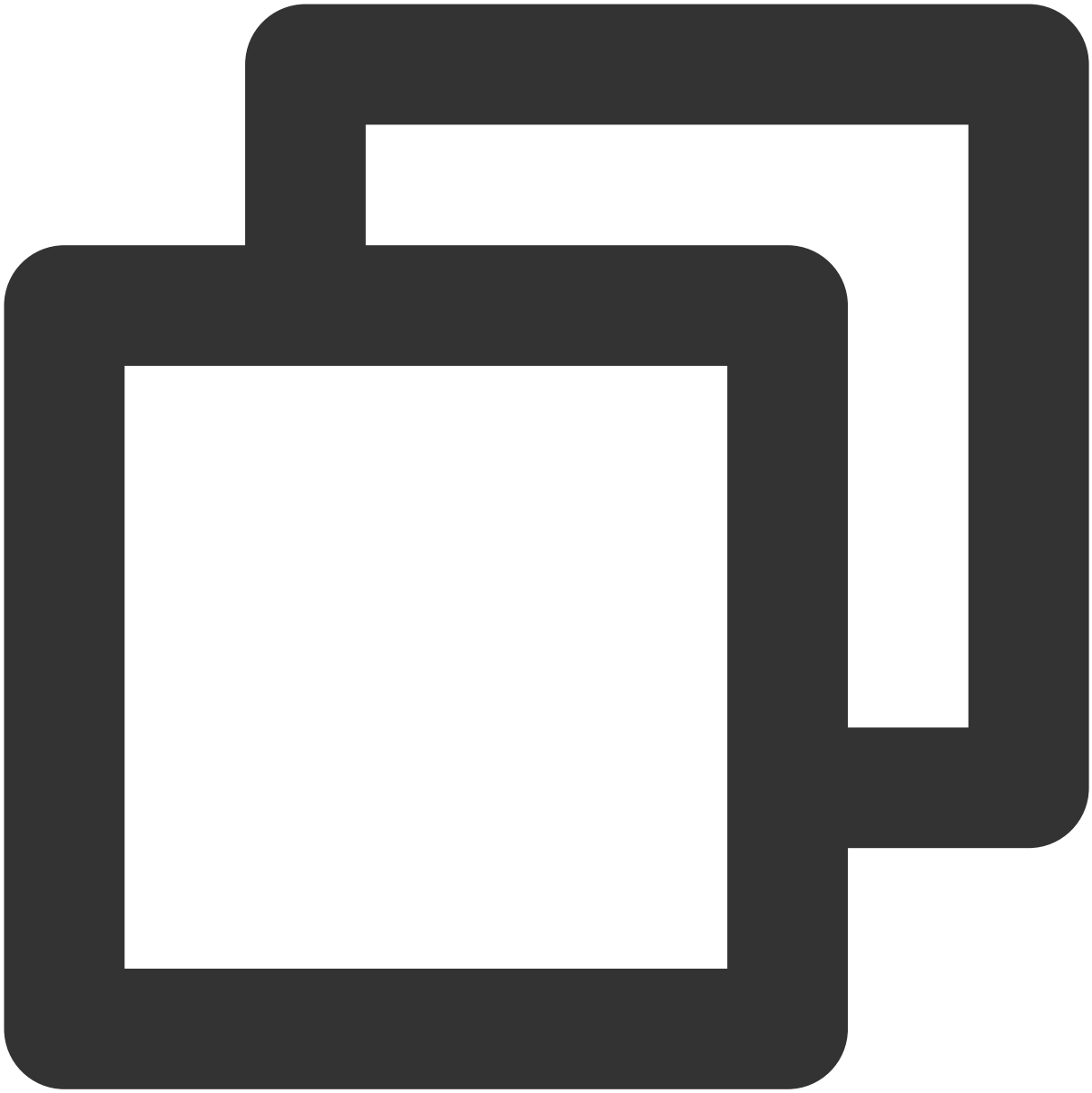
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userVolumes	List	ユーザーリスト。
totalVolume	int	音量の大きさ。値：0～100。

マイクコールバック

onSeatListChange

全量のマイクリストの変更です。全てのマイクリストを含みます。



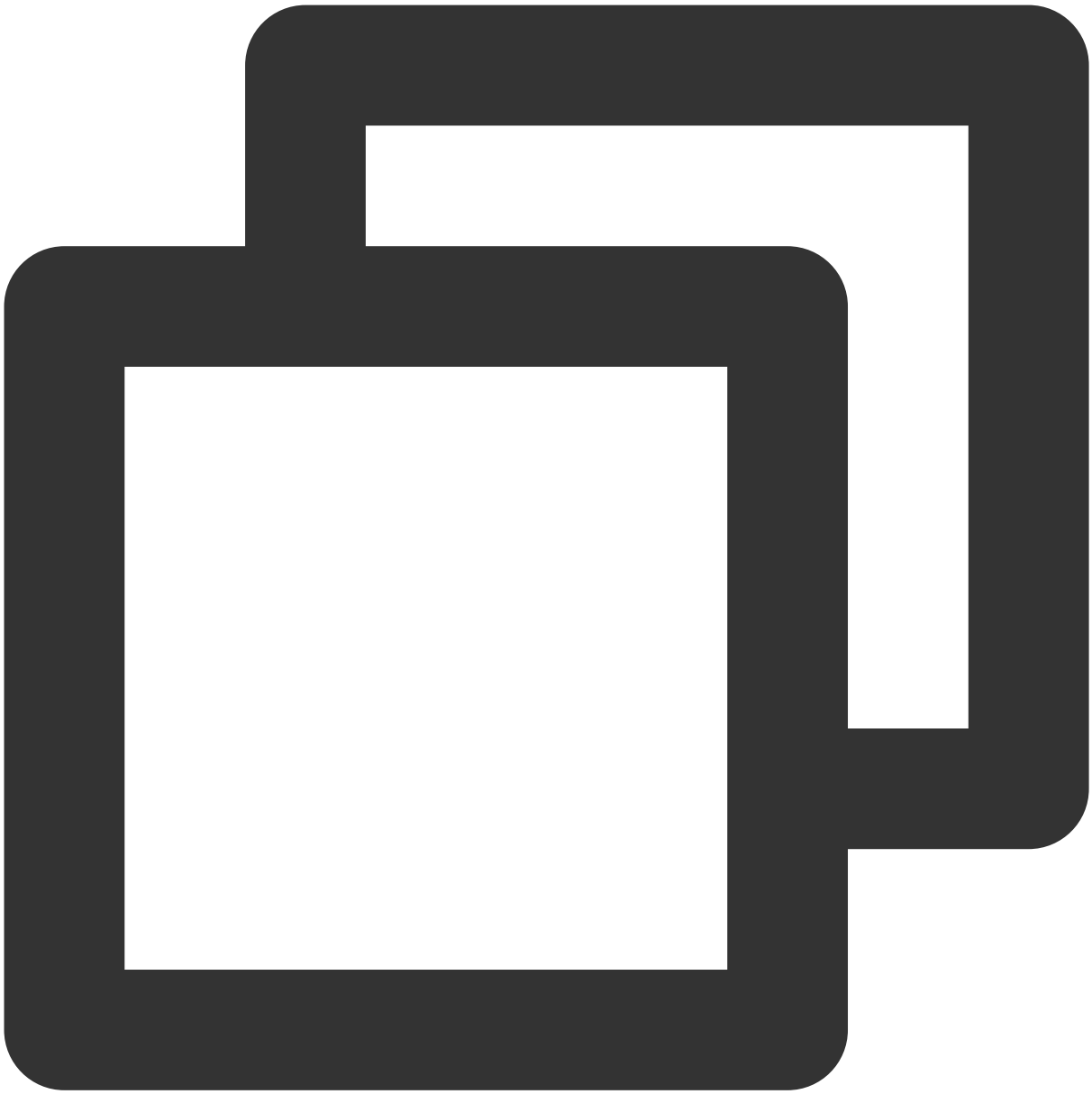
```
- (void)onSeatInfoChange:(NSArray<KaraokeSeatInfo *> *)seatInfolist
NS_SWIFT_NAME(onSeatListChange(seatInfoList:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatInfoList	List<SeatInfo>	全量のマイクリスト。

onAnchorEnterSeat

発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。



```
- (void)onAnchorEnterSeat:(NSInteger)index
                             user:(KaraokeUserInfo *)user
NS_SWIFT_NAME(onAnchorEnterSeat(index:user:));
```

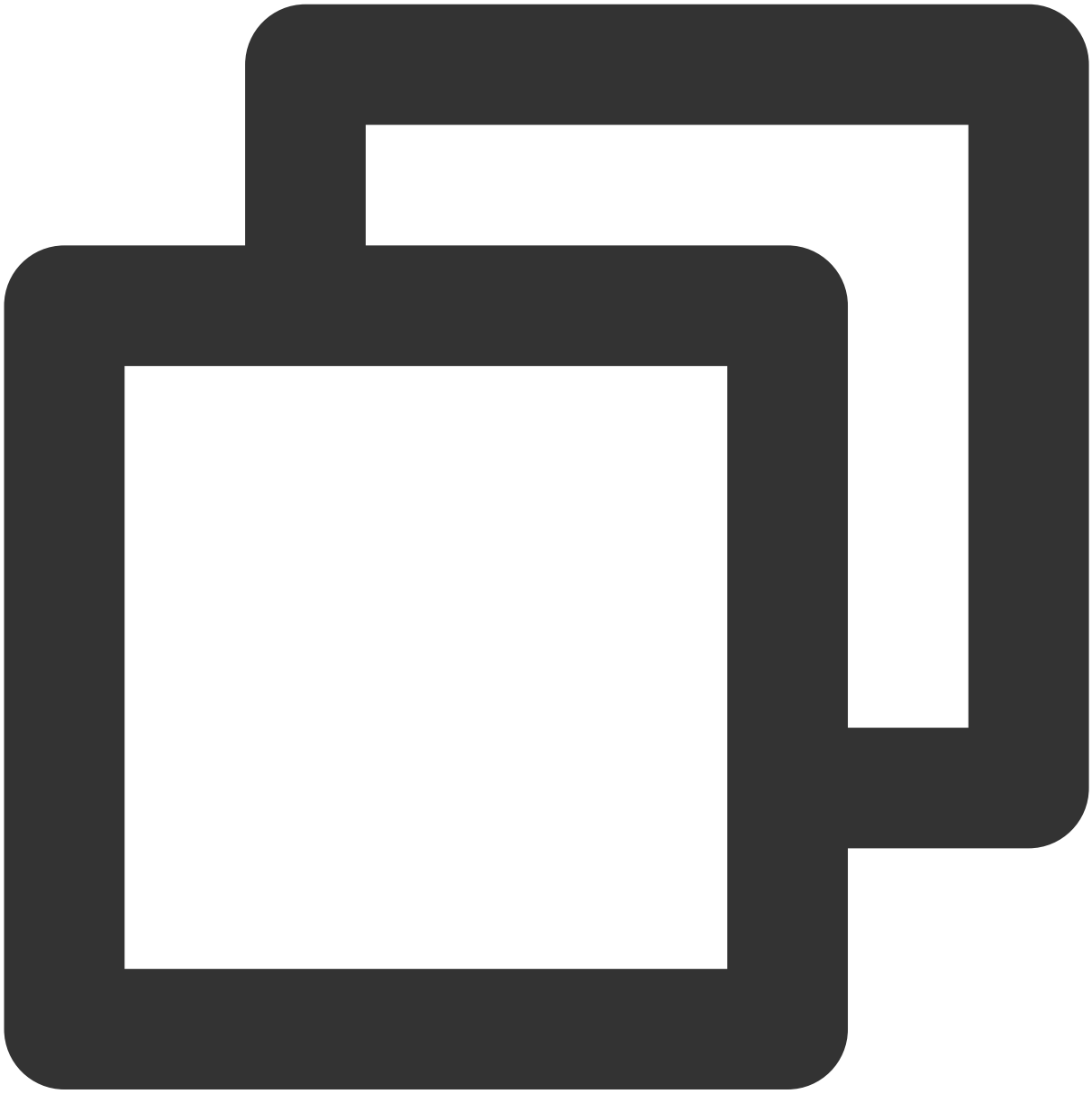
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
index	int	メンバーがマイク・オンのマイク。

user	UserInfo	マイク・オンのユーザーの詳細情報。

onAnchorLeaveSeat

視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。



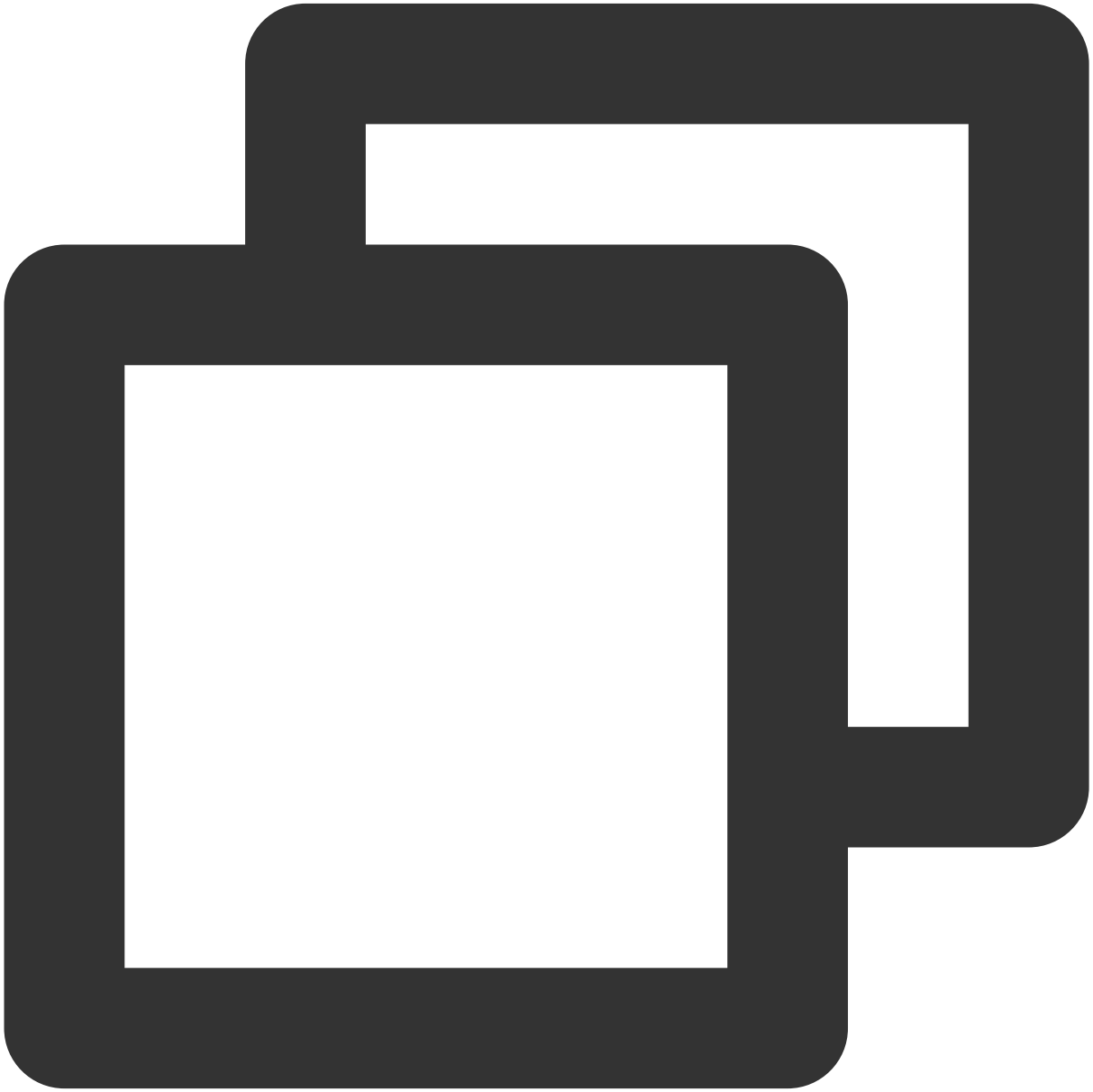
```
- (void)onAnchorLeaveSeat:(NSInteger)index
                     user:(KaraokeUserInfo *)user
NS_SWIFT_NAME(onAnchorLeaveSeat(index:user:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
index	int	マイク・オフのマイク。
user	UserInfo	マイク・オンのユーザーの詳細情報。

onSeatMute

管理者のマイクミュート。



```
- (void)onSeatMute:(NSInteger)index
```

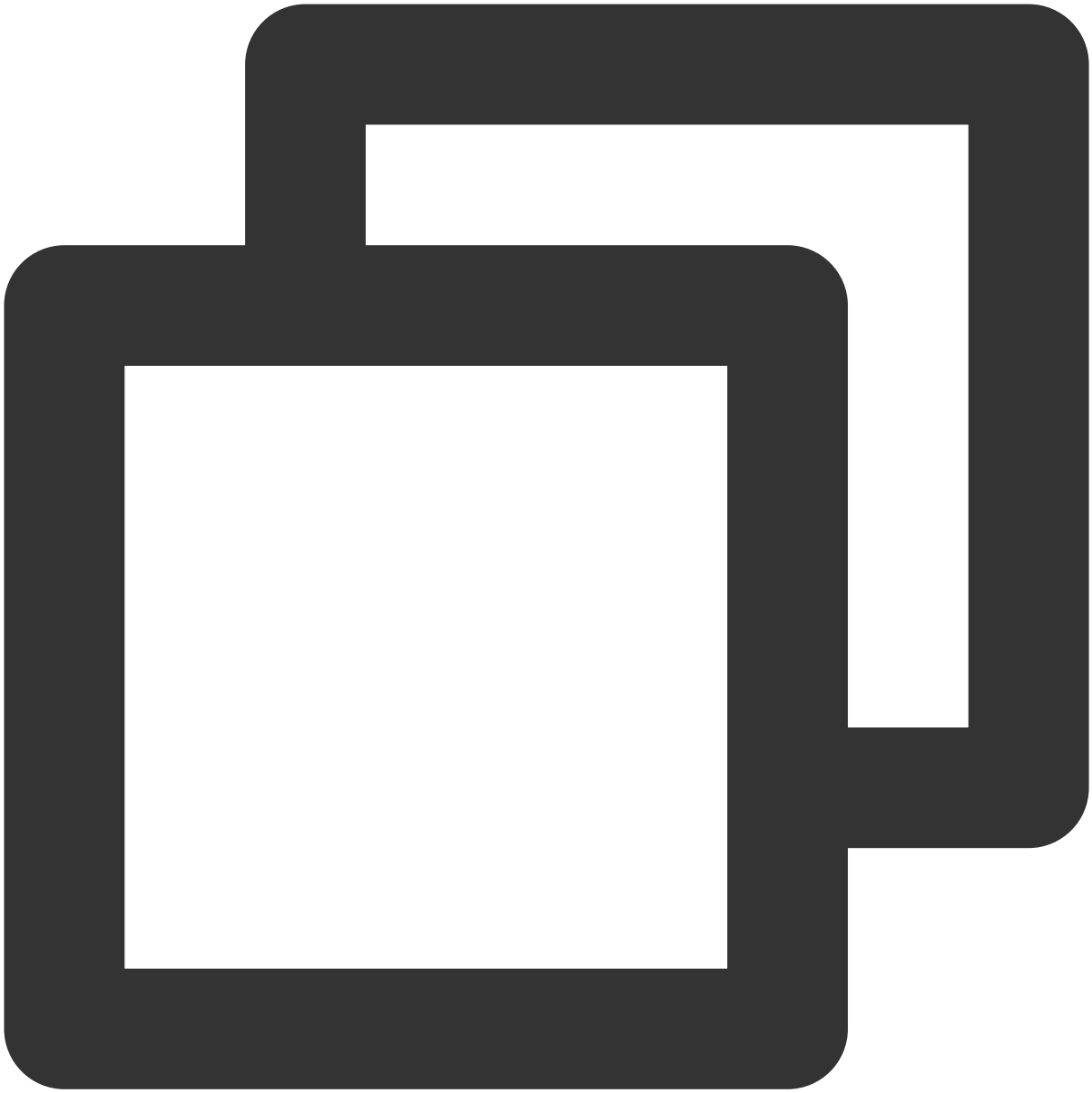
```
                isMute:(BOOL)isMute
NS_SWIFT_NAME(onSeatMute(index:isMute:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
index	int	操作するマイク。
isMute	boolean	true：マイクミュート； false：ミュート解除。

onSeatClose

管理者のマイククローズ。



```
- (void)onSeatClose:(NSInteger)index
            isClose:(BOOL)isClose
NS_SWIFT_NAME(onSeatClose(index:isClose:));
```

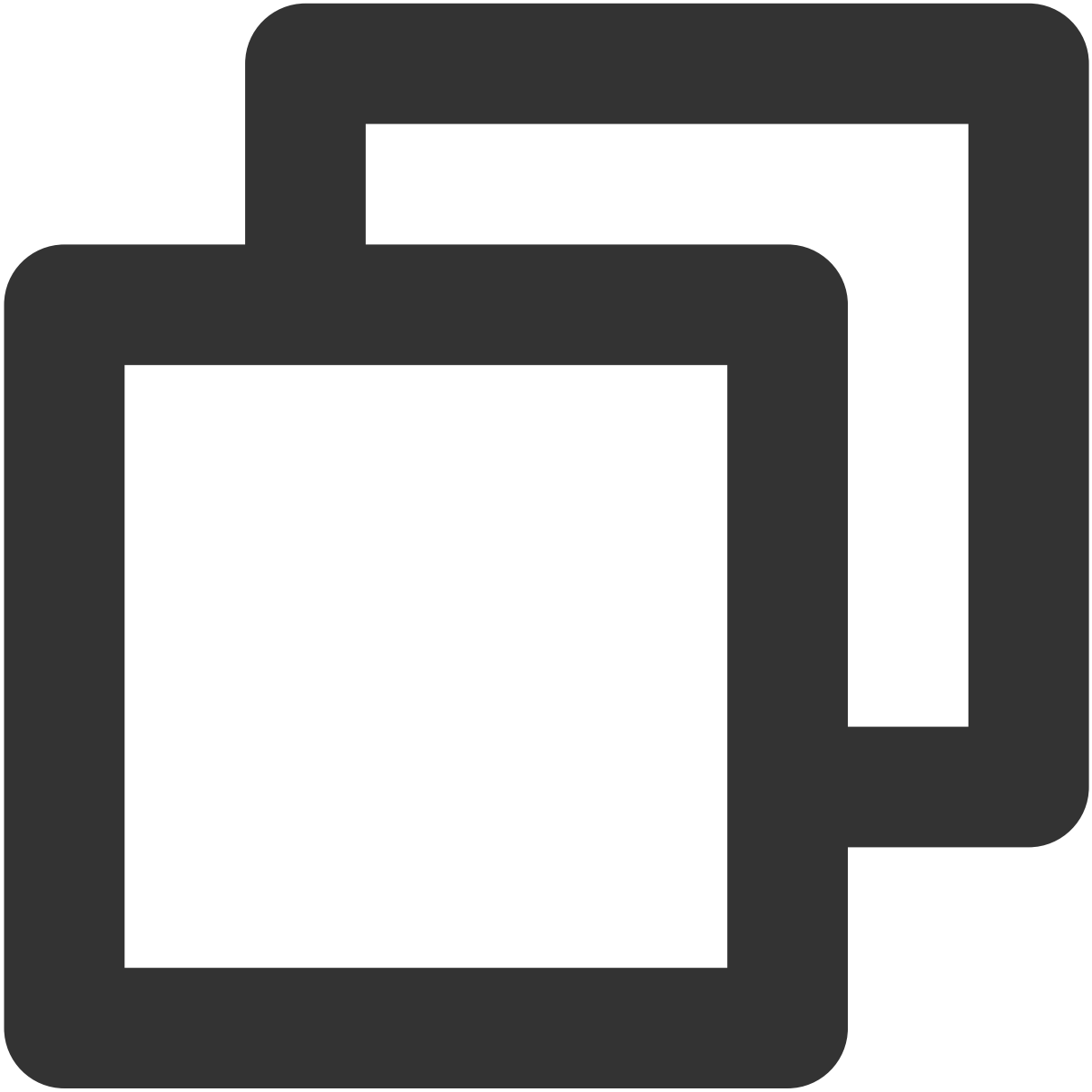
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
index	int	操作するマイク。
isClose	boolean	true：マイクのクローズ； false：マイクのクローズ解除。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。



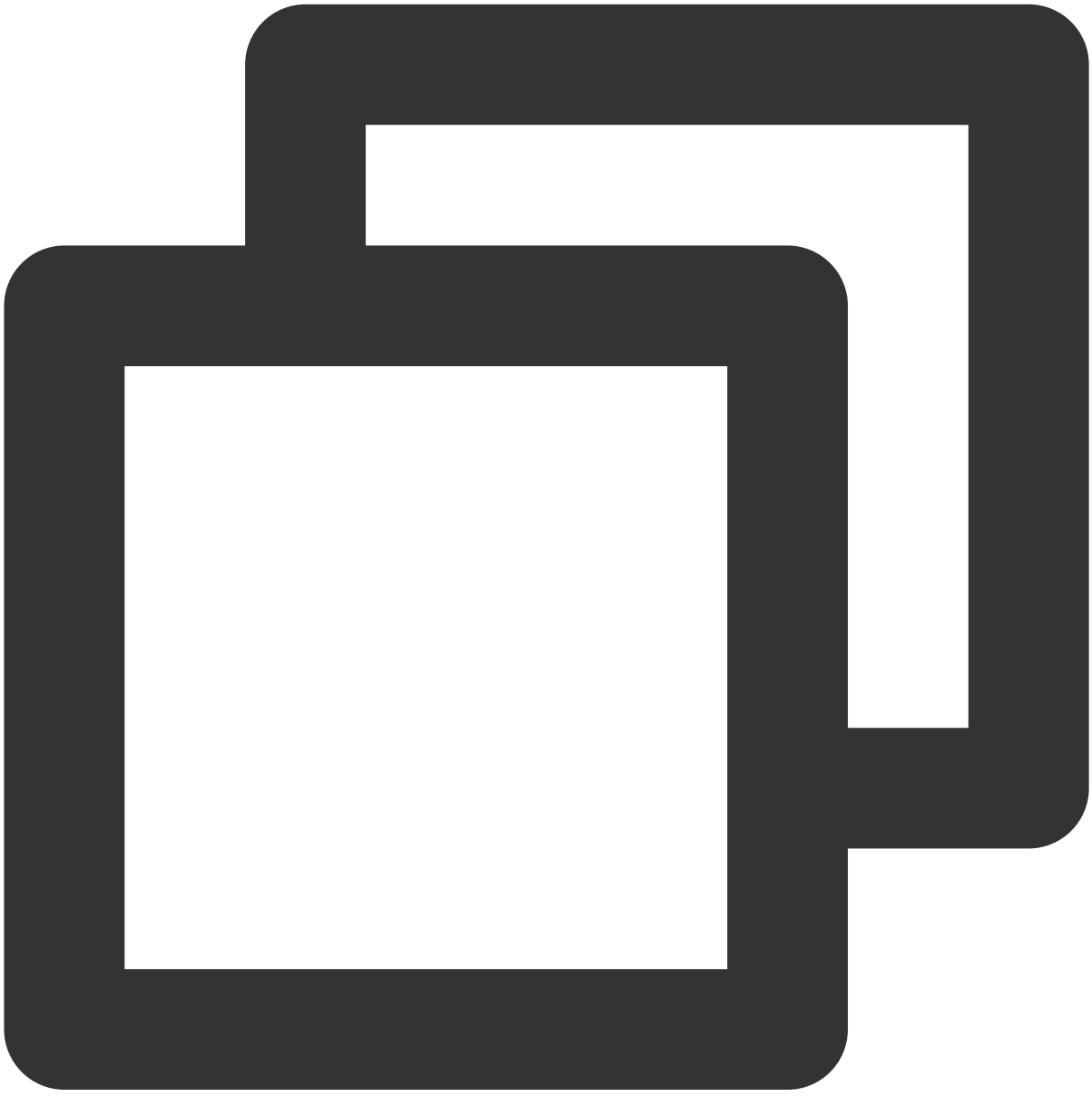
```
- (void)onAudienceEnter:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceEnter(userInfo:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userInfo	UserInfo	入室したリスナーの情報。

onAudienceExit

リスナー退室通知の受信。



```
- (void)onAudienceExit:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onAudienceExit(userInfo:));
```

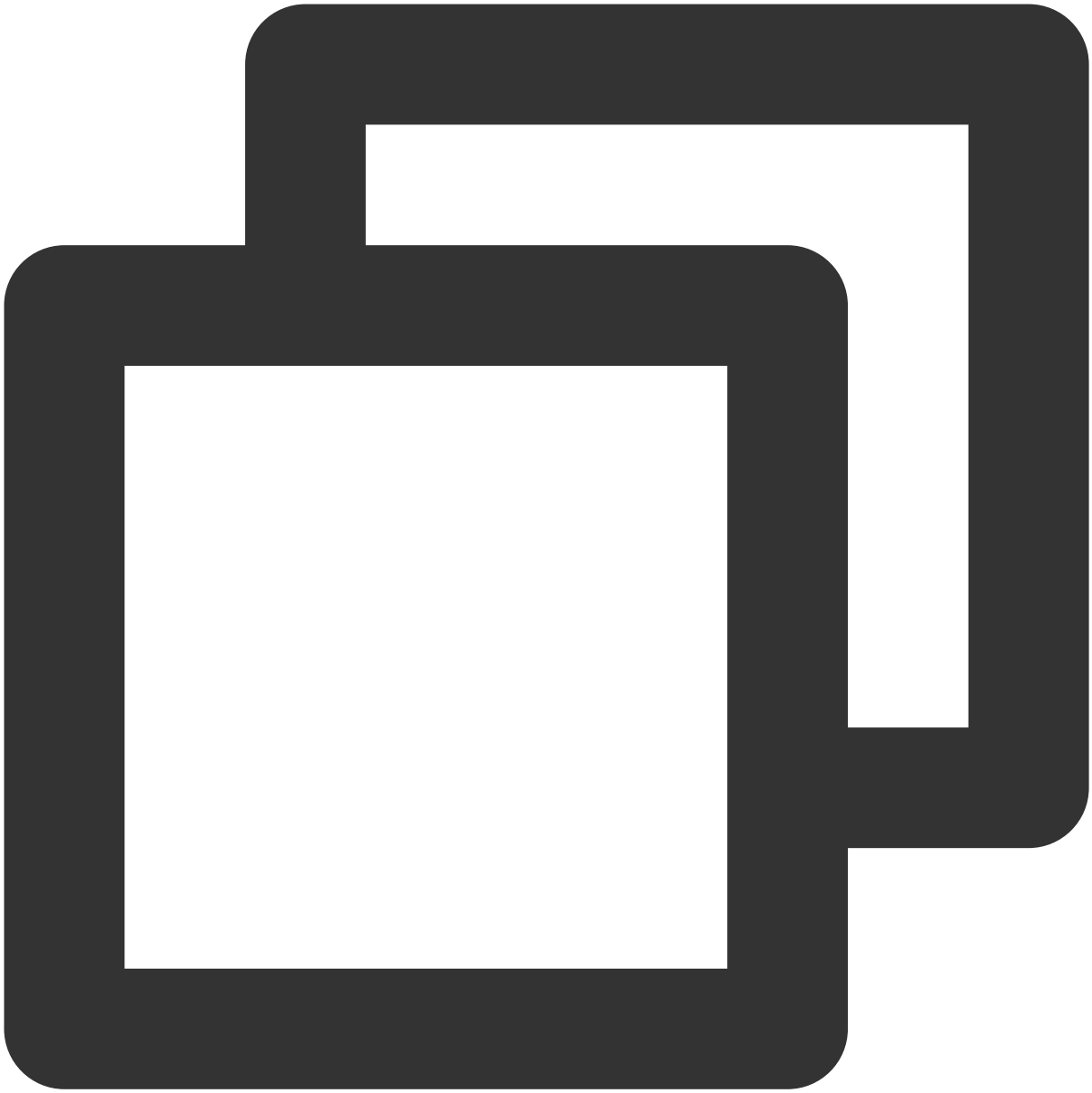
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userInfo	UserInfo	退室したリスナーの情報。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージを受信します。



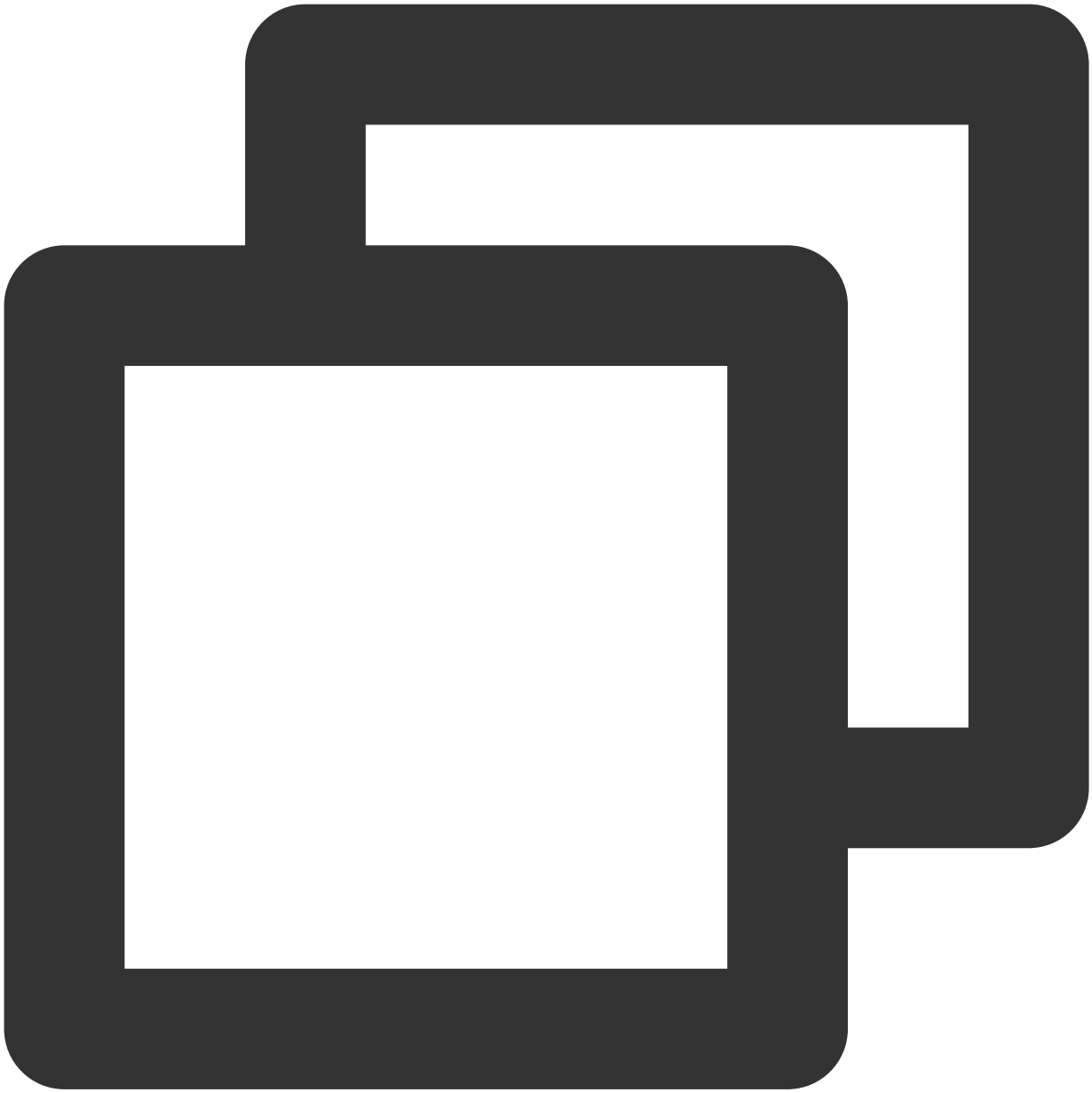
```
- (void)onRecvRoomTextMsg:(NSString *)message
                 userInfo:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomTextMsg(message:userInfo:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージを受信します。



```
- (void)onRecvRoomCustomMsg:(NSString *)cmd
                    message:(NSString  *)message
                   userInfo:(KaraokeUserInfo *)userInfo
NS_SWIFT_NAME(onRecvRoomCustomMsg(cmd:message:userInfo:));
```

パラメータは下表に示すとおりです：

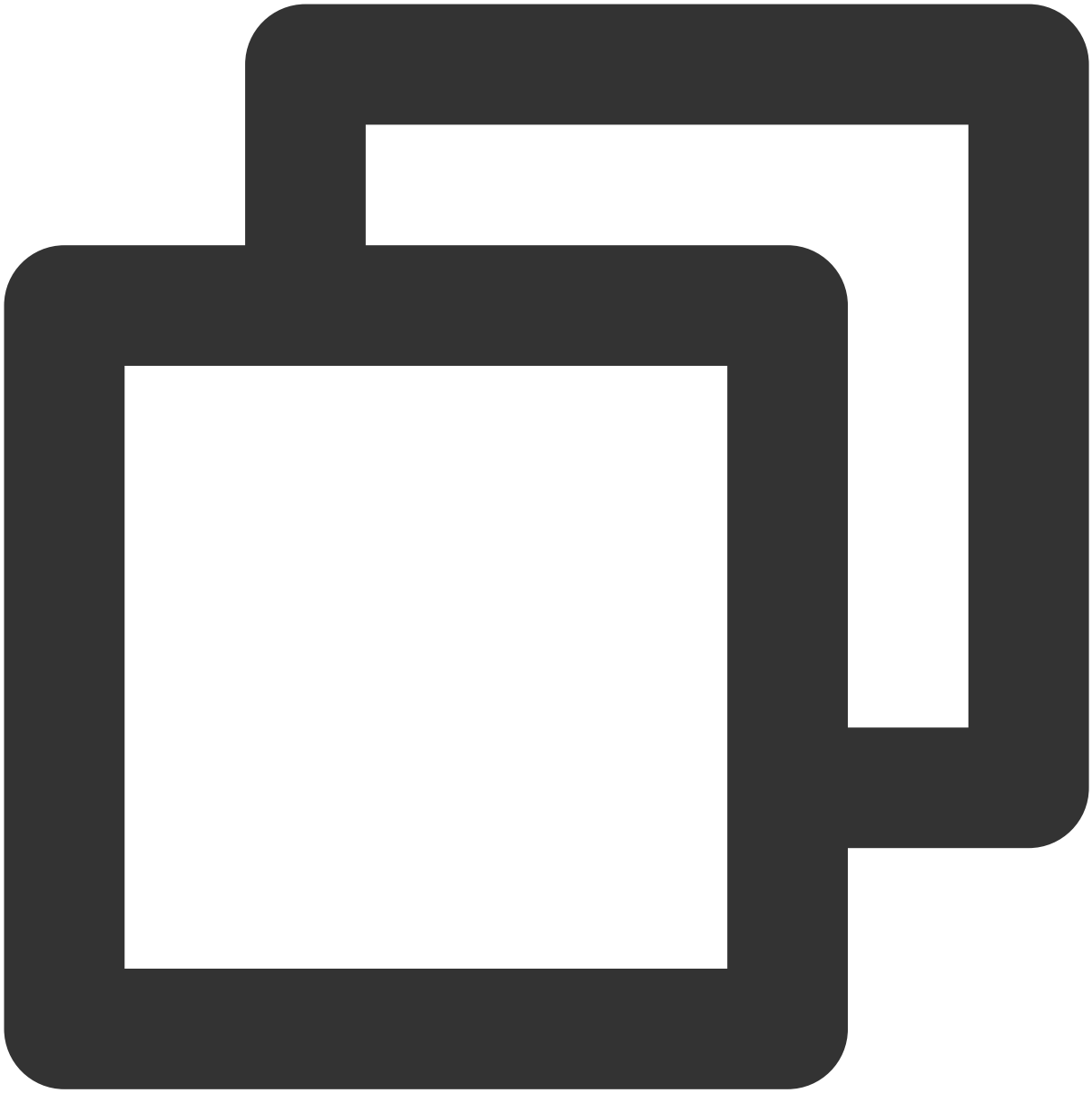
パラメー	タイプ	意味

タ		
command	String	コマンドワードです。開発者がカスタマイズするもので、主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

招待シグナリングイベントのコールバック

onReceiveNewInvitation

新規招待リクエストの受信。



```
- (void)onReceiveNewInvitation:(NSString *)identifier
                       inviter:(NSString *)inviter
                           cmd:(NSString *)cmd
                       content:(NSString *)content
NS_SWIFT_NAME(onReceiveNewInvitation(identifier:inviter:cmd:content:));
```

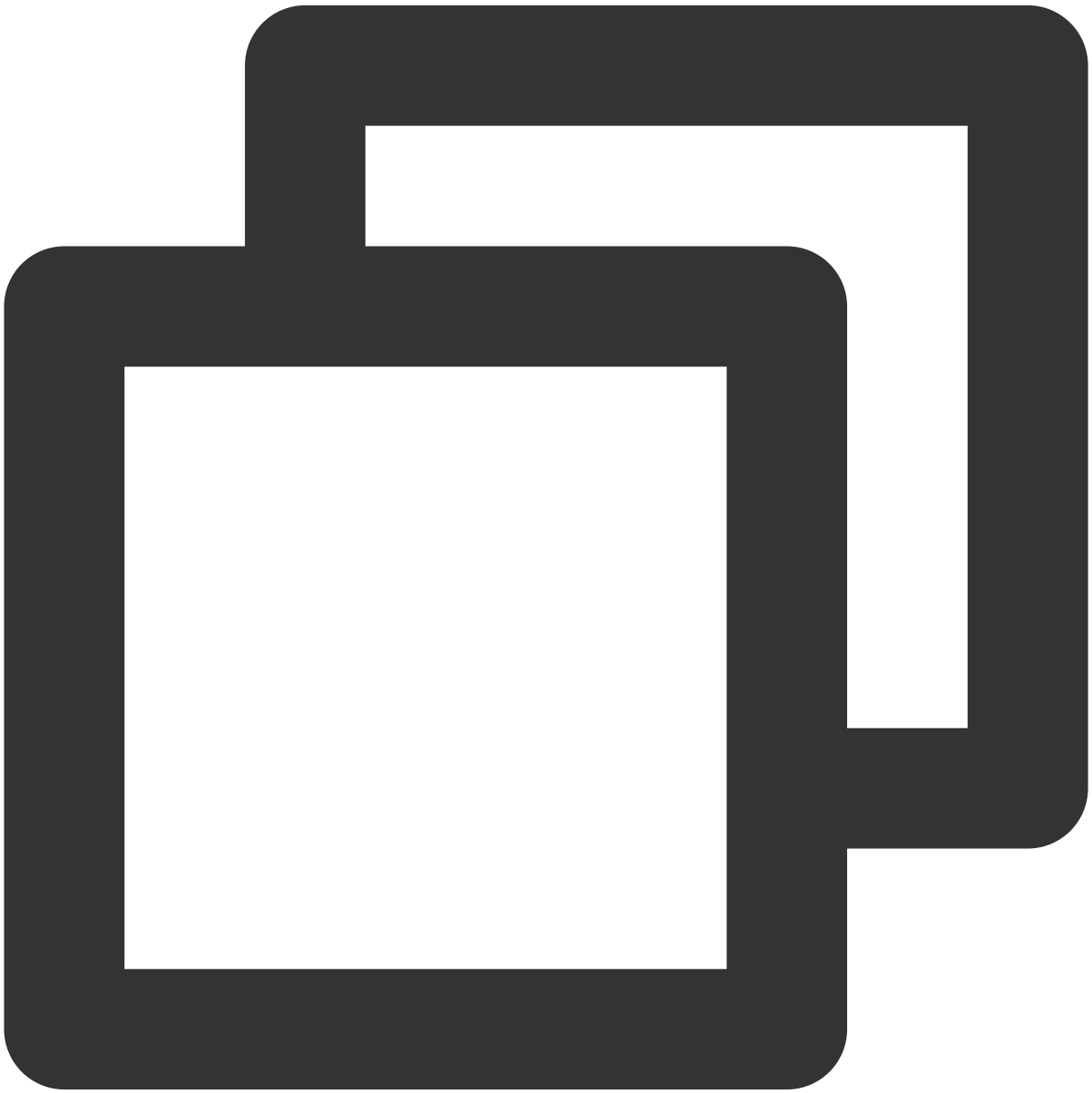
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。

inviter	String	招待者のユーザーID。
cmd	String	開発者がカスタマイズする業務指定のコマンドワードです。
content	UserInfo	業務指定のコンテンツ。

onInviteeAccepted

被招待者が招待に同意。



```
- (void)onInviteeAccepted:(NSString *)identifier
```

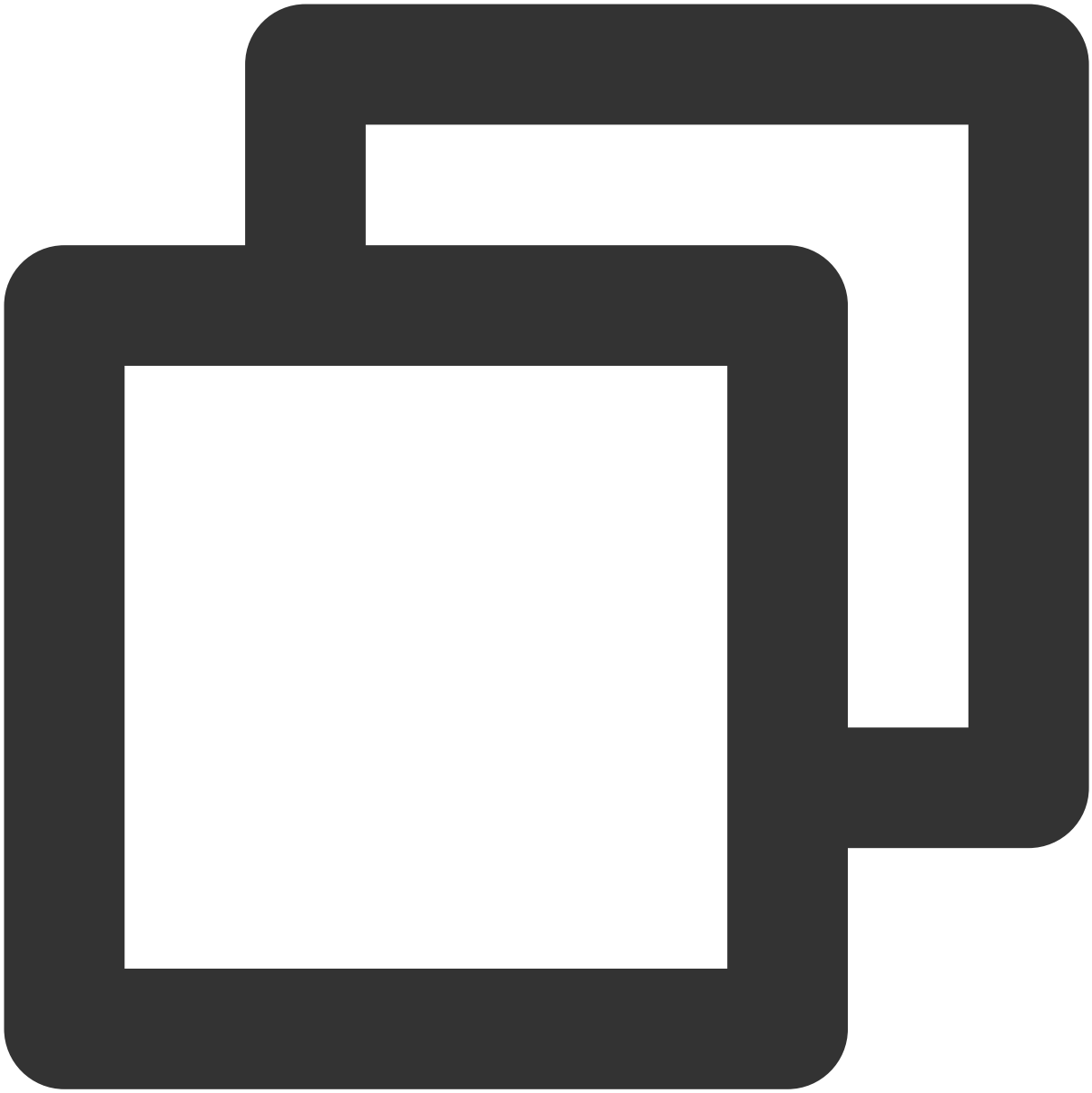
```
                    invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeAccepted(identifier:invitee:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInviteeRejected

被招待者による招待の拒否。



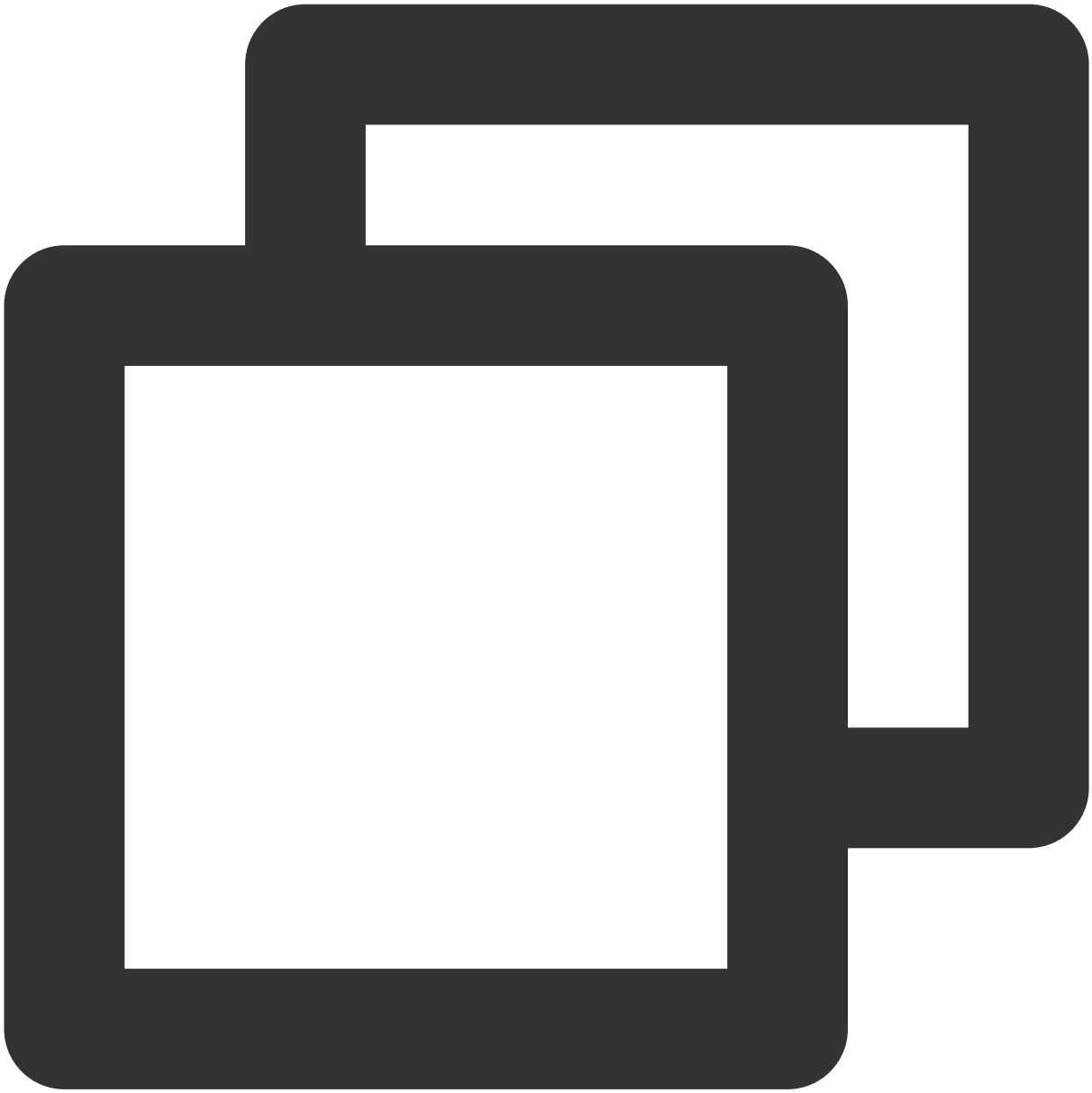
```
- (void)onInviteeRejected:(NSString *)identifier
                  invitee:(NSString *)invitee
NS_SWIFT_NAME(onInviteeRejected(identifier:invitee:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInvitationCancelled

招待者が招待を取り消し。



```
- (void)onInvitationCancelled:(NSString *)identifier
                      invitee:(NSString *)invitee NS_SWIFT_NAME(onInvitationCancell
```

パラメータは下表に示すとおりです：

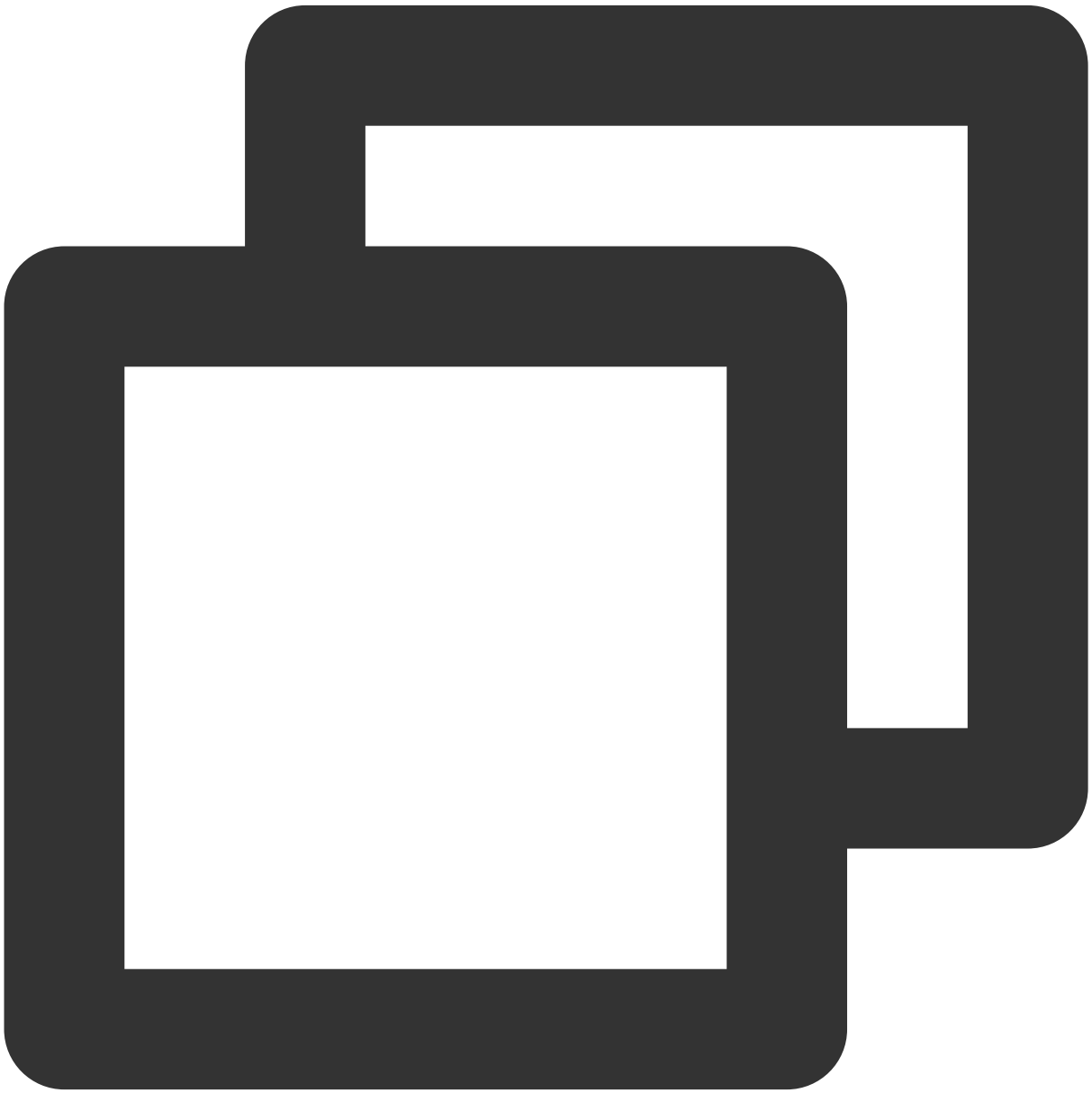
パラメータ	タイプ	意味
id	String	招待ID。

invitee	String	招待者のユーザーID。

音楽再生ステータスコールバック

onMusicPrepareToPlay

音楽再生準備のコールバック



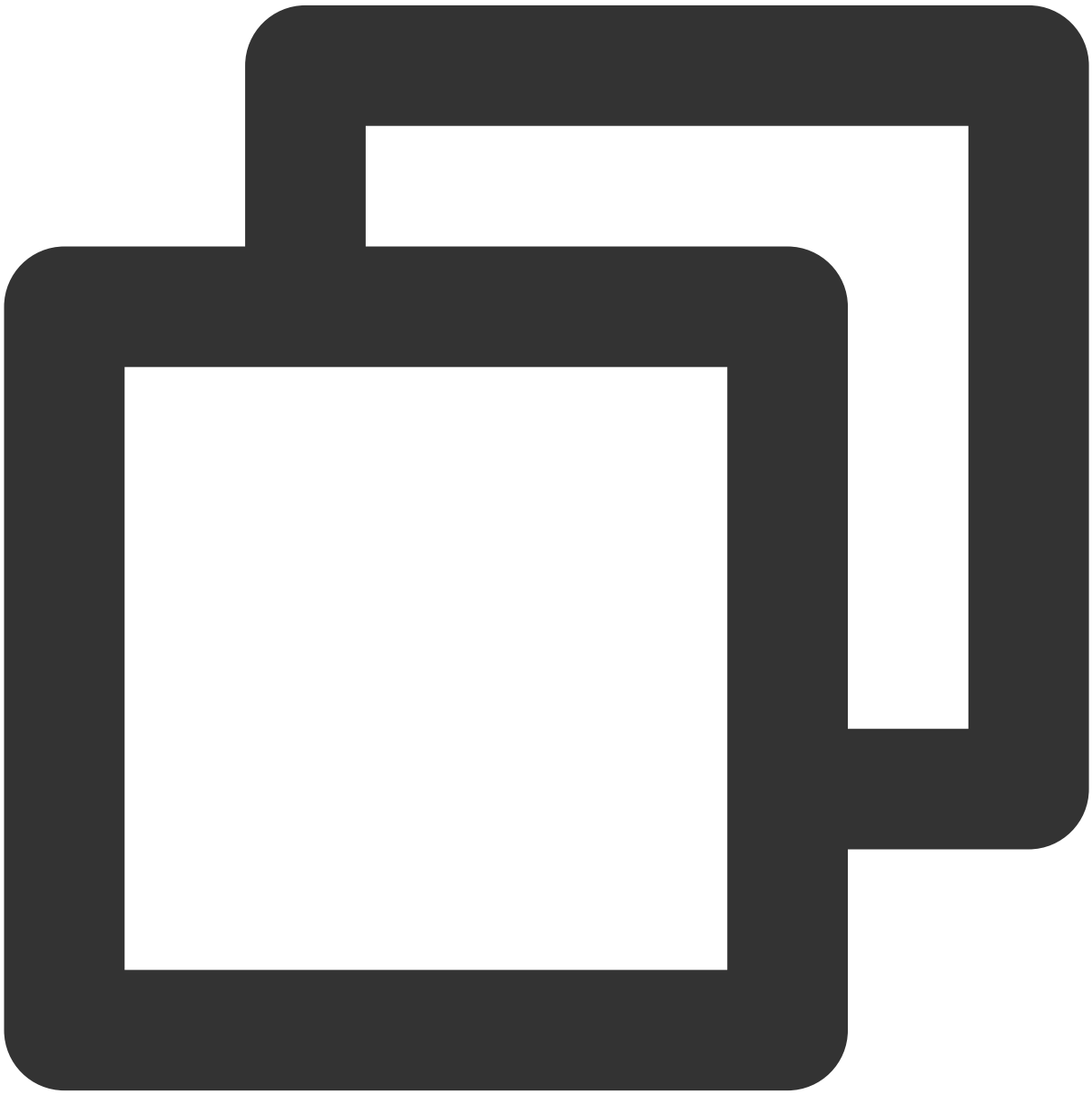
```
- (void)onMusicPrepareToPlay:(int32_t)musicID
NS_SWIFT_NAME(onMusicPrepareToPlay(musicID:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int32_t	再生時に渡されたmusicID。

onMusicProgressUpdate

楽曲再生進捗度のコールバック



```
- (void)onMusicProgressUpdate:(int32_t)musicID
```

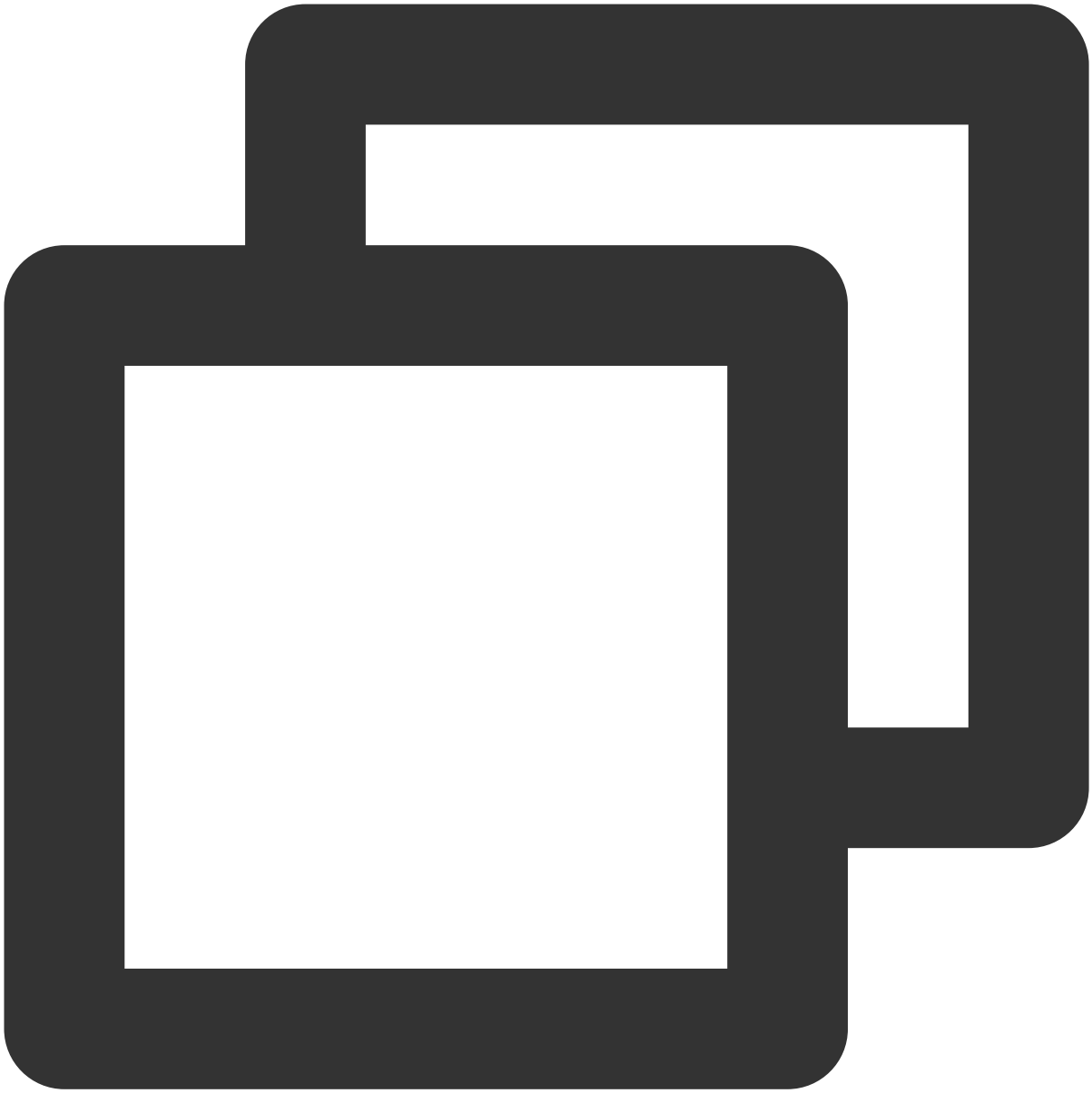
```
                    progress:(NSInteger)progress total:(NSInteger)total
NS_SWIFT_NAME(onMusicProgressUpdate(musicID:progress:total:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int32_t	再生時に渡されたmusicID。
progress	NSInteger	現在の再生時間。単位： ms。
total	NSInteger	合計時間。単位： ms。

onMusicCompletePlaying

音楽再生完了のコールバック



```
- (void)onMusicCompletePlaying:(int32_t)musicID
NS_SWIFT_NAME(onMusicCompletePlaying(musicID:));
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int32_t	再生時に渡されたmusicID。

TRTCKaraoke(Android)

最終更新日：：2024-07-19 15:32:54

TRTCKaraokeRoomは、Tencent CloudのTRTCおよびIMサービスを基に組み合わせたコンポーネントで、以下の機能をサポートしています。
管理者が作成した新しいKaraokeルームが配信を開始すると、リスナーはKaraokeルームに入室して聴取/インタラクションを行います。
管理者は、楽曲の順序を管理し、マイク・オンのキャスターをキックアウトすることもできます。
管理者はまた、座席をクローズすることができ、その他のリスナーはマイク・オンを申請することができなくなります。
リスナーはマイク・オンを申請して、マイク・オンのキャスターになり、マイク・オン後は楽曲の選択や歌唱ができるようになります。また、いつでもマイク・オフにして、通常のリスナーになることも可能です。
ギフトや各種のテキストメッセージ、カスタムメッセージの送信をサポートします。カスタムメッセージを弾幕、「いいね」などを実装するために使用することができます。
説明：
TUIKitシリーズコンポーネントはTencent CloudのTRTCとIMという2つの基本的なPaaSサービスを同時に使用し、TRTCをアクティブにした後、IMサービスを同期的にアクティブにすることができます。IMサービスの課金ルールの詳細については、Instant Messagingの料金説明をご参照ください。TRTCをアクティブにすると、デフォルトでは、100DAUまでサポートするIM SDK体験版もアクティブになります。
TRTCKaraokeRoomはオープンソースのClassであり、Tencent Cloudの2つのクローズドソースのSDKに依存しています。具体的な実装プロセスについては、Karaoke (Android)をご参照ください。
TRTC SDK：TRTC SDKを低遅延のボイスチャットコンポーネントとして使用しています。
IM SDK：IM SDKのAVChatroomを使用してチャットルーム機能を実装します。同時にIMの属性インターフェースによって、マイクリストなどのルーム情報を保存し、招待シグナリングはマイク・オン/ピックのリクエストに用いることができます。

TRTCKaraokeRoom API概要

SDK基本関数

API	説明
sharedInstance	シングルトンオブジェクトを取得します。
destroySharedInstance	シングルトンオブジェクトを廃棄します。
setDelegate	イベントコールバックを設定します。
setDelegateHandler	イベントのコールバックが配置されているスレッドを設定します。

login	ログイン。
logout	ログアウト。
setSelfProfile	個人情報を修正します。

ルーム関連インターフェース関数

API	説明
createRoom	ルームの作成（管理者が呼び出し）。ルームが存在しない場合は、システムが新しいルームを自動的に作成します。
destroyRoom	ルームの破棄（管理者が呼び出し）。
enterRoom	入室（リスナーが呼び出し）。
exitRoom	退室（リスナーが呼び出し）。
getRoomInfoList	ルームリストの詳細情報を取得します。
getUserInfoList	指定されたuserIdのユーザー情報を取得します。 nullの場合は、ルーム内全員の情報を取得します。

音楽再生インターフェース

API	説明
startPlayMusic	音楽の再生を開始します。
stopPlayMusic	音楽の再生を停止します。
pausePlayMusic	音楽の再生を一時停止します。
resumePlayMusic	音楽の再生を再開します。

マイク管理インターフェース

API	説明
enterSeat	ユーザーが発言者になる（リスナー側/管理者ともに呼び出し可）。
leaveSeat	ユーザーが視聴者になる（キャスターが呼び出し）。
pickSeat	視聴者が発言できるように招待（管理者が呼び出し）。

kickSeat	キックアウトしてマイク・オフ（管理者が呼び出し）。
muteSeat	任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
closeSeat	任意のマイクのクローズ/解除（管理者が呼び出し）。

ローカルのオーディオ操作インターフェース

API	説明
startMicrophone	マイクの集音開始。
stopMicrophone	マイクの集音停止。
setAudioQuality	音質の設定。
muteLocalAudio	ローカルオーディオミュートの開始/停止。
setSpeaker	スピーカーの起動設定。
setAudioCaptureVolume	マイクの集音音量設定。
setAudioPlayoutVolume	再生音量の設定。
setVoiceEarMonitorEnable	インイヤーモニタリングのオン/オフ。

リモートユーザー音声操作インターフェース

API	説明
muteRemoteAudio	指定メンバーをミュート/ミュート解除。
muteAllRemoteAudio	全メンバーをミュート/ミュート解除。

BGMサウンドエフェクト関連インターフェース

API	説明
getAudioEffectManager	BGMサウンドエフェクト管理オブジェクトTXAudioEffectManagerを取得します。

メッセージ送信関連インターフェース

API	説明
sendRoomTextMsg	ルーム内でのテキストメッセージのブロードキャスト。通常、弾幕によるチャットに使用します。

sendRoomCustomMsg	カスタマイズしたテキストメッセージを送信します。

招待シグナリング関連インターフェース

API	説明
sendInvitation	ユーザーに招待を送信。
acceptInvitation	招待の同意。
rejectInvitation	招待の辞退。
cancelInvitation	招待の取り消し。

TRTCKaraokeRoomDelegate API概要

一般的なイベントコールバック

API	説明
onError	エラーのコールバック。
onWarning	警告のコールバック。
onDebugLog	Logコールバック。

ルームイベントのコールバック

API	説明
onRoomDestroy	ルームが廃棄された時のコールバック。
onRoomInfoChange	Karaoke情報変更のコールバック。
onUserVolumeUpdate	ユーザー通話音量のコールバック。

マイク変更コールバック

API	説明
onSeatListChange	全量のマイクリストの変更。
onAnchorEnterSeat	発言者のメンバーがいます（ユーザーが発言者になります/管理者が視聴者を発言できるように招待）。

onAnchorLeaveSeat	視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。
onSeatMute	管理者のマイクミュート。
onUserMicrophoneMute	ユーザーのマイクがミュートされているかどうか。
onSeatClose	管理者のマイククローズ。

リスナーの入退室イベントのコールバック

API	説明
onAudienceEnter	リスナー入室通知の受信。
onAudienceExit	リスナー退室通知の受信。

メッセージイベントのコールバック

API	説明
onRecvRoomTextMsg	テキストメッセージを受信します。
onRecvRoomCustomMsg	カスタムメッセージを受信します。

シグナリングイベントのコールバック

API	説明
onReceiveNewInvitation	新規招待リクエストの受信。
onInviteeAccepted	被招待者が招待に同意。
onInviteeRejected	被招待者が招待を拒否。
onInvitationCancelled	招待者が招待を取り消し。

楽曲イベントコールバック

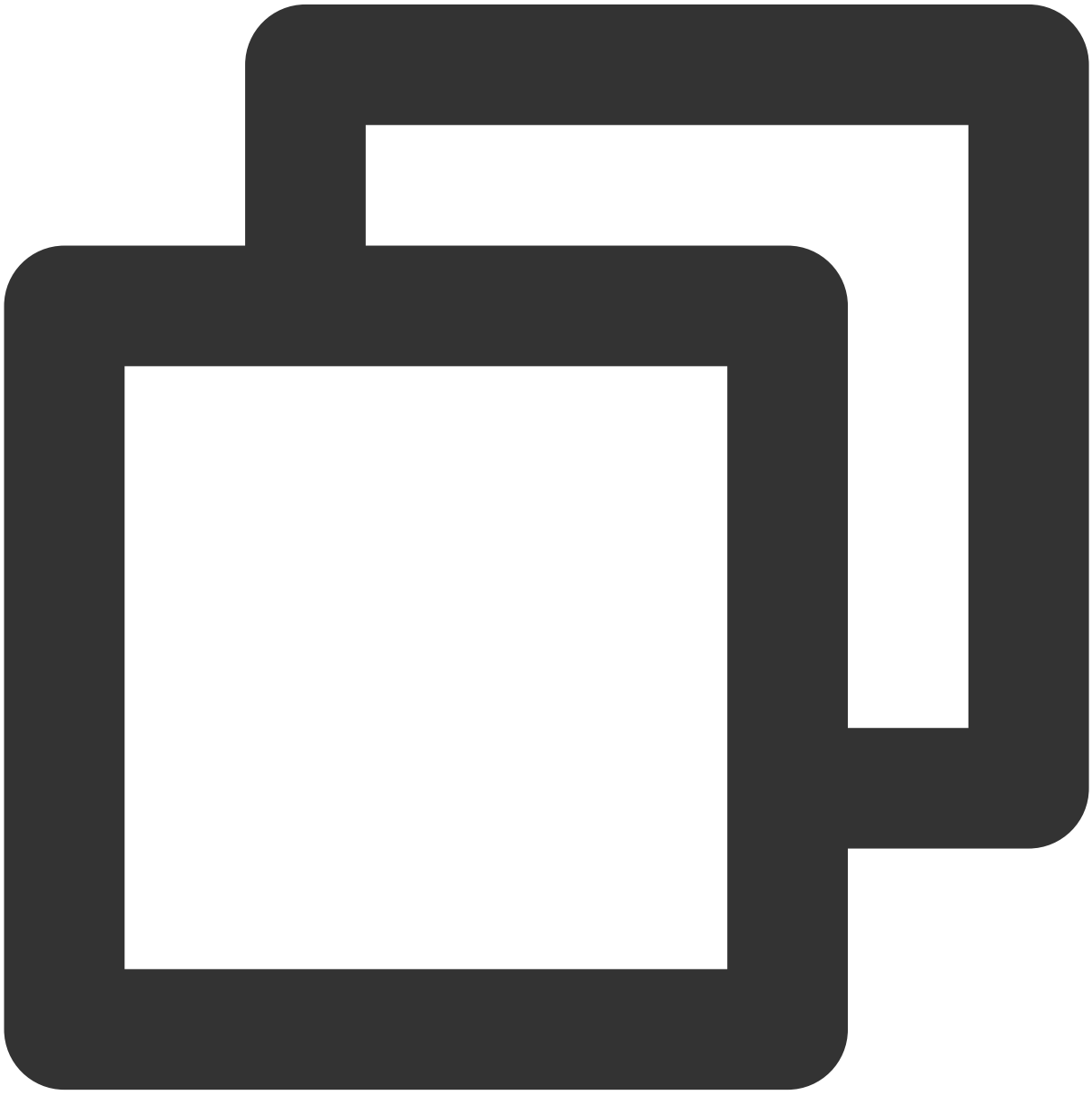
API	説明
onMusicProgressUpdate	楽曲再生進捗度のコールバック。

onMusicPrepareToPlay	音楽再生準備のコールバック。
onMusicCompletePlaying	音楽再生完了のコールバック。

SDK基本関数

sharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを取得します。



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

パラメータは下表に示すとおりです：

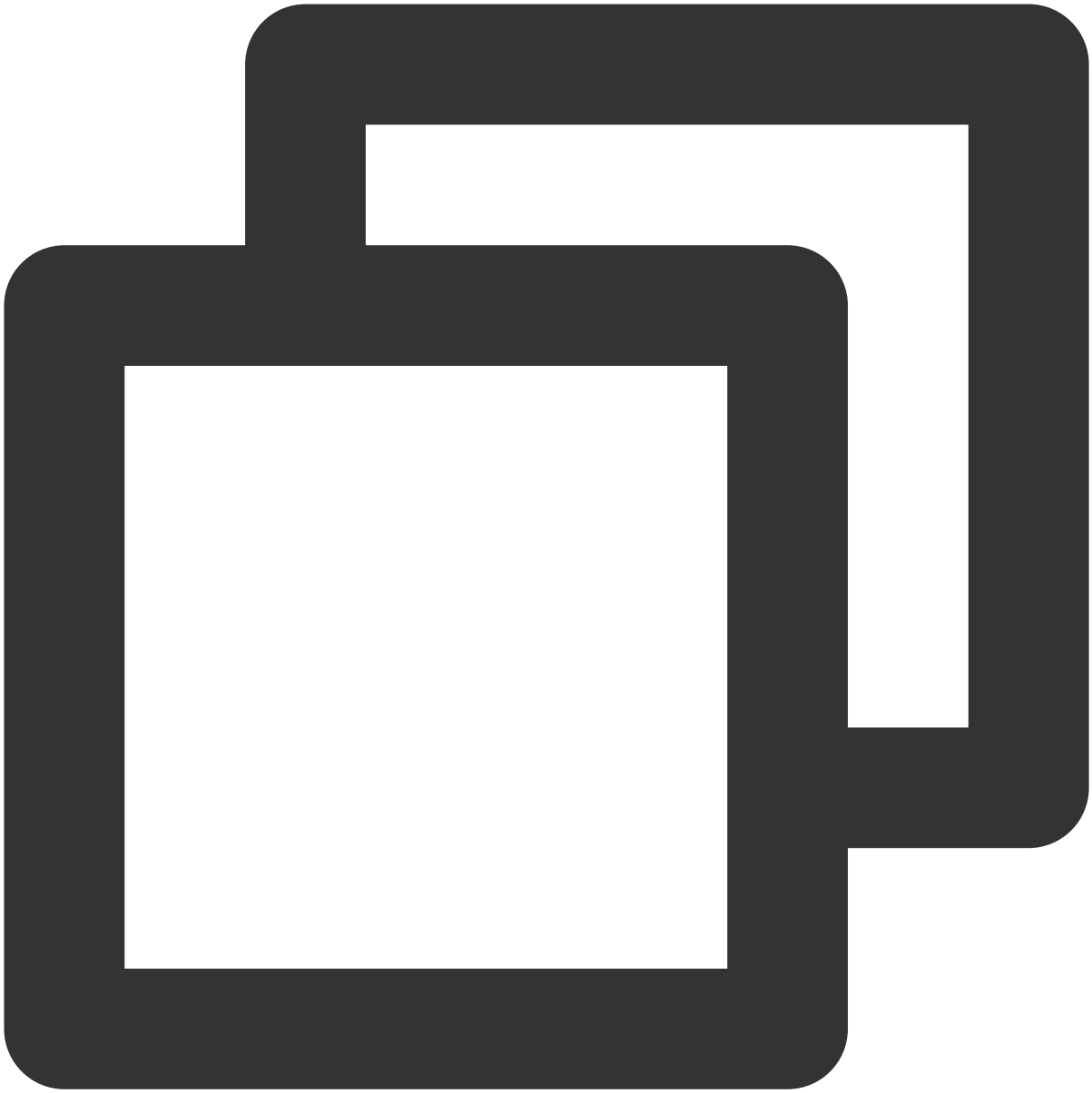
パラメータ	タイプ	意味
context	Context	Androidコンテキスト。内部ではApplicationContextに変換してシステムAPIの呼び出しに使用します

destroySharedInstance

TRTCKaraokeRoomシングルトンオブジェクトを破棄します。

説明：

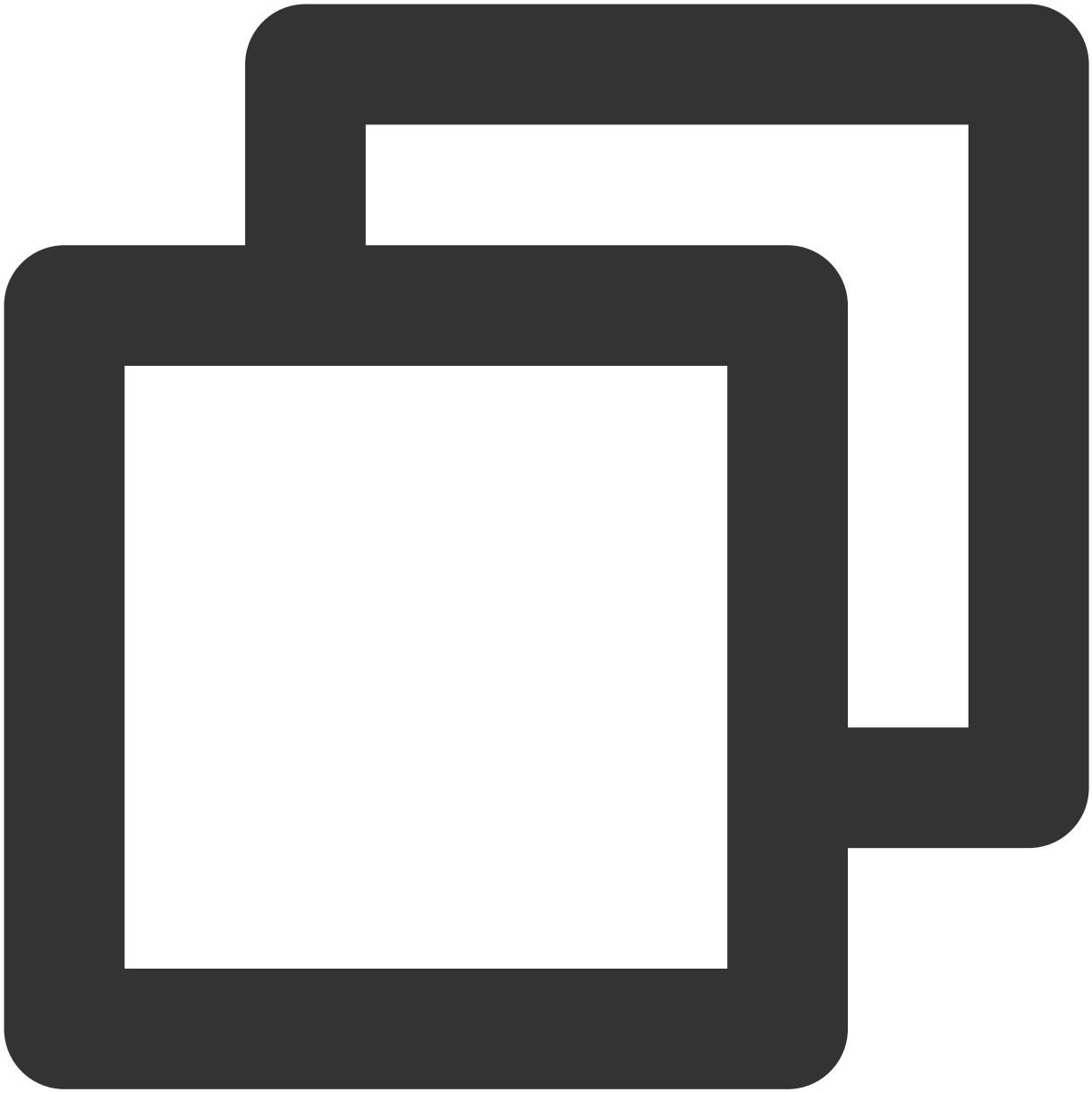
インスタンスの破棄後は、外部にキャッシュされたTRTCKaraokeRoomインスタンスの再使用ができません。改めてsharedInstanceを呼び出し、インスタンスを新規取得してください。



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

setDelegate

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



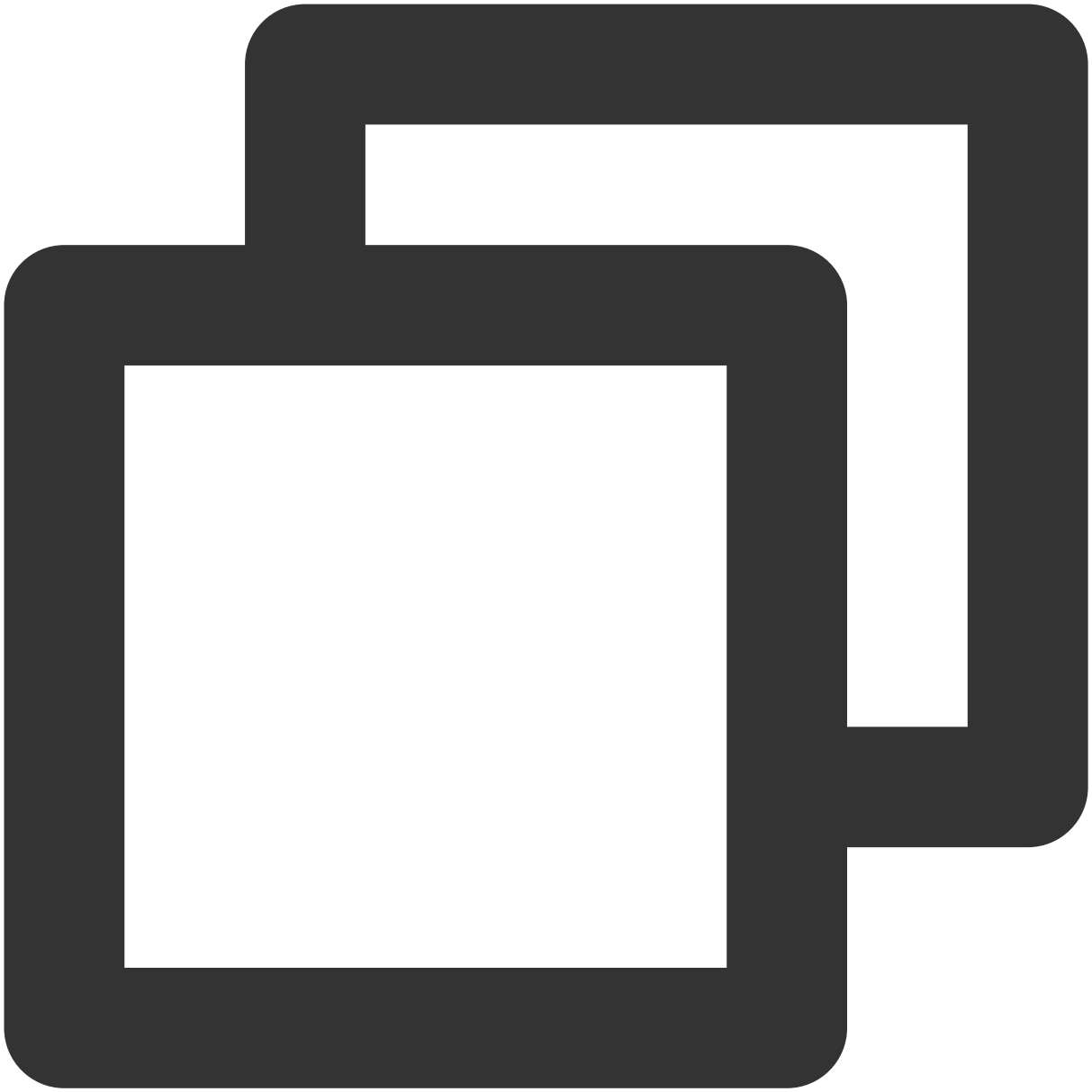
```
public abstract void setDelegate(TRTCKaraokeRoomDelegate delegate);
```

説明：

setDelegateはTRTCKaraokeRoomのプロキシコールバックです。

setDelegateHandler

イベントコールバックが配置されているスレッドを設定します。



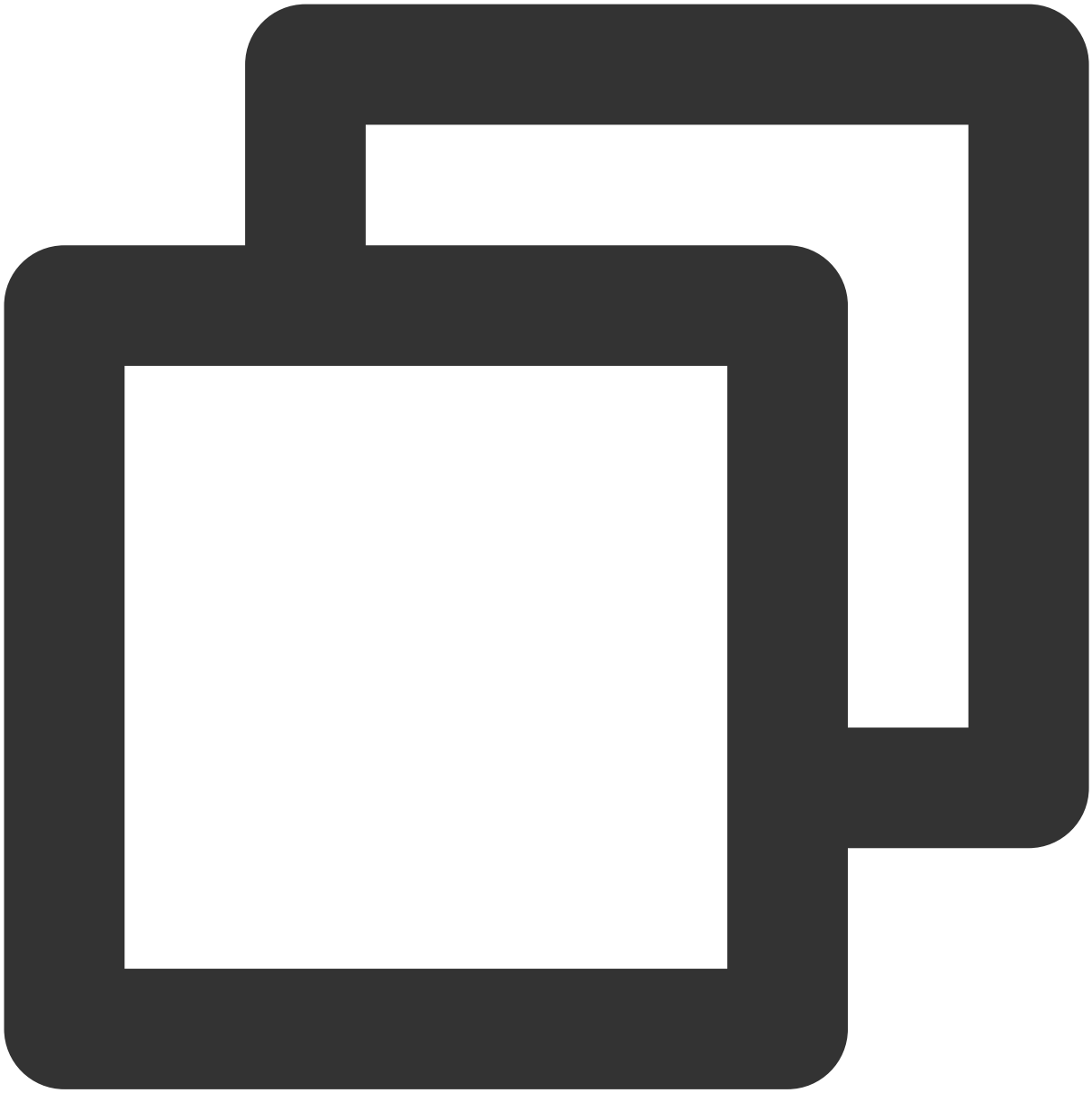
```
public abstract void setDelegateHandler(Handler handler);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
handler	Handler	TRTCKaraokeRoomの各種ステータス通知は、指定したhandlerスレッドに発信されます。

login

ログイン。



```
public abstract void login(int sdkAppId,
 String userId, String userSig,
TRTCKaraokeRoomCallback.ActionCallback callback);
```

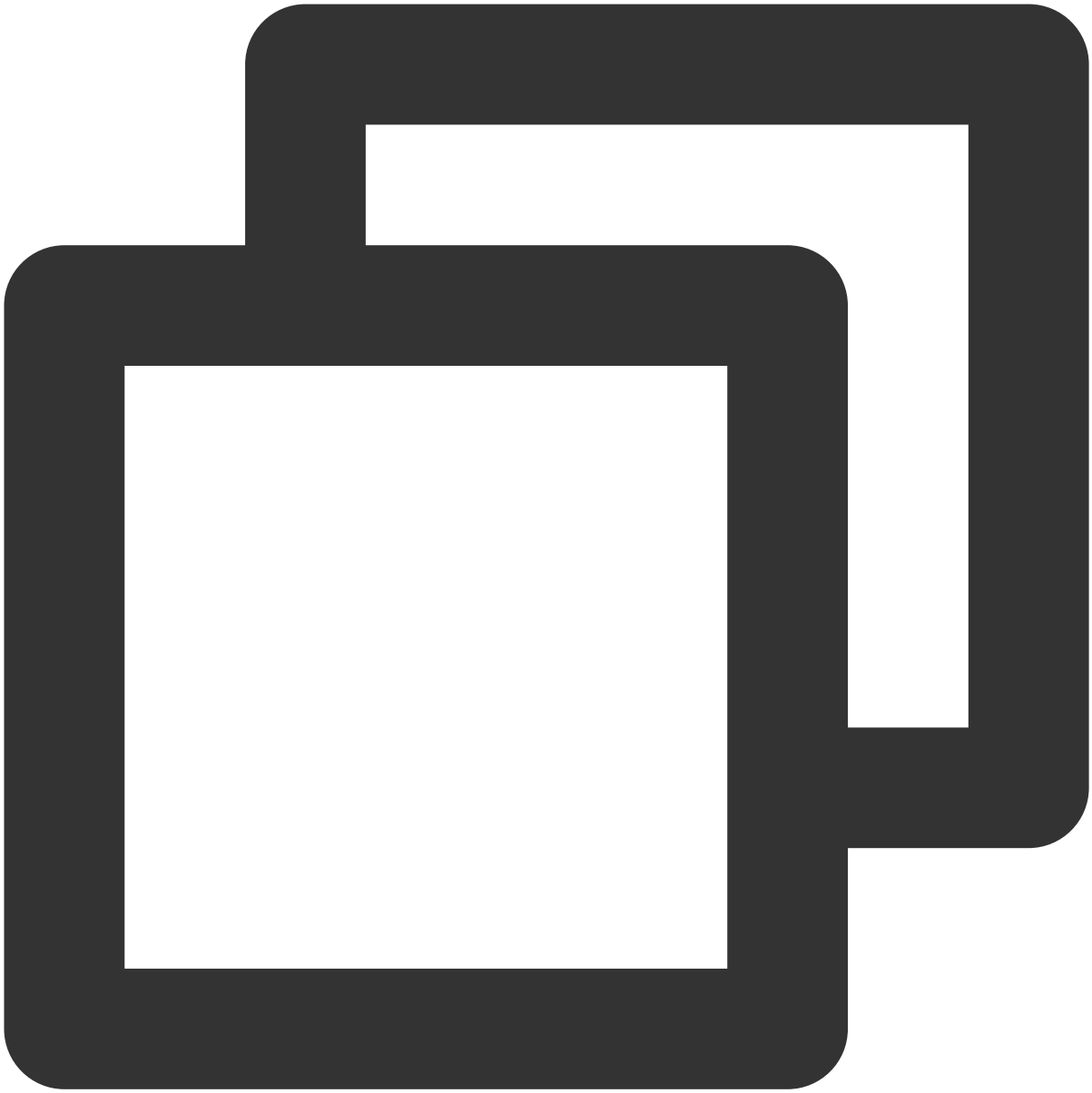
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

sdkAppId	int	TRTCコンソール> 【アプリケーション管理】>アプリケーション情報の中でSDKAppIDを確認できます。
userId	String	現在のユーザーID。文字列タイプであり、英語のアルファベット（a-zとA-Z）、数字（0-9）、ハイフン（-）とアンダーライン（_）のみ使用できます。
userSig	String	Tencent Cloudによって設計されたセキュリティ保護署名。取得方法については、 UserSigの計算、使用方法をご参照ください。
callback	ActionCallback	ログインのコールバック。成功時にcodeは0になります。

logout

ログアウト。



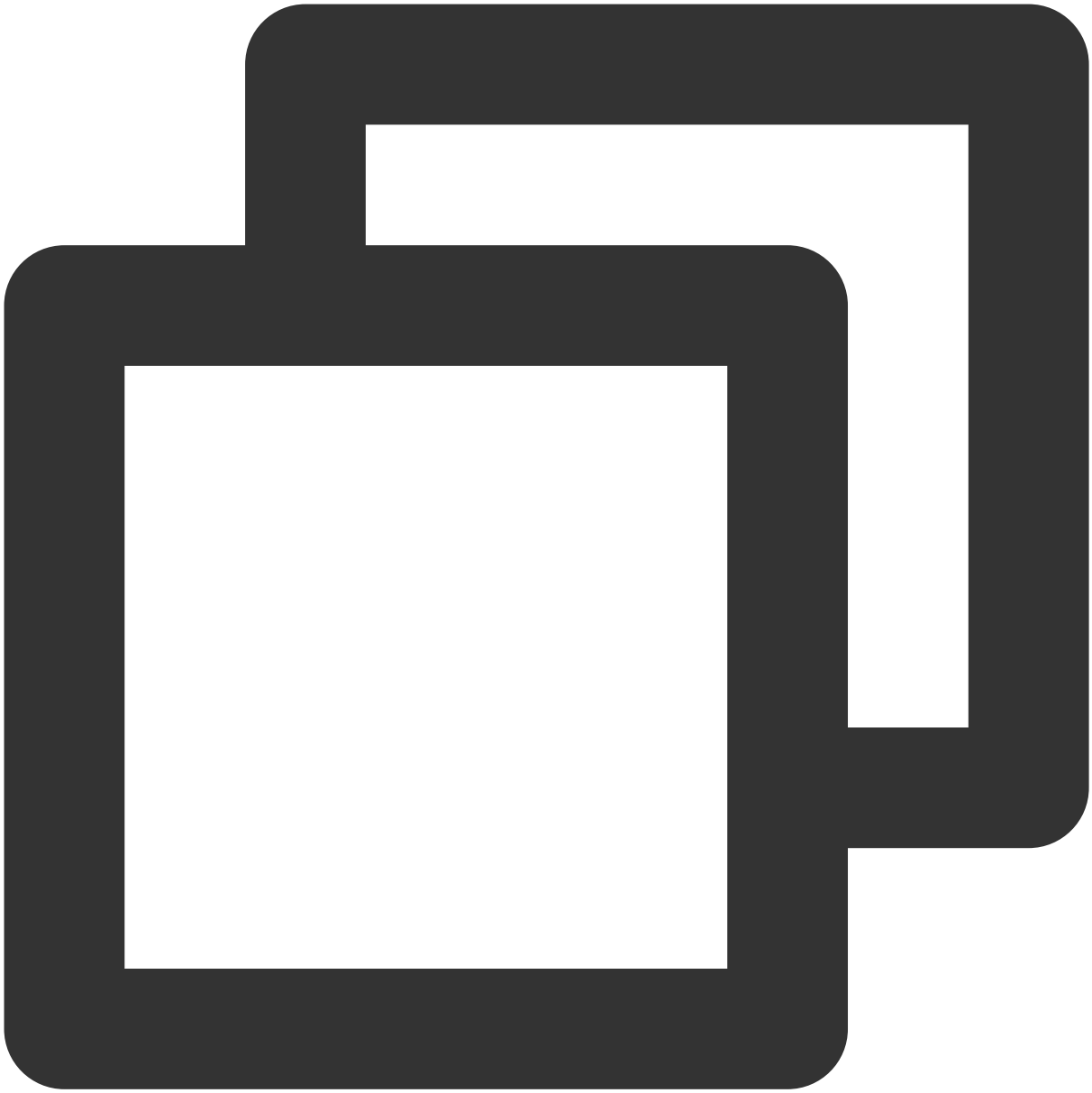
```
public abstract void logout(TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです：

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

setSelfProfile

個人情報の修正。



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

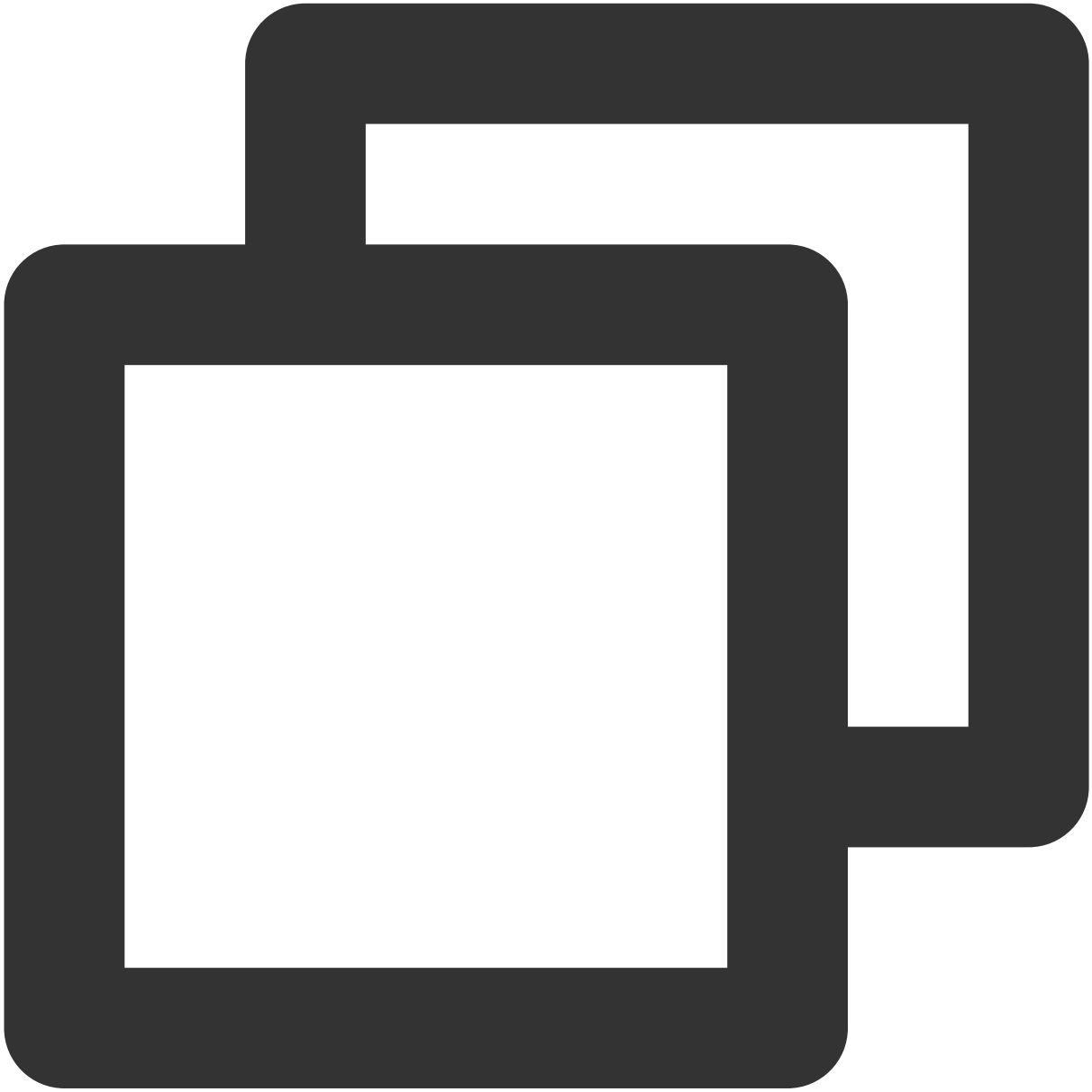
パラメータは下表に示すとおりです：

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

ルーム関連インターフェース関数

createRoom

ルームの作成（管理者が呼び出し）：



```
public abstract void createRoom(int roomId, TRTCKaraokeRoomDef.RoomParam roomParam,
```

パラメータは下表に示すとおりです。

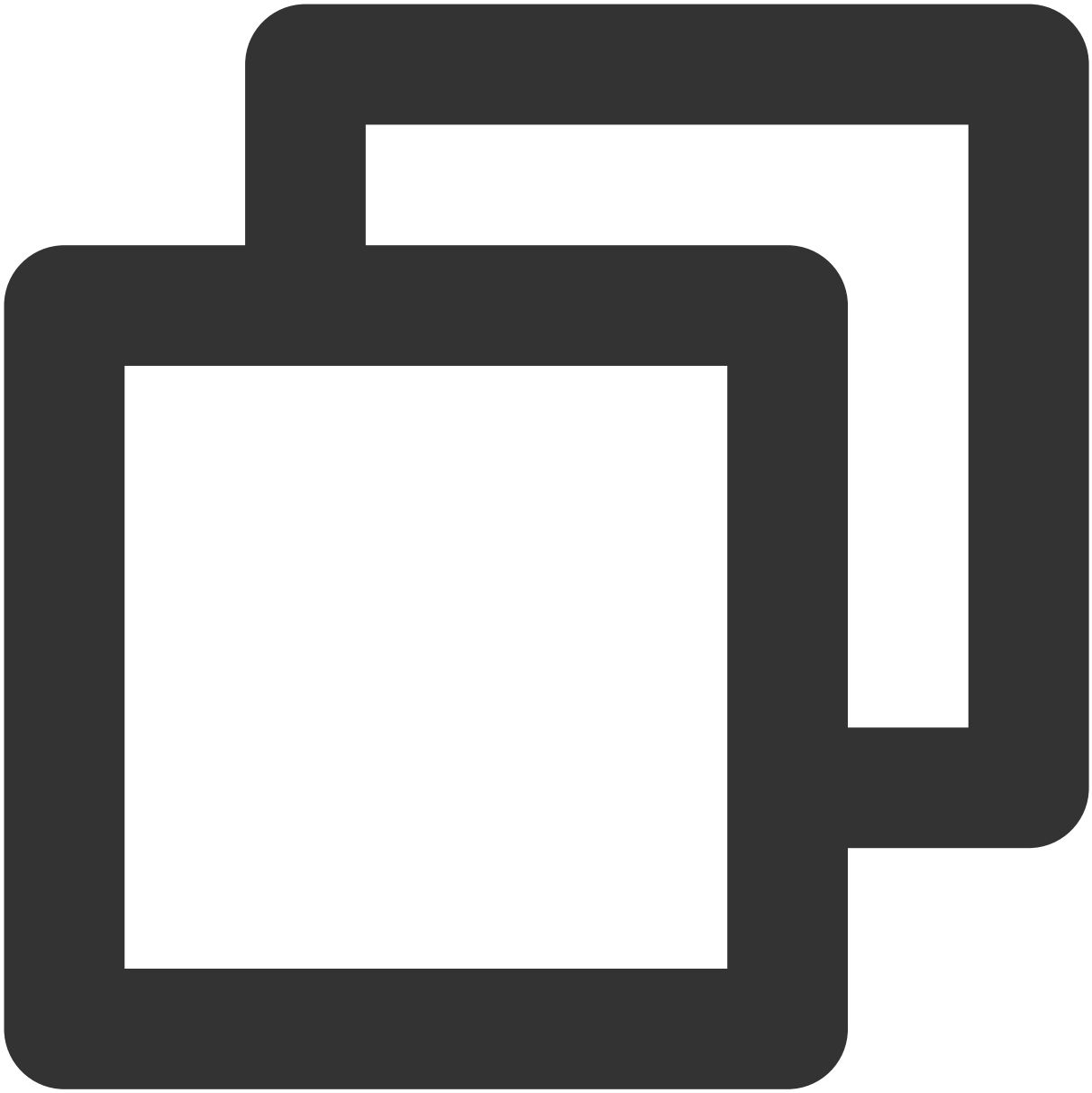
パラメータ	タイプ	意味

roomI	int	ルームIDは、ご自身でアサインし、一元管理してください。複数のroomIDを、一つのKaraokeルームリストにまとめることができます。Tencent Cloudでは現在、Karaokeルームリストの管理サービスを行っていませんので、ご自身でKaraokeルームリストを管理してください。
roomParam	TRTCCreateRoomParam	ルーム情報です。ルーム名、マイク情報、カバー情報など、ルームを説明するために用いる情報に使用します。マイク管理が必要な場合は、ルームのマイク数を記入してください。
callback	ActionCallback	ルームの新規作成結果のコールバック。成功時にcodeは0になります。

管理者が配信を開始する際の通常の呼び出しプロセスは次のとおりです：
1. 管理者は、 `createRoom` を呼び出して新しいKaraokeルームを作成します。この時、ルームID、マイク・オンにすることの管理者の確認の要否、ルームタイプなどルームの属性情報を渡します。
2. 管理者は、ルーム作成に成功した後、 `enterSeat` を呼び出して参加します。
3. 管理者は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上で更新することができます。
4. 管理者は、マイクリストのメンバーが参加した `onAnchorEnterSeat` というイベント通知も受信します。この時、マイク集音は自動的に開始されます。

destroyRoom

ルームの破棄（管理者が呼び出し）。管理者は、ルーム作成後、この関数を呼び出してルームを破棄します。



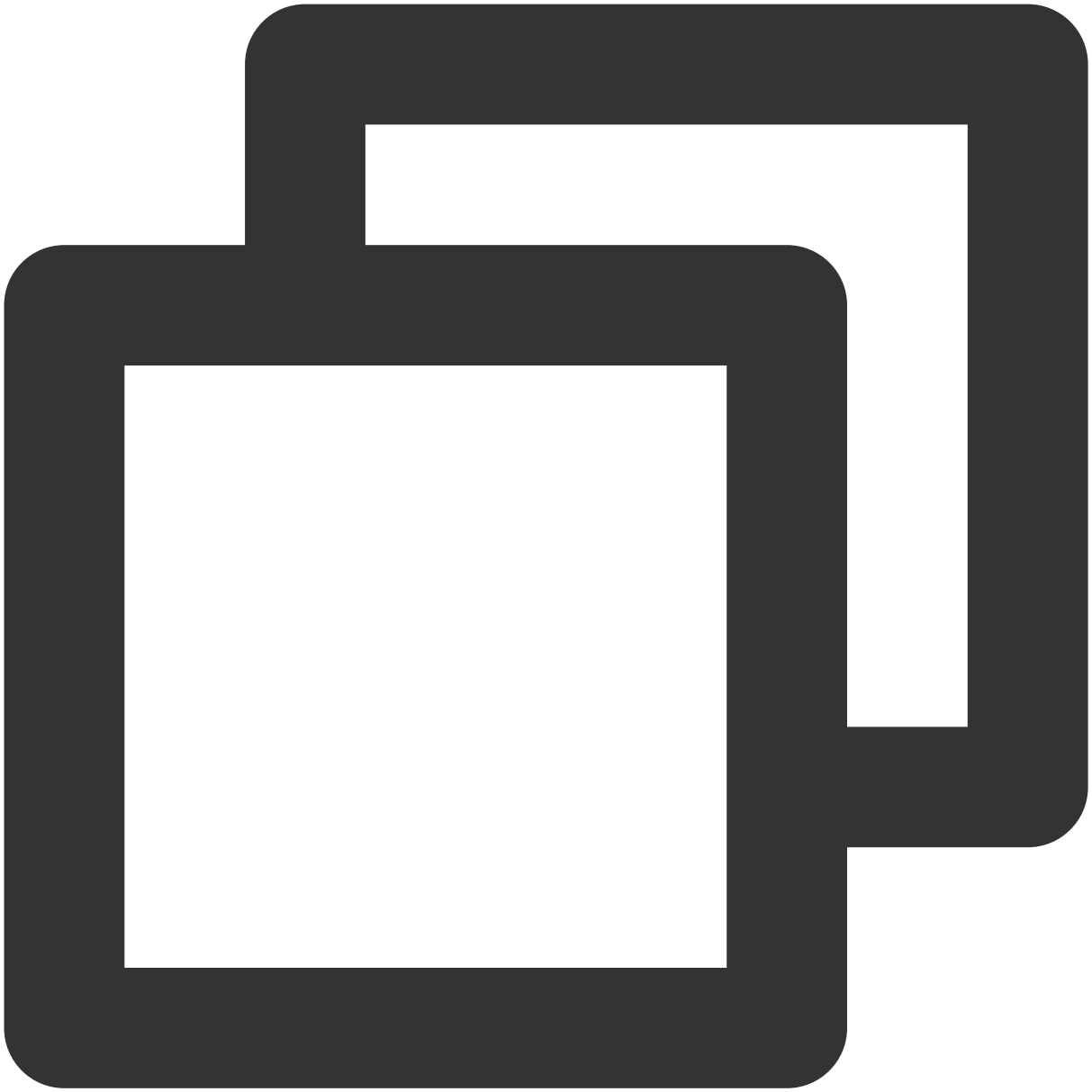
```
public abstract void destroyRoom(TRTCKaraokeRoomCallback.ActionCallback callback);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
callback	ActionCallback	ルームの廃棄結果のコールバック。成功時にcodeは0になります。

enterRoom

入室（リスナーが呼び出し）。



```
public abstract void enterRoom(int roomId, TRTCKaraokeRoomCallback.ActionCallback c
```

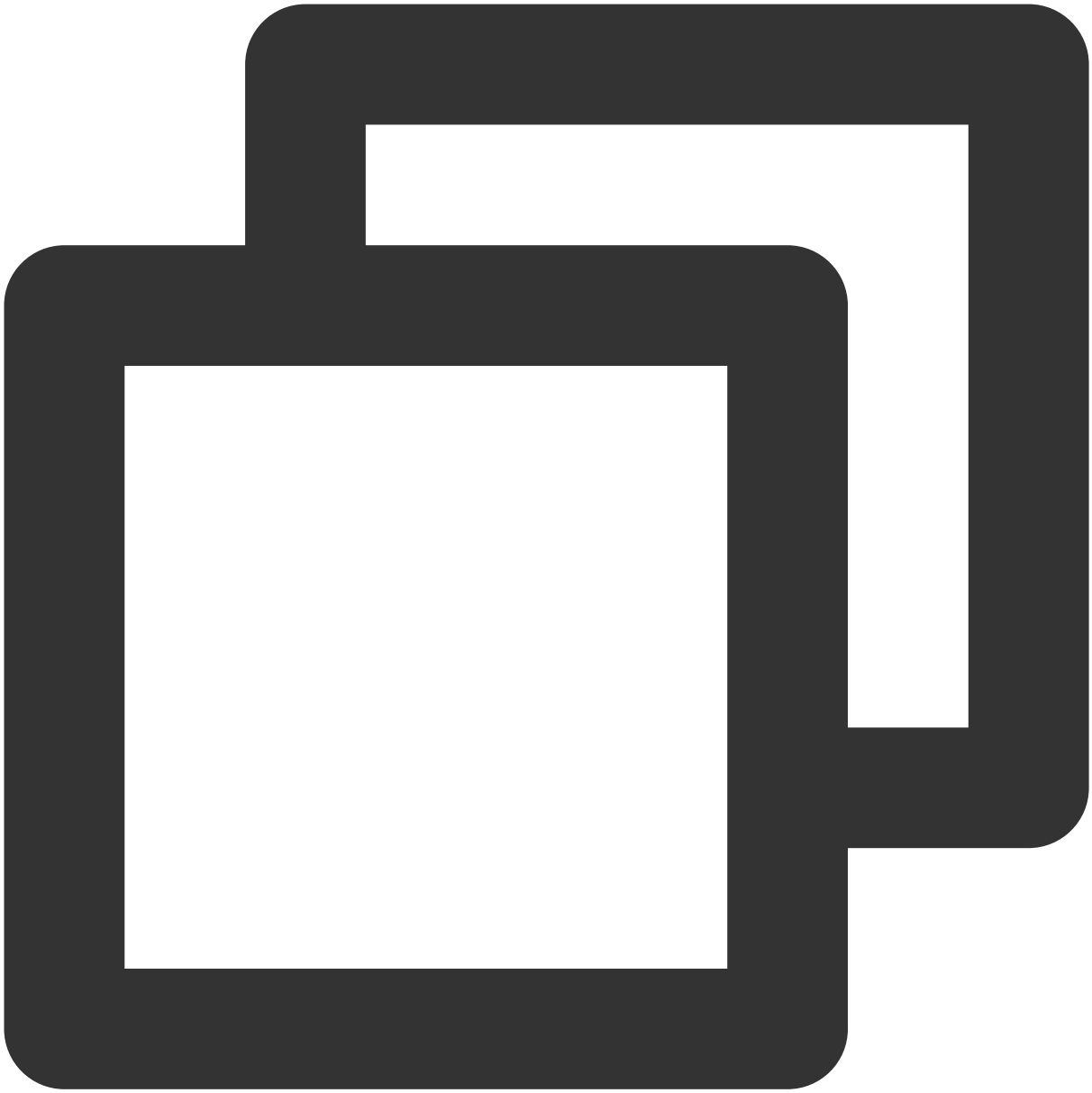
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
roomId	int	ルームID。
callback	ActionCallback	入室結果のコールバック。成功時にcodeは0になります。

リスナーが入室し聴取する際の通常の呼び出しプロセスは次のとおりです：
1. リスナーがサーバーから取得する最新のKaraokeルームリストには、多くのKaraokeルームのroomIdおよびルーム情報が含まれる場合があります。
2. リスナーは1つのKaraokeルームを選択し、 `enterRoom` を呼び出してルームナンバーを渡すと、そのルームに参加できます。
3. 入室後、コンポーネントの `onRoomInfoChange` ルーム属性変更イベント通知を受信します。この時、ルーム属性を記録し、それに応じた修正を行うことができます。例：UIに表示するルーム名、マイク・オンの際の管理者への同意リクエストの要否の記録など。
4. 入室後は、コンポーネントの `onSeatListChange` マイクリスト変更イベント通知を受信します。この時、マイクリストの変更をUI上に更新することができます。
5. 入室後に、マイクリストにキャスターが参加した `onAnchorEnterSeat` のイベント通知も受信します。

exitRoom

ルームから退出します。



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

パラメータは下表に示すとおりです：

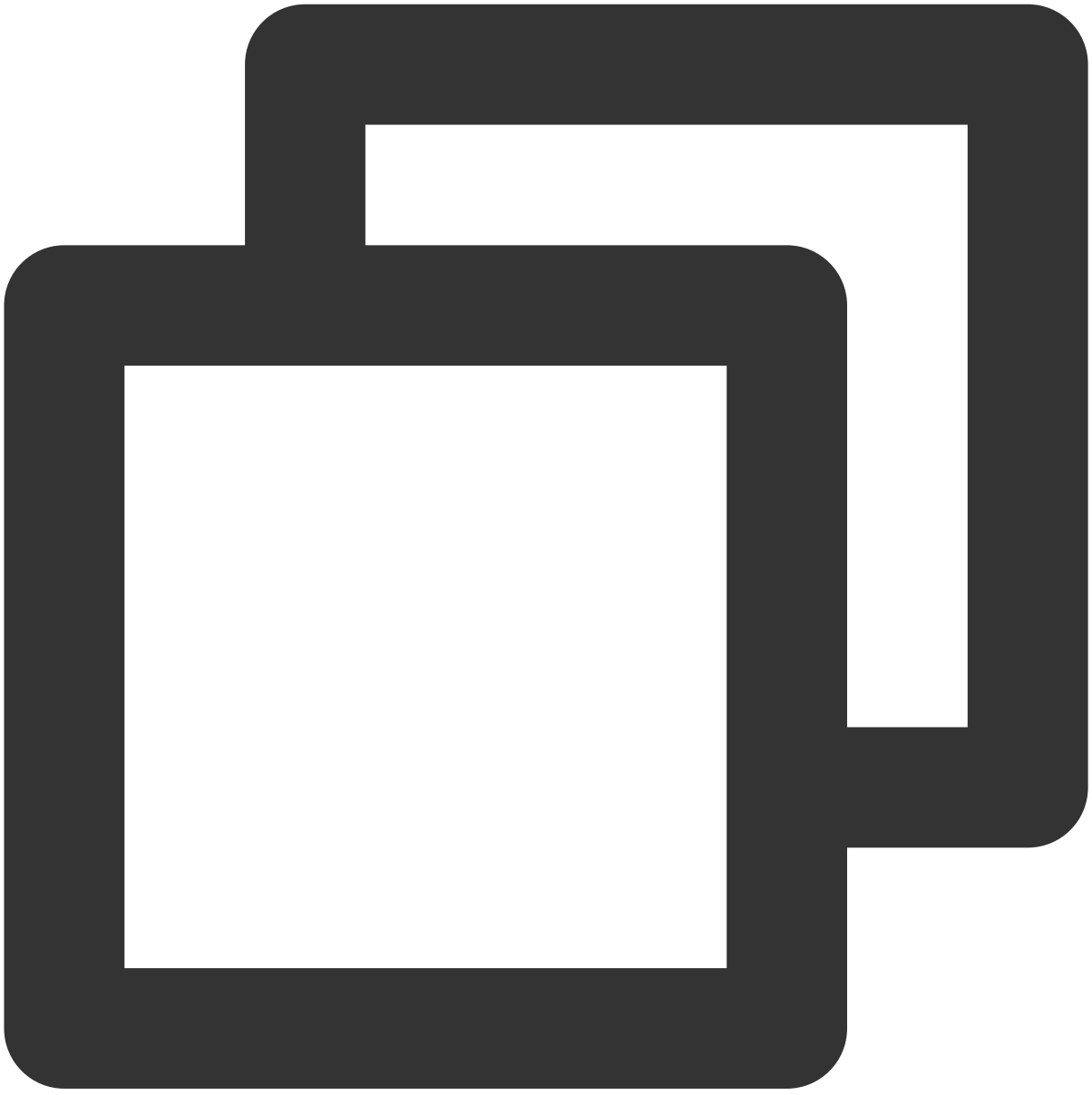
パラメータ	タイプ	意味
callback	ActionCallback	退室結果のコールバック。成功時にcodeは0になります。

getRoomInfoList

ルームリストの詳細情報を取得します。このうち、ルーム名、ルームカバーは、管理者が `createRoom()` 作成時にroomInfoによって設定したものになります。

説明：

ルームリストおよびルーム情報をご自身で管理する場合は、この関数は無視できます。



```
public abstract void getRoomInfoList(List<Integer> roomIdList, TRTCKaraokeRoomCallb
```

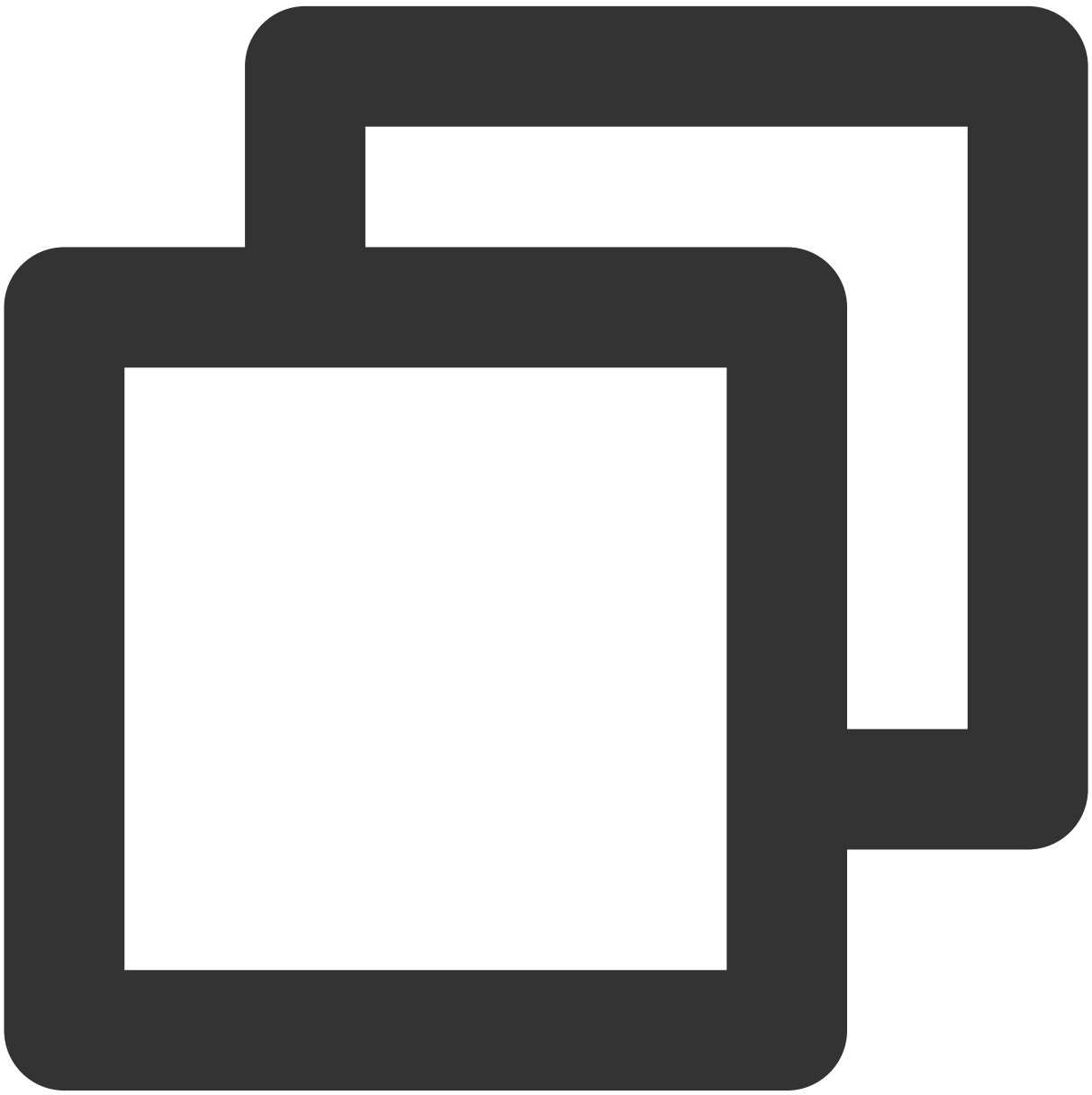
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

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

getUserInfoList

指定されたuserIdのユーザー情報を取得します。



```
public abstract void getUserInfoList(List<String> userIdList, TRTCKaraokeRoomCallba
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userIdList	List<String>	取得すべきユーザーIDリスト。nullの場合は、ルーム内全員の情報を取得します。
userlistcallback	UserListCallback	ユーザーの詳細情報のコールバック。

音楽再生インターフェース

startPlayMusic

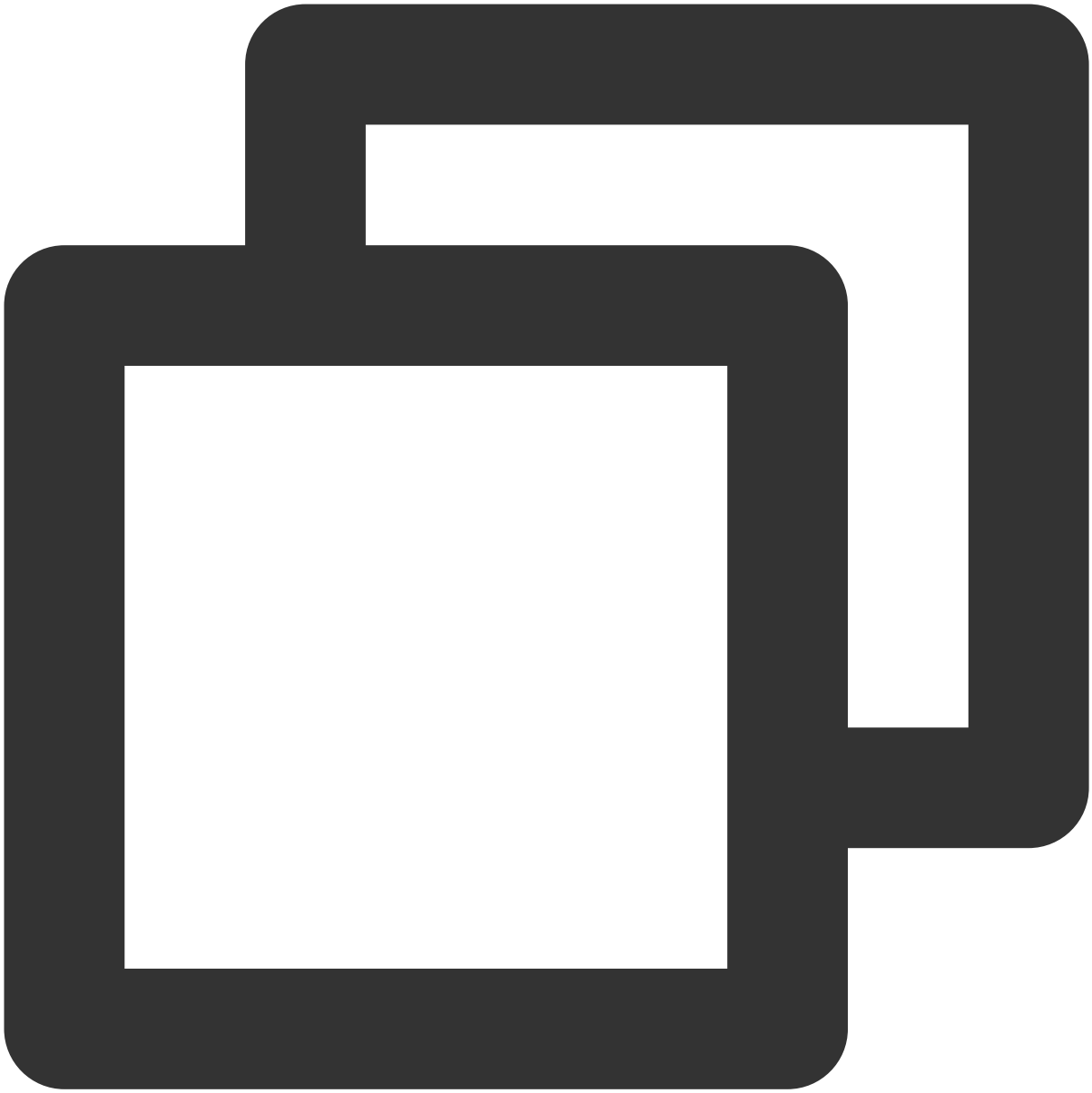
音楽を再生します（マイク・オン後に呼び出し）。

説明：

音楽を再生すると、 `onMusicPrepareToPlay` というイベント通知を受信します。

音楽の再生中、ルーム内の全メンバーは、 `onMusicProgressUpdate` というイベント通知を継続して受け取ります。

音楽の再生が完了すると、 `onMusicCompletePlaying` というイベント通知を受信します。



```
public abstract void startPlayMusic(int musicID, String originalUrl, String accompa
```

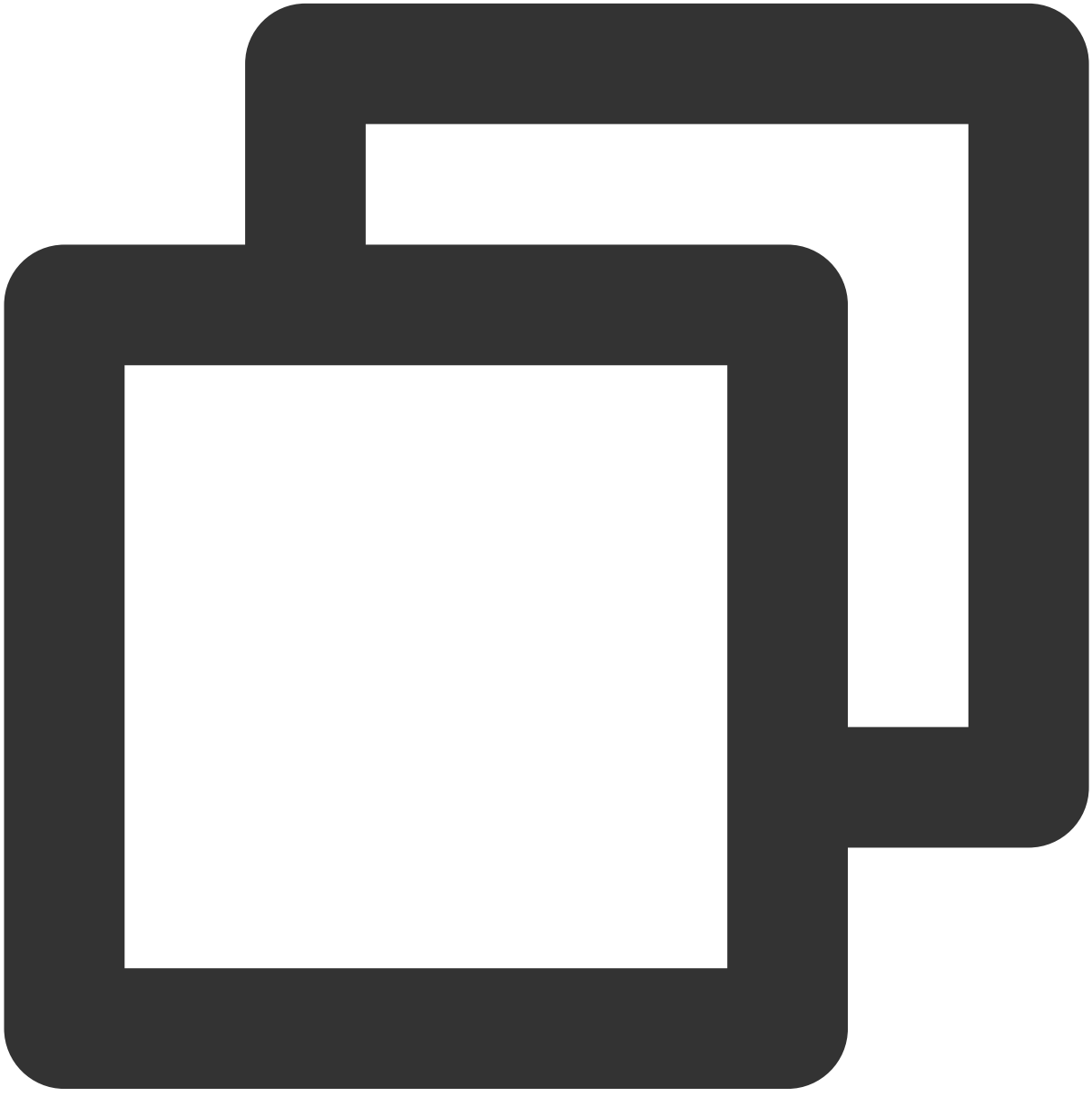
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int	音楽のID。
originalUrl	String	原曲の絶対パス。
accompanyUrl	String	伴奏の絶対パス。

このインターフェースを呼び出すと、最後に再生されていた楽曲が停止します。

stopPlayMusic

音楽の再生を停止します（音楽を再生するときに呼び出します）。
説明：
再生が停止すると、 `onMusicCompletePlaying` というイベント通知を受信します。



```
public abstract void stopPlayMusic();
```

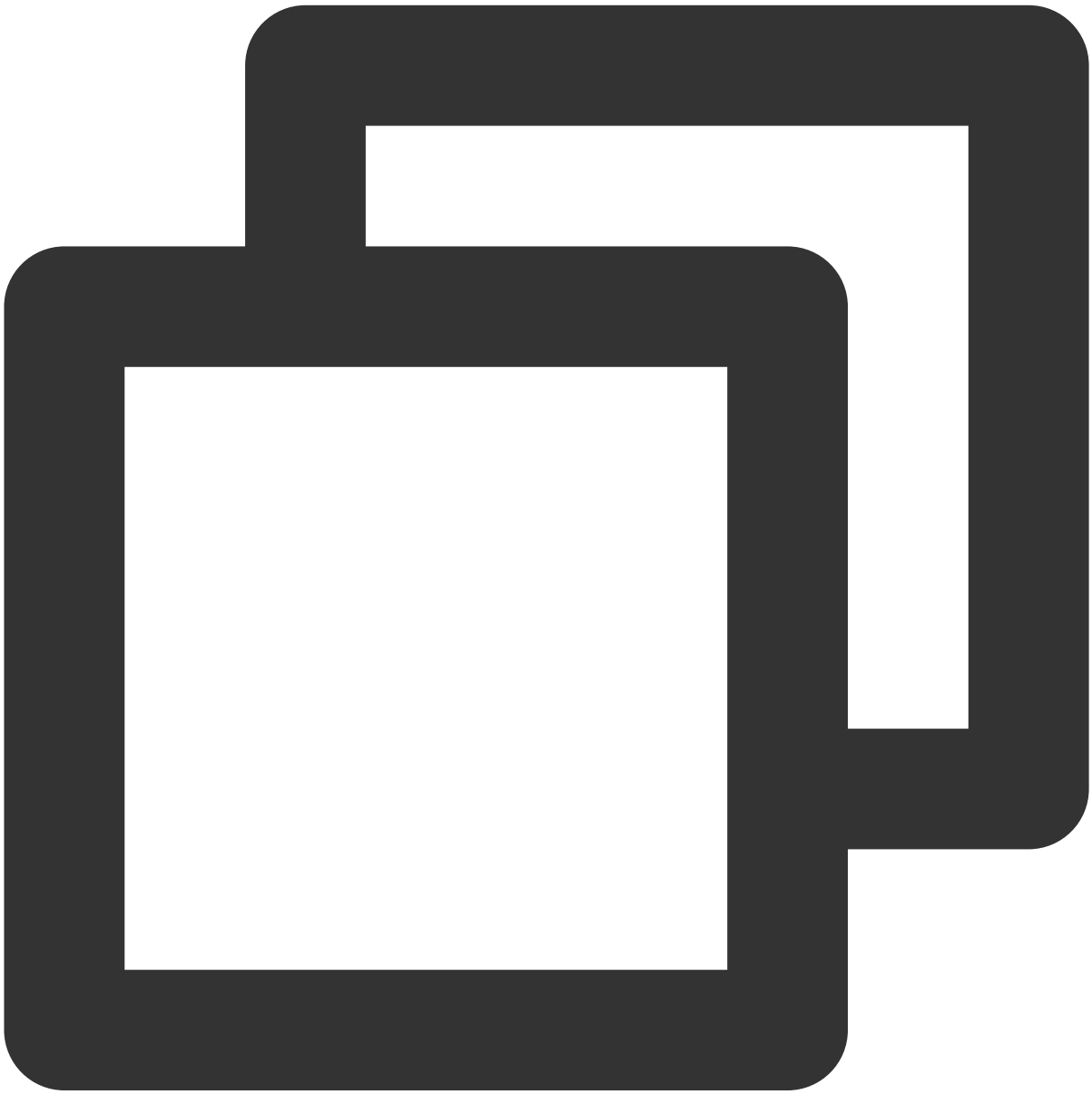
pausePlayMusic

音楽の再生を一時停止します（音楽を再生するときに呼び出します）。

説明：

`onMusicProgressUpdate` というイベント通知が一時停止されます

`onMusicCompletePlaying` というイベント通知を受信しません。



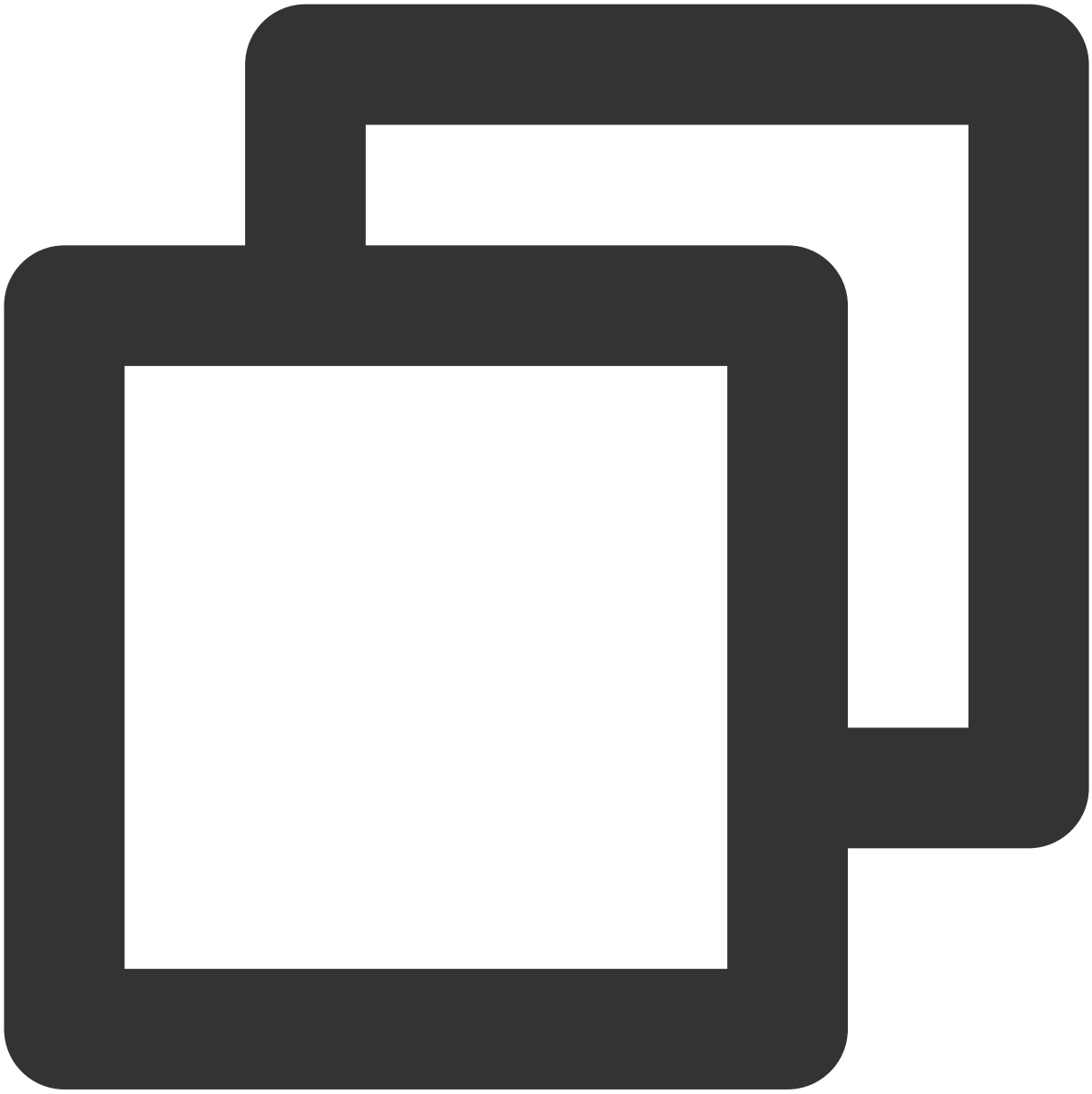
```
public abstract void pausePlayMusic();
```

resumePlayMusic

一時停止した音楽を再開します（一時停止後に呼び出します）。

説明：

`onMusicPrepareToPlay` というイベント通知を受信しません。



```
public abstract void resumePlayMusic();
```

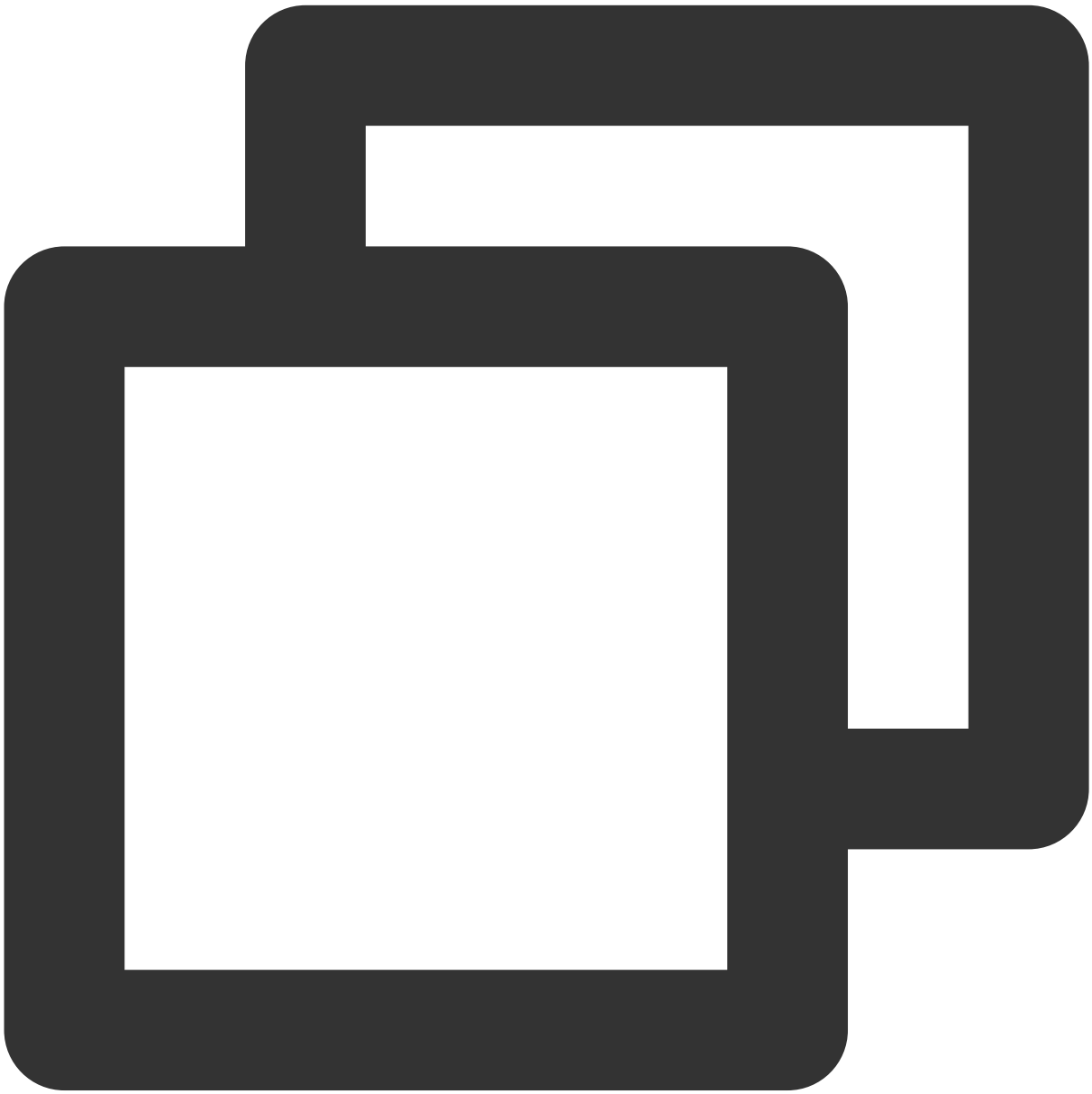
マイク管理インターフェース

enterSeat

ユーザーが発言者になります（リスナー側/管理者ともに呼び出し可）。

説明：

マイク・オンの成功後、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorEnterSeat` というイベント通知を受信します。



```
public abstract void enterSeat(int seatIndex, TRTCKaraokeRoomCallback.ActionCallbac
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	マイク・オンの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

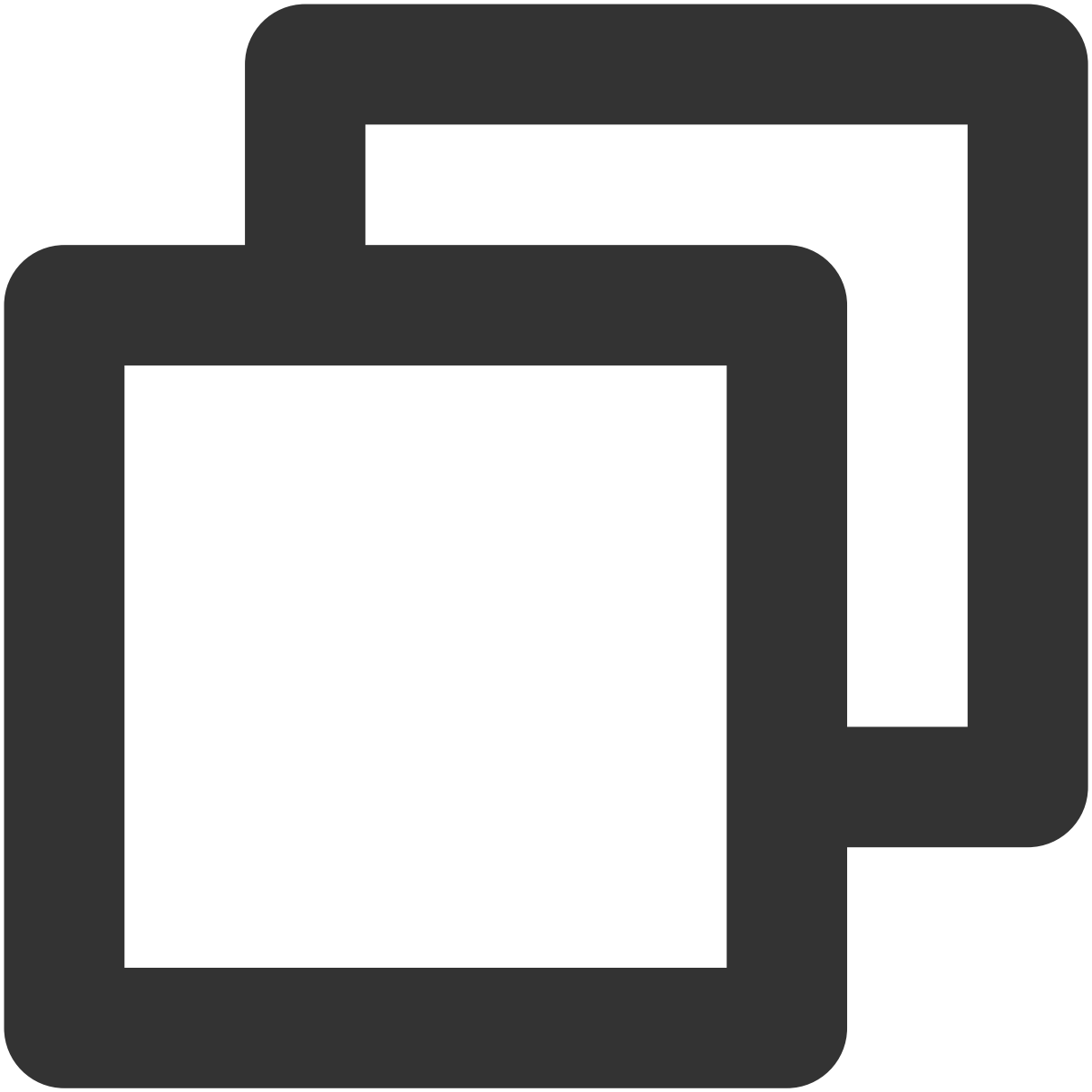
このインターフェースを呼び出すと、直ちにマイクリストが変更されます。リスナーによるマイク・オンの申請に管理者の同意が必要となるケースの場合は、まず `sendInvitation` を呼び出してから管理者に申請し、`onInvitationAccept` を受信するとこの関数を呼び出せるようになります。

leaveSeat

ユーザーが視聴者になります（キャスターが呼び出し）。

説明：

マイク・オフの成功後、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。



```
public abstract void leaveSeat(TRTCKaraokeRoomCallback.ActionCallback callback);
```

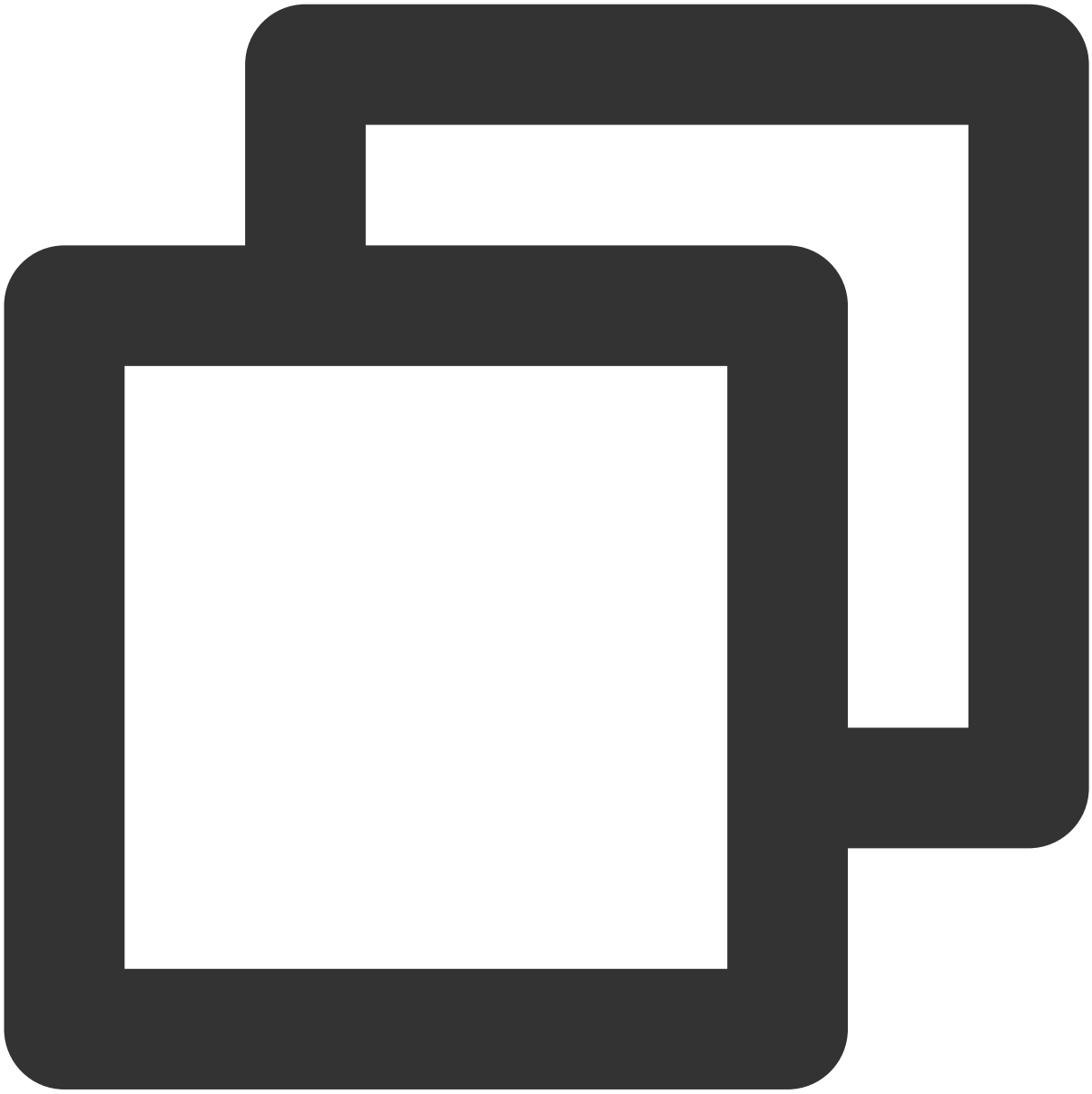
パラメータは下表に示すとおりです：

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

pickSeat

視聴者が発言できるように招待（管理者が呼び出し）。

説明：
管理者が視聴者を発言できるように招待すると、ルーム内の全メンバーは、`onSeatListChange` と `onAnchorEnterSeat` というイベント通知を受信します。



```
public abstract void pickSeat(int seatIndex, String userId, TRTCKaraokeRoomCallback
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	視聴者が発言できるように招待する必要があるマイク番号。

userId	String	ユーザーID。
callback	ActionCallback	操作コールバック。

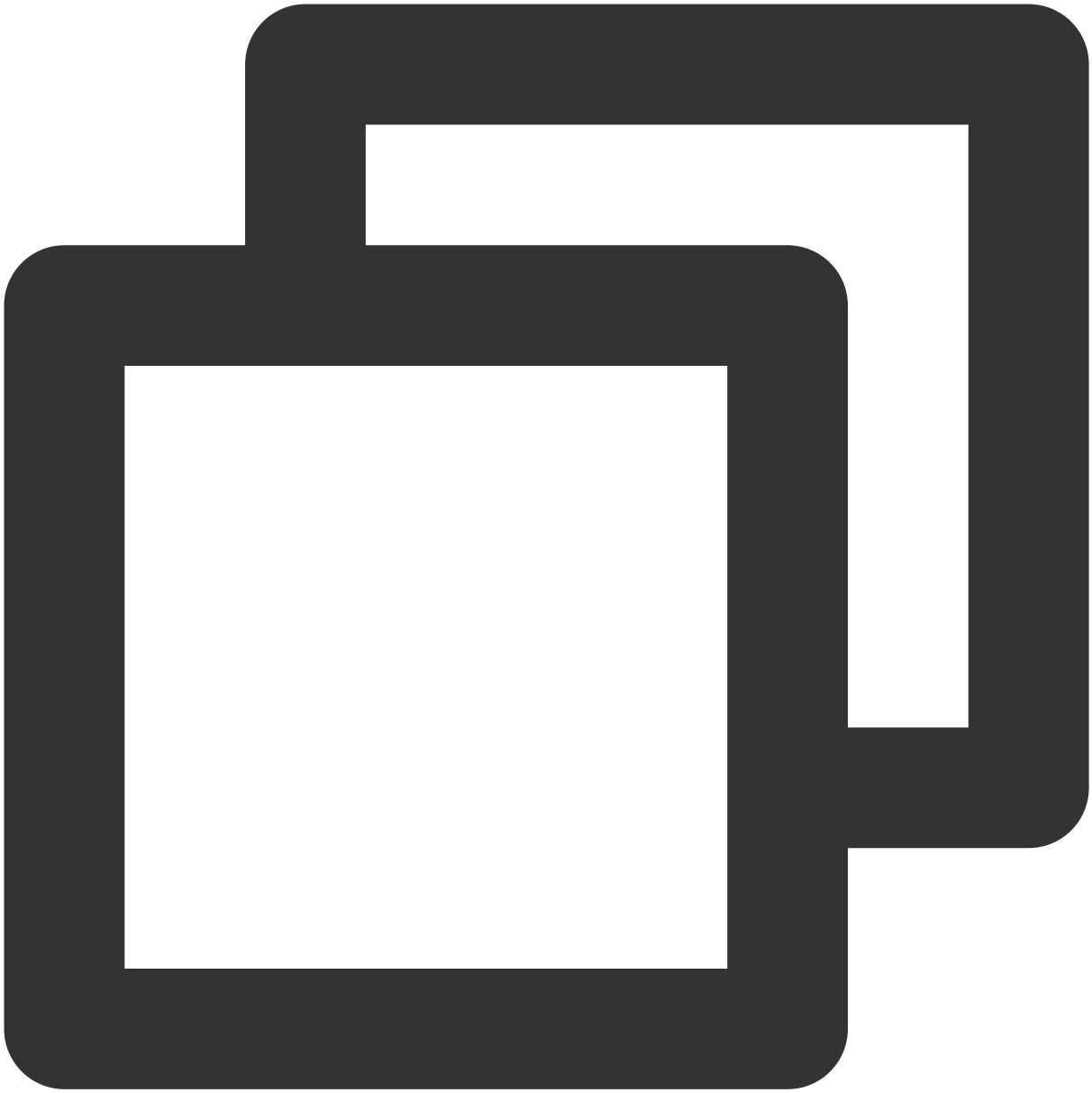
このインターフェースを呼び出すと、すぐにマイクリストが修正されます。管理者がリスナーの同意がなければマイク・オンできないケースの場合は、まず `sendInvitation` を呼び出してからリスナーに申請し、`onInvitationAccept` を受信すると、この関数をコールできるようになります。

kickSeat

キックアウトしてマイク・オフ（管理者が呼び出し）。

説明：

管理者がキックアウトしてマイク・オフにすると、ルーム内の全メンバーは、`onSeatListChange` および `onAnchorLeaveSeat` というイベント通知を受信します。



```
public abstract void kickSeat(int seatIndex, TRTCKaraokeRoomCallback.ActionCallback
```

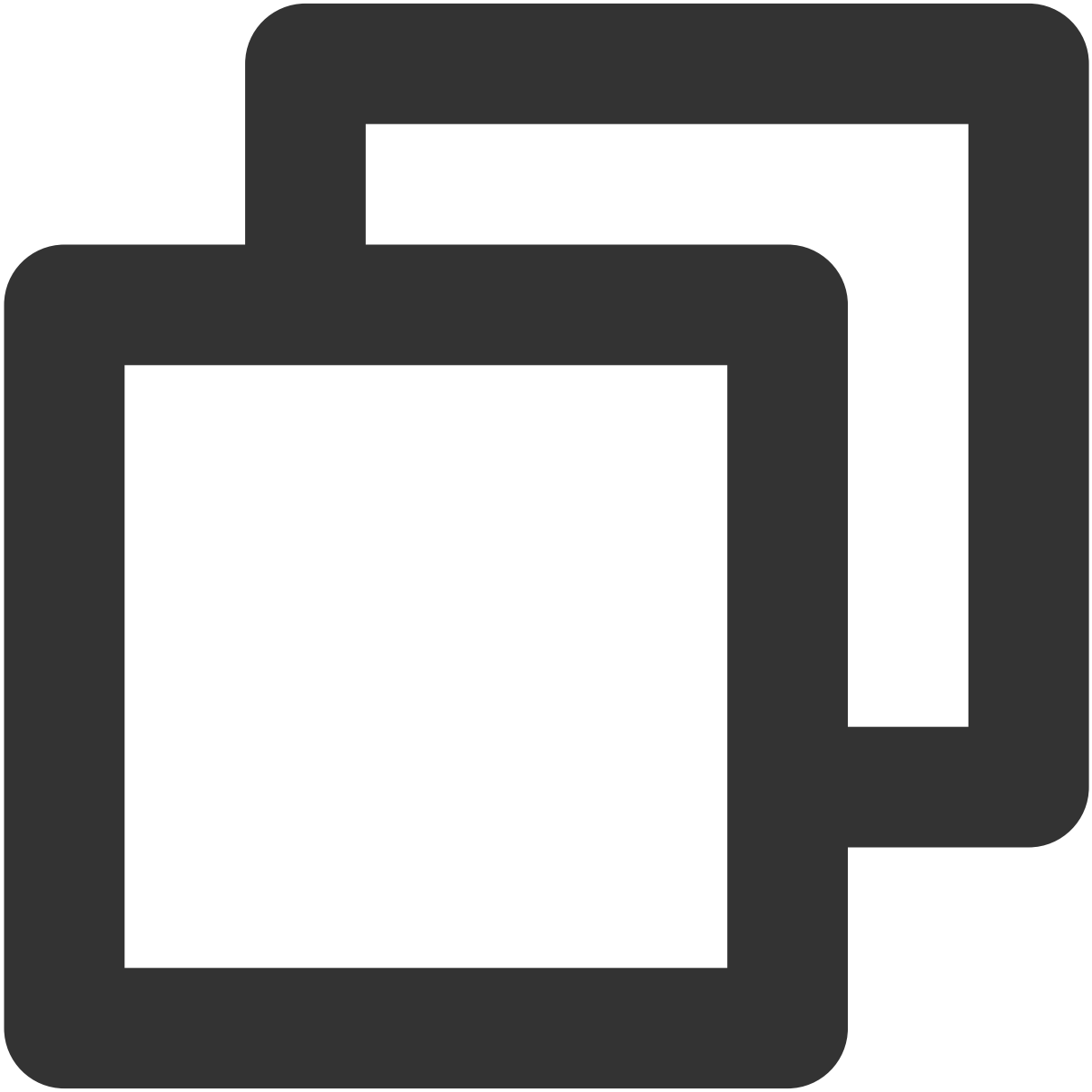
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	キックアウトしてマイク・オフの必要があるマイク番号。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。

muteSeat

任意のマイクのミュート/ミュート解除（管理者が呼び出し）。
説明：
任意のマイクをミュート/ミュート解除します。ルーム内の全メンバーが `onSeatListChange` および `onSeatMute` というイベント通知を受信します。



```
public abstract void muteSeat(int seatIndex, boolean isMute, TRTCKaraokeRoomCallbac
```

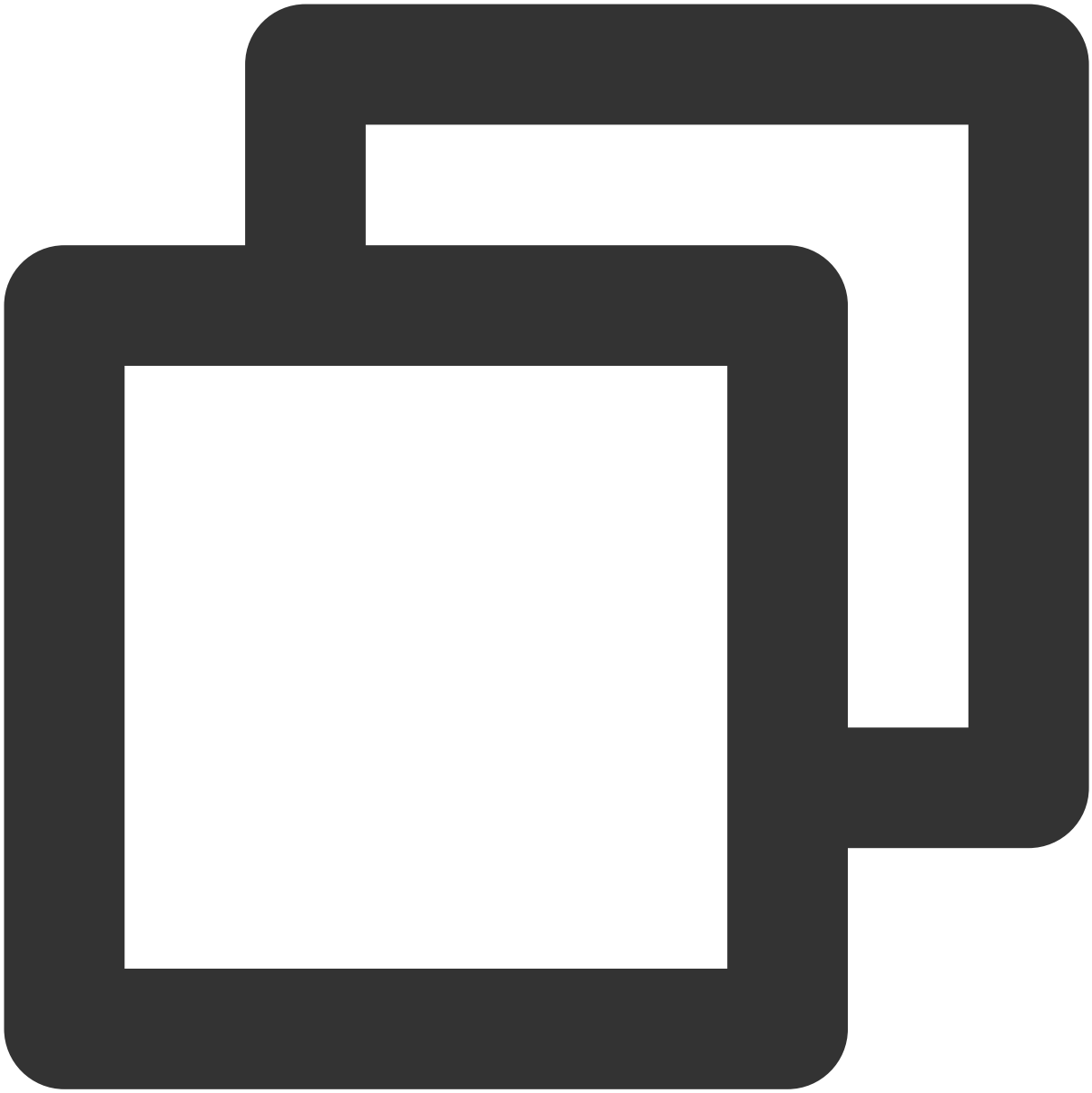
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isMute	boolean	true：該当するマイクをミュートします；false：該当するマイクをミュート解除します。
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、直ちにマイクリストが変更されます。seatIndexの座席に該当するキャスターは、muteAudioを自動的に呼び出してミュート/ミュート解除にします。

closeSeat

任意のマイクのクローズ/解除（管理者が呼び出し）。
説明：
管理者は、該当するマイクをクローズ/解除し、ルーム内の全メンバーは `onSeatListChange` および `onSeatClose` というイベント通知を受信します。



```
public abstract void closeSeat(int seatIndex, boolean isClose, TRTCKaraokeRoomCallb
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatIndex	int	操作の必要があるマイク番号。
isClose	boolean	true：該当するマイクをクローズします； false：該当するマイクをクローズ解除します。

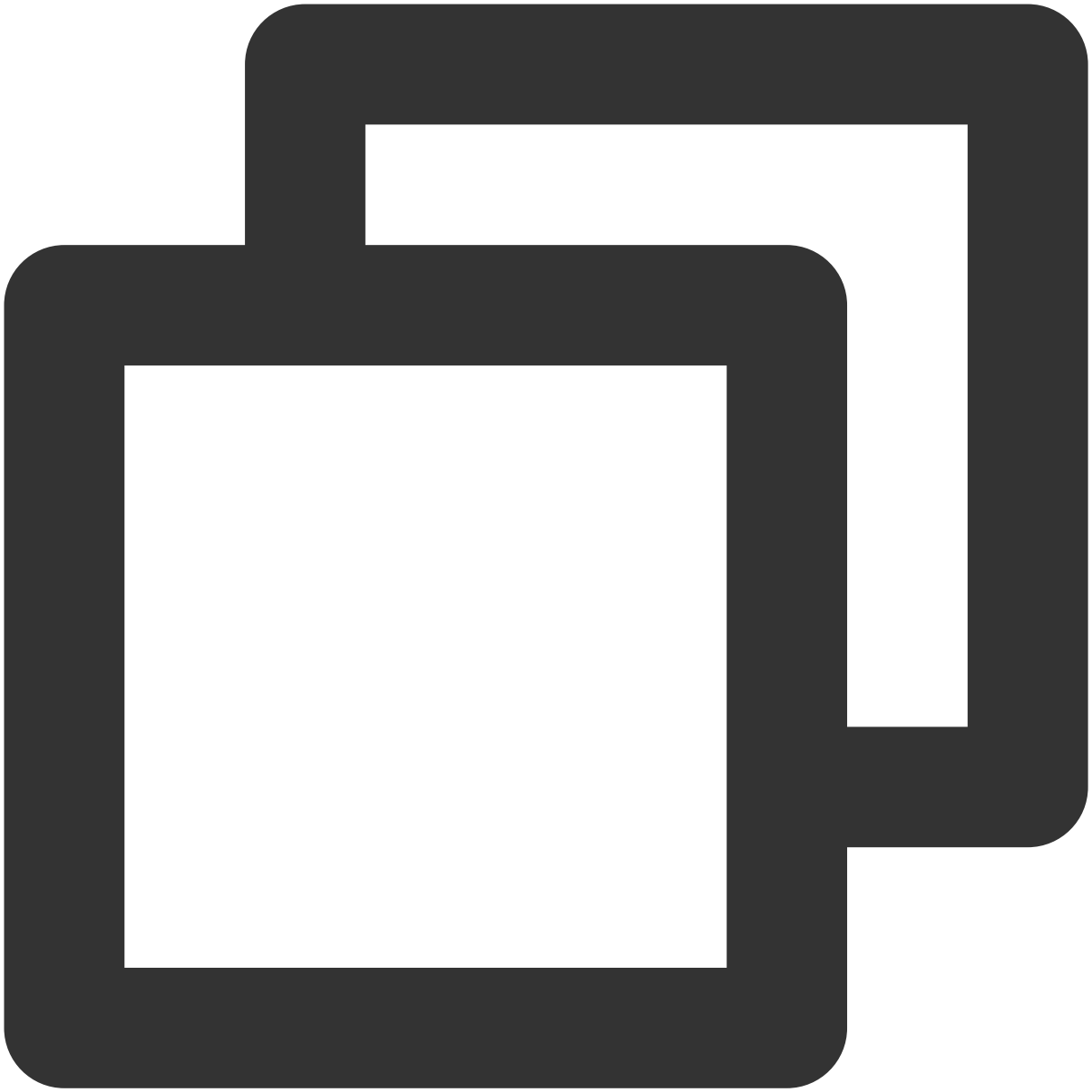
callback	ActionCallback	操作コールバック。

このインターフェースを呼び出すと、すぐにマイクリストが修正されます。該当するseatIndexの座席上のキャスターはクローズされ、自動的にマイク・オフになります。

ローカル音声操作のインターフェース

startMicrophone

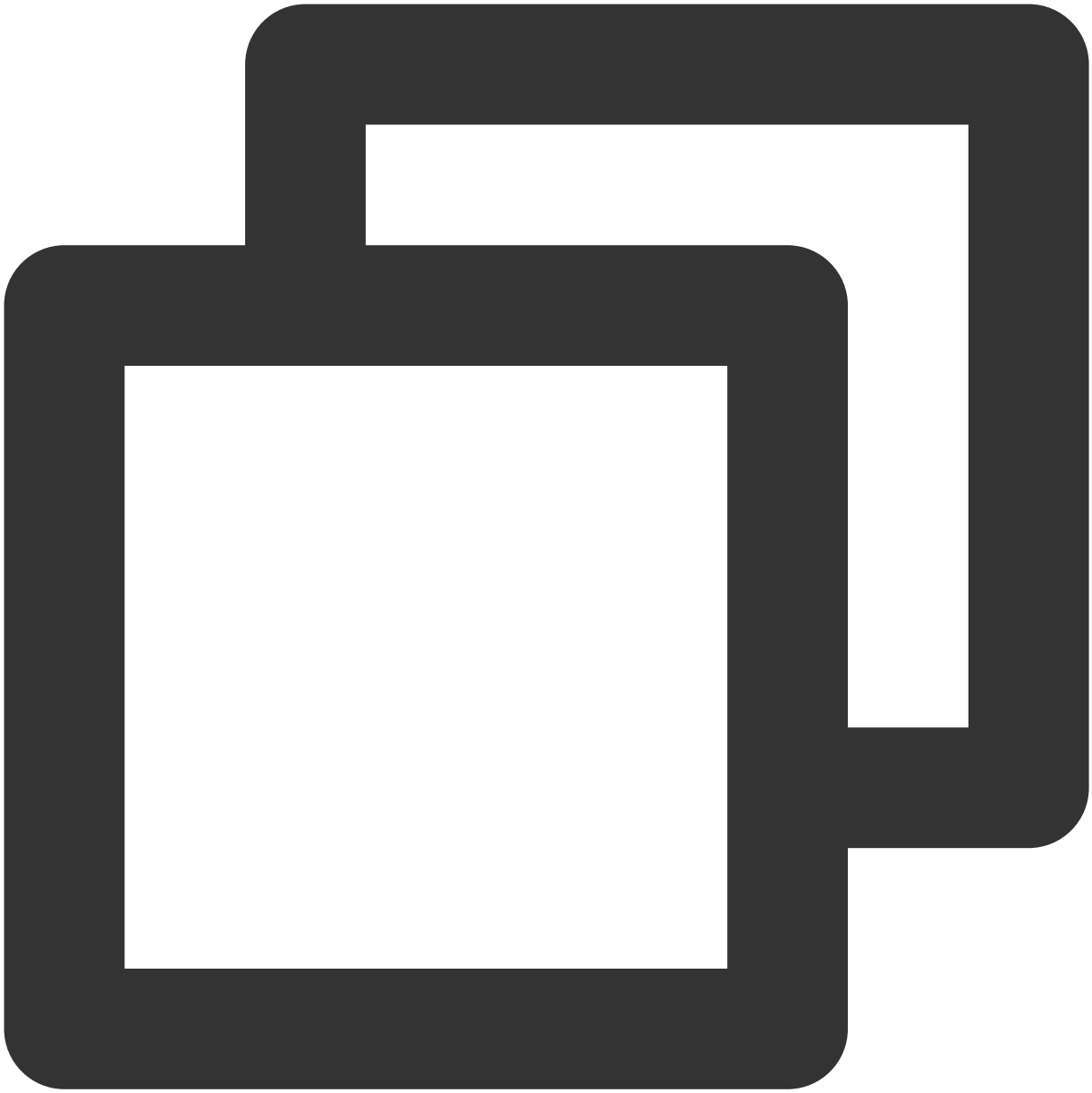
マイクの集音開始。



```
public abstract void startMicrophone();
```

stopMicrophone

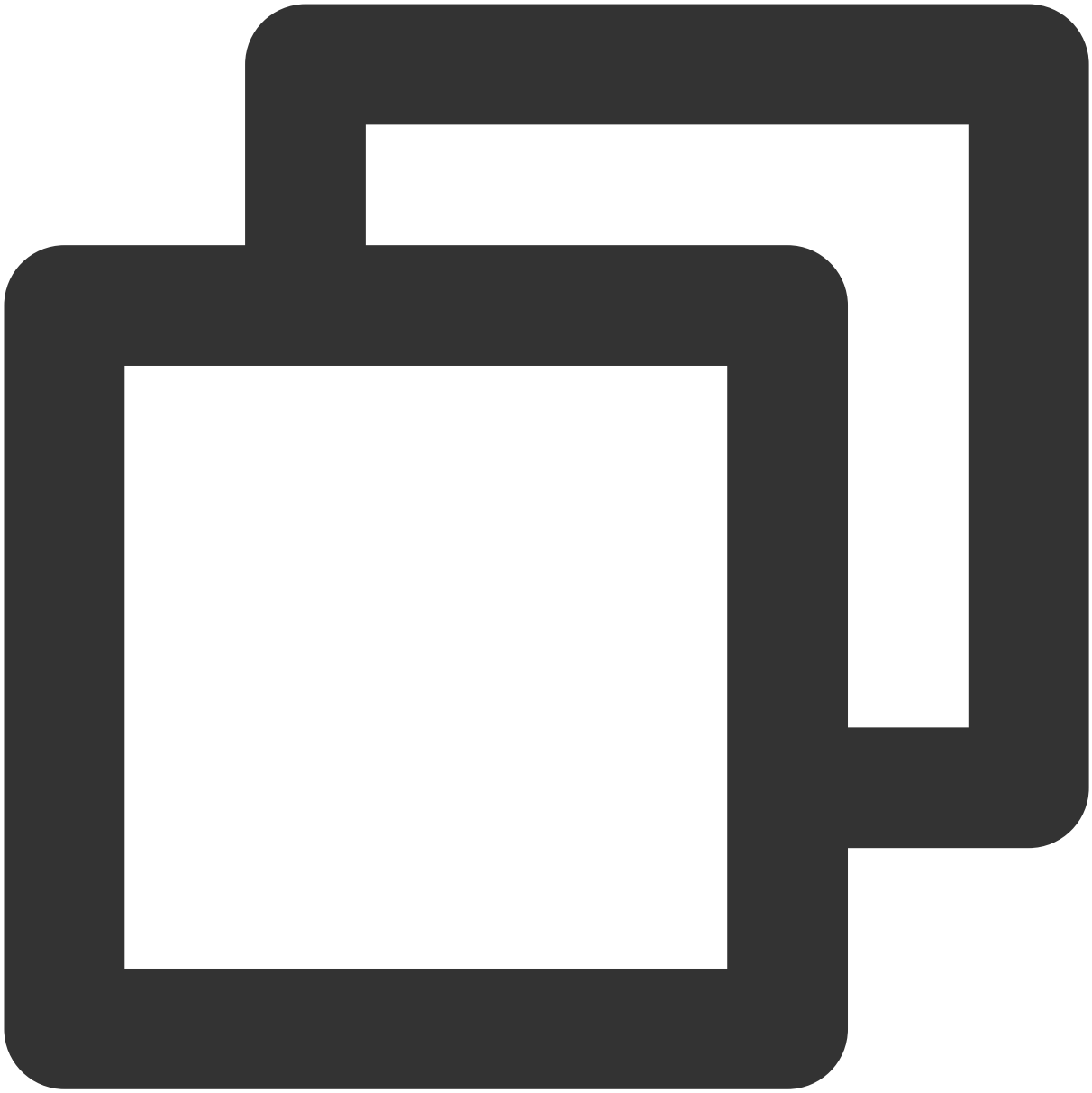
マイクの集音停止。



```
public abstract void stopMicrophone();
```

setAudioQuality

音質の設定。



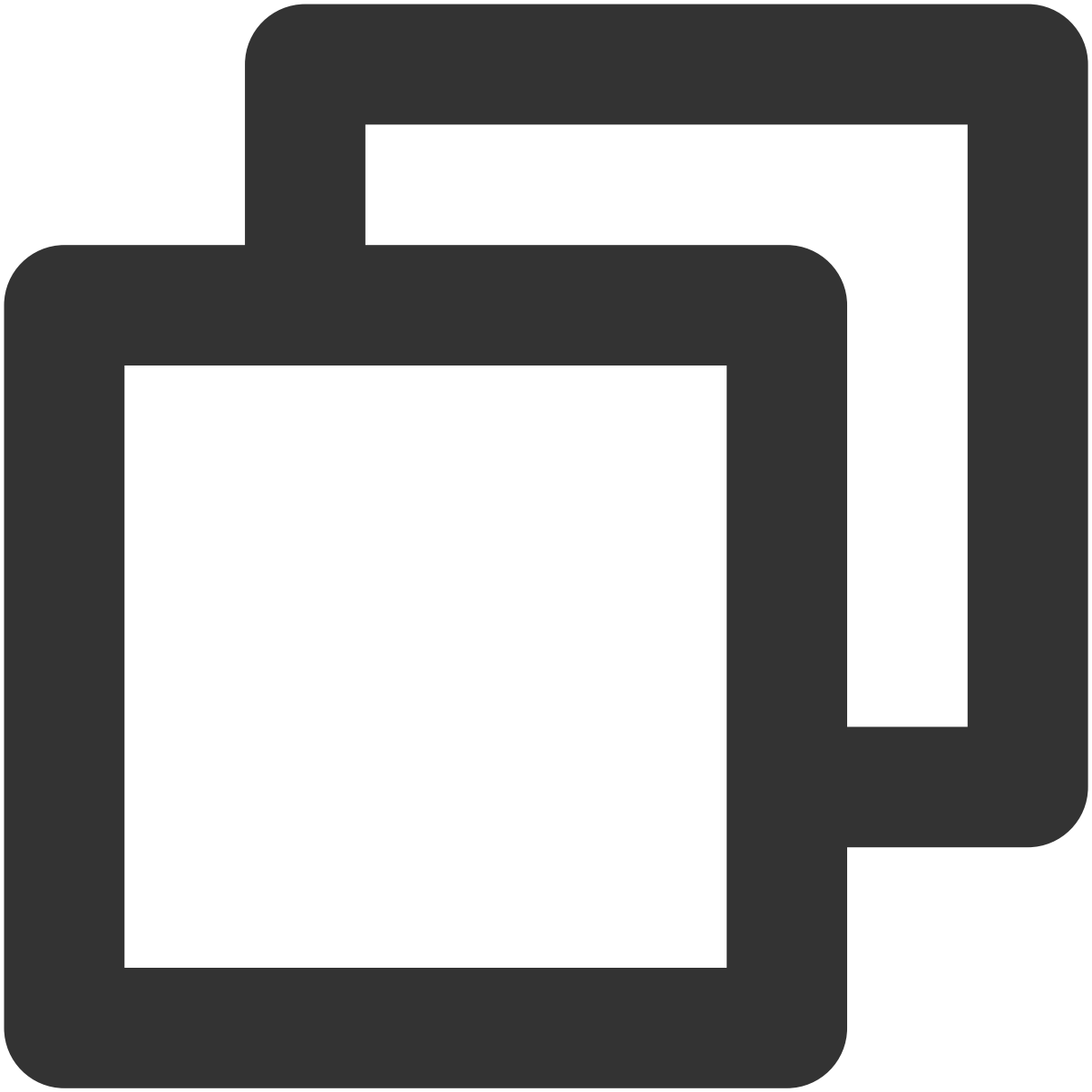
```
public abstract void setAudioQuality(int quality);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
quality	int	音声品質。詳細は TRTC SDKをご参照ください。

muteLocalAudio

ローカルの音声のミュート/ミュート取り消し。



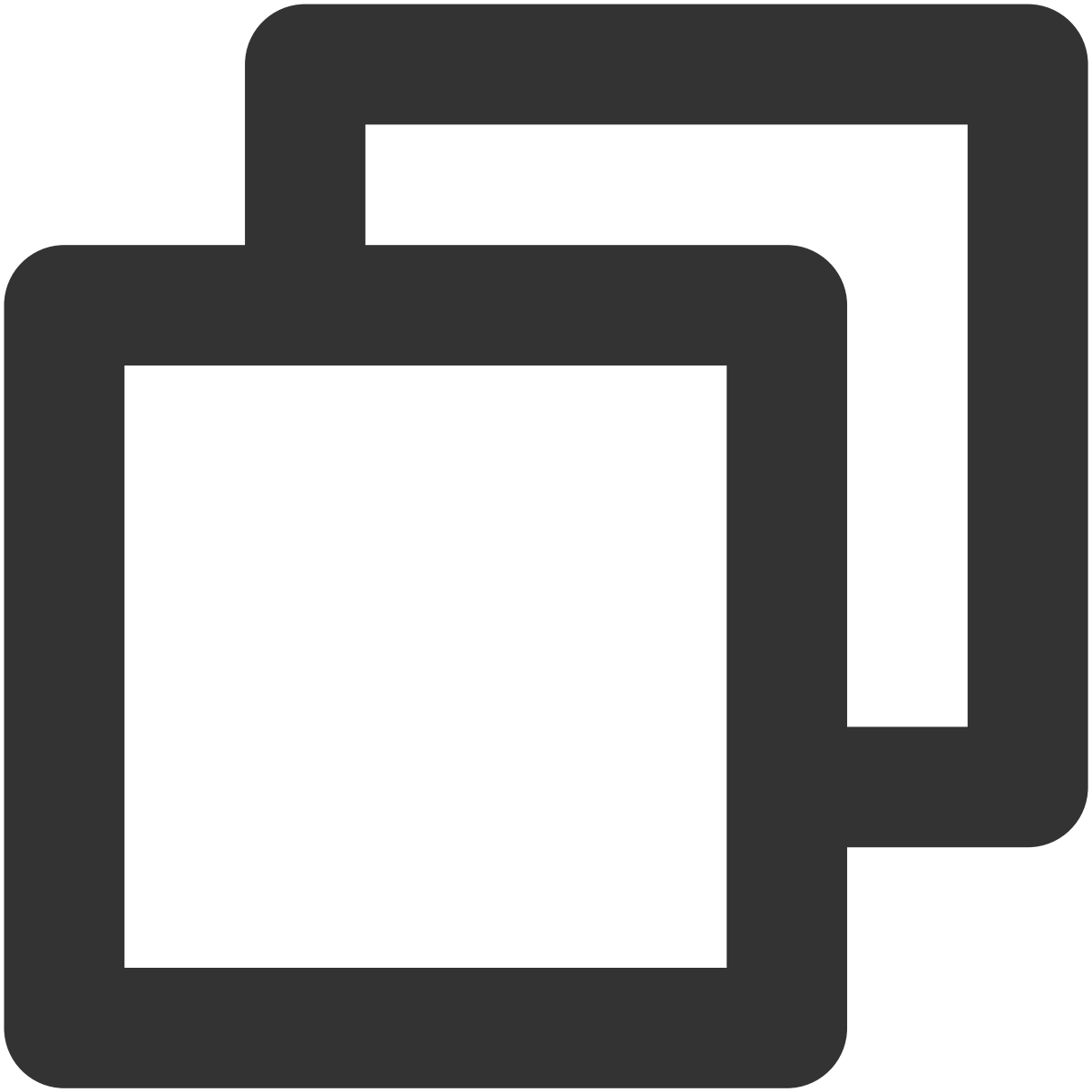
```
public abstract void muteLocalAudio(boolean mute);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
mute	boolean	ミュート/ミュート取り消し。詳細は TRTC SDKをご参照ください。

setSpeaker

スピーカーの起動設定。



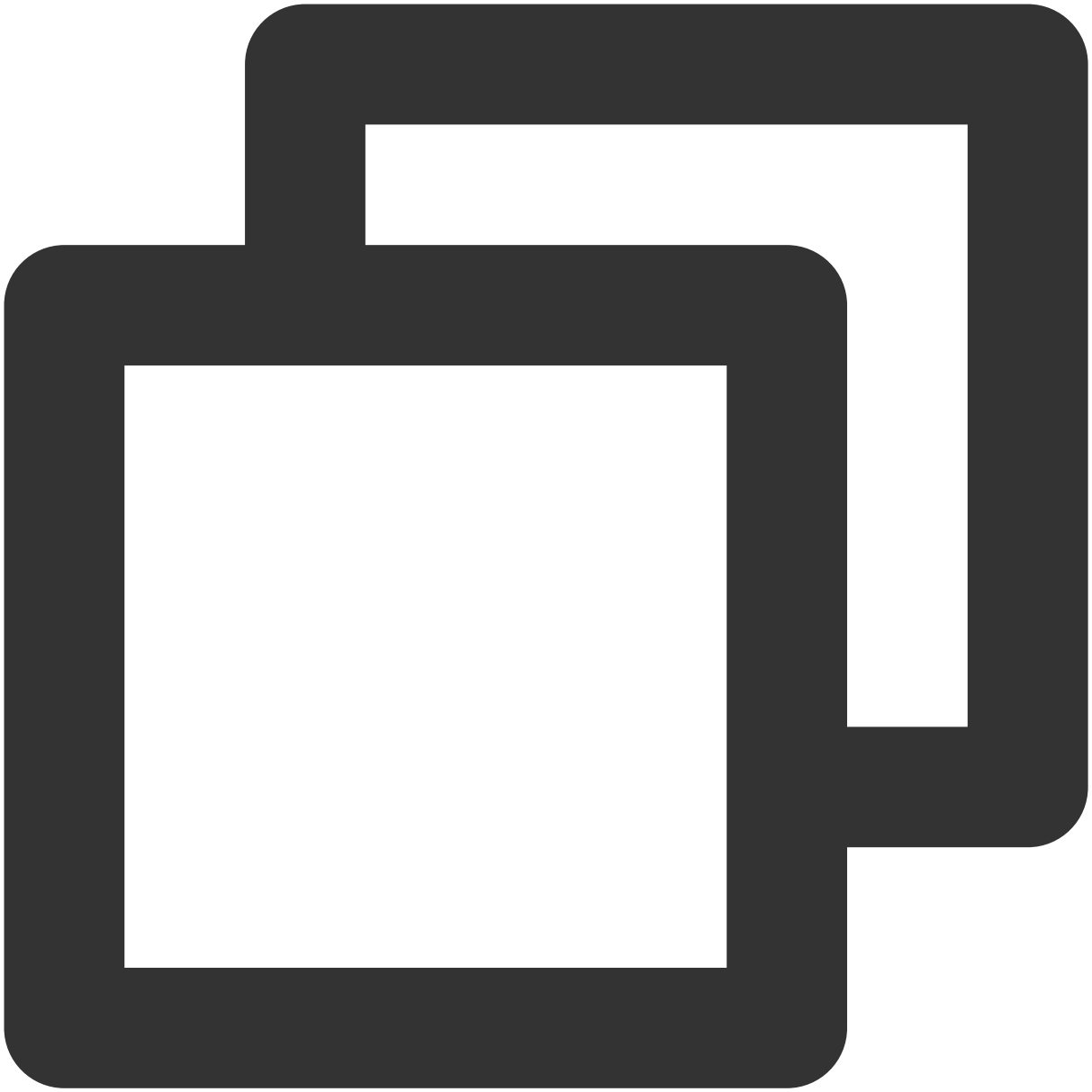
```
public abstract void setSpeaker(boolean useSpeaker);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
useSpeaker	boolean	true：スピーカー、false：ヘッドホン。

setAudioCaptureVolume

マイクの集音音量設定。



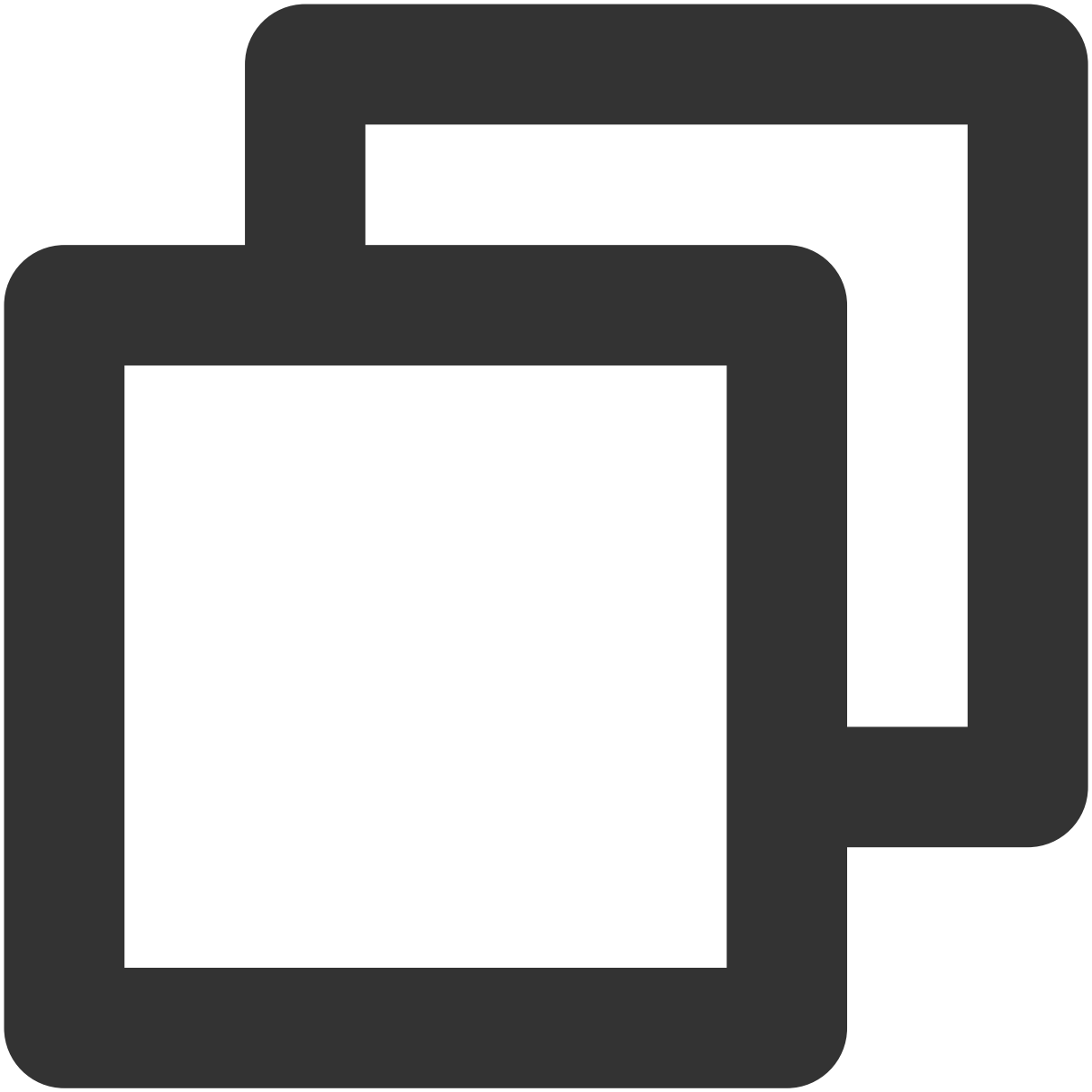
```
public abstract void setAudioCaptureVolume(int volume);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
volume	int	集音音量、0 - 100、 デフォルト100。

setAudioPlayoutVolume

再生音量の設定。



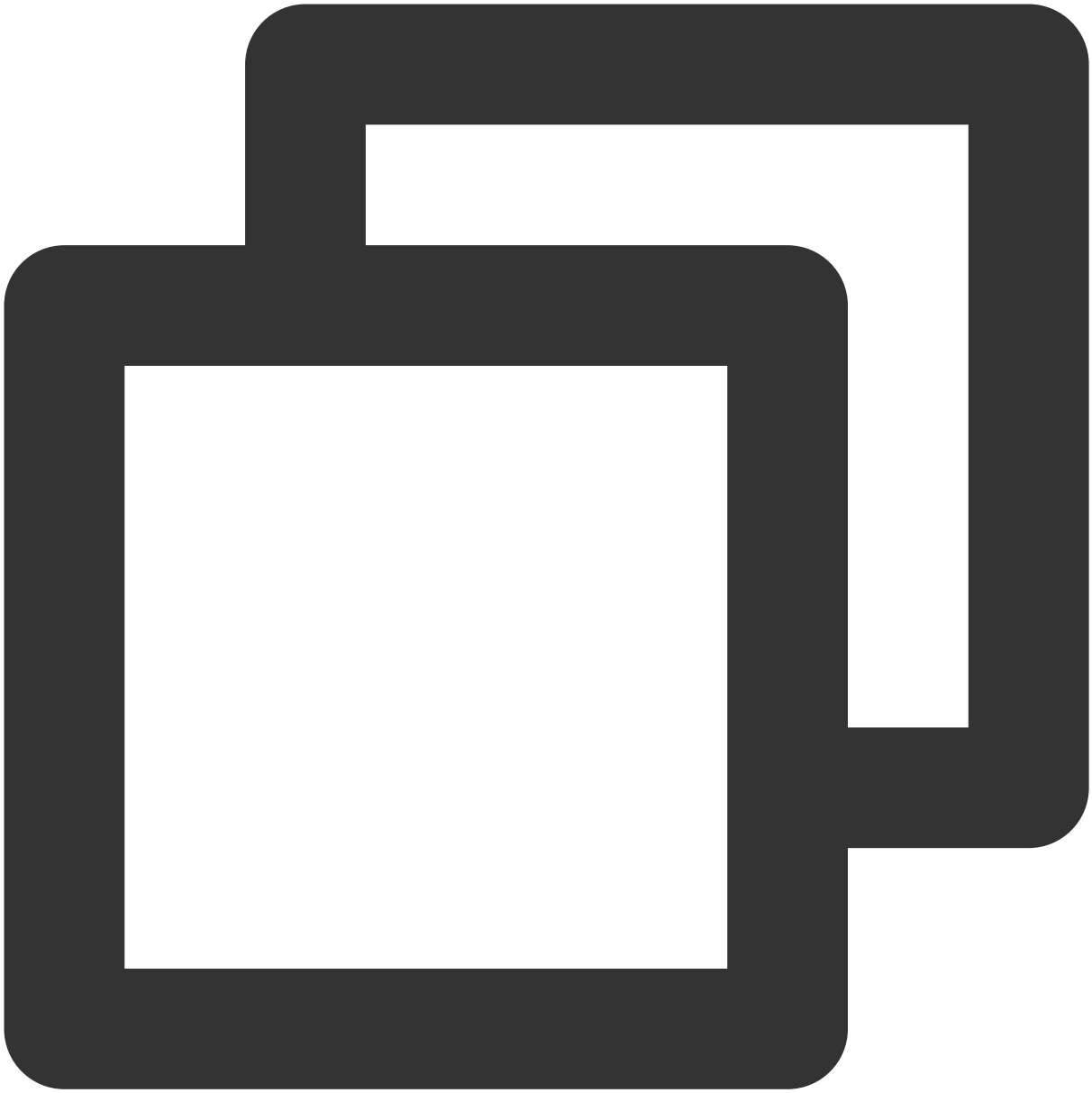
```
public abstract void setAudioPlayoutVolume(int volume);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
volume	int	再生音量、0 - 100、 デフォルト100。

muteRemoteAudio

指定メンバーのミュート/ミュート解除。



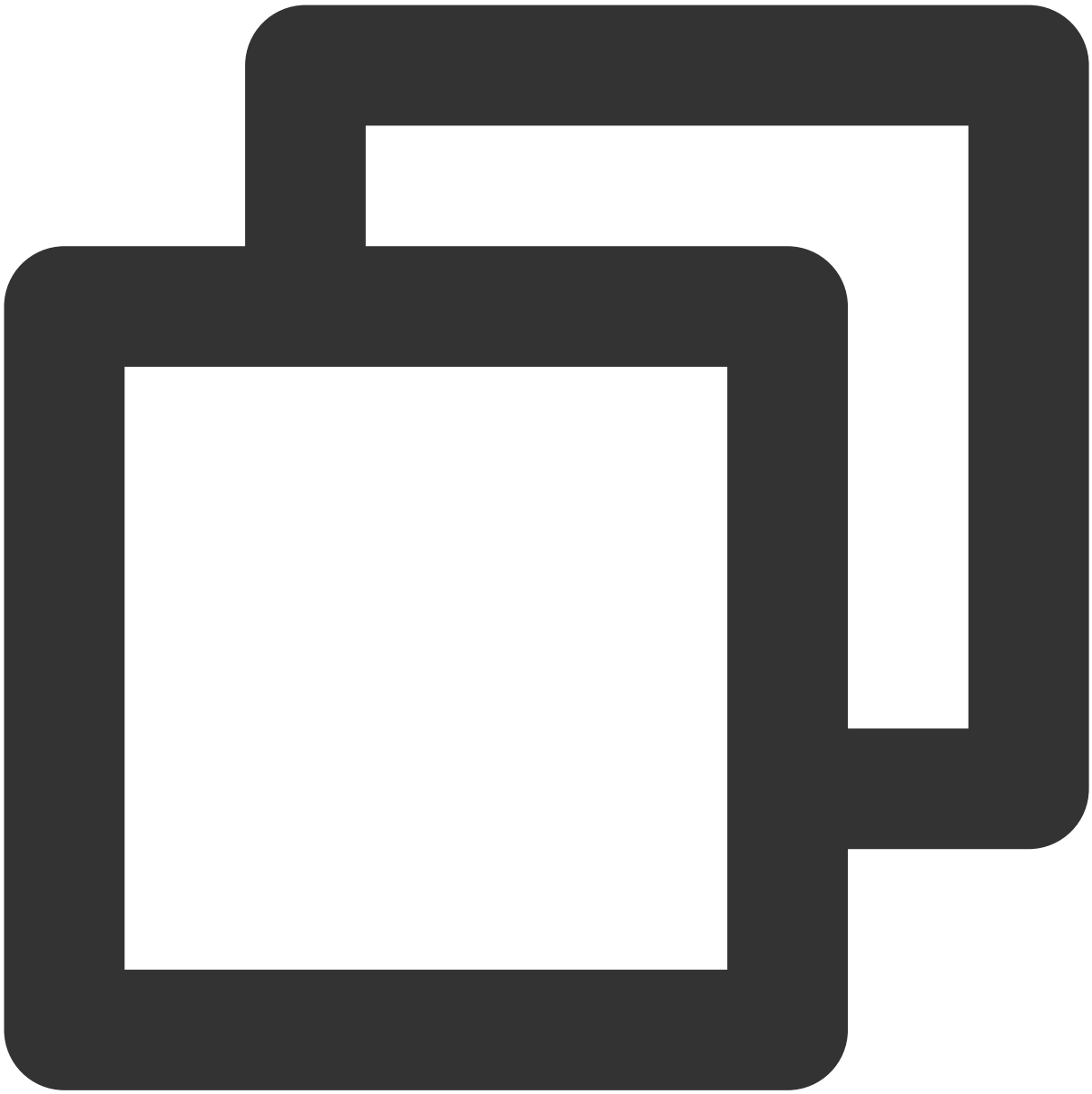
```
public abstract void muteRemoteAudio(String userId, boolean mute);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userId	String	指定ユーザーID。
mute	boolean	true：ミュート起動；false：ミュート停止。

muteAllRemoteAudio

全メンバーのミュート/ミュート解除。



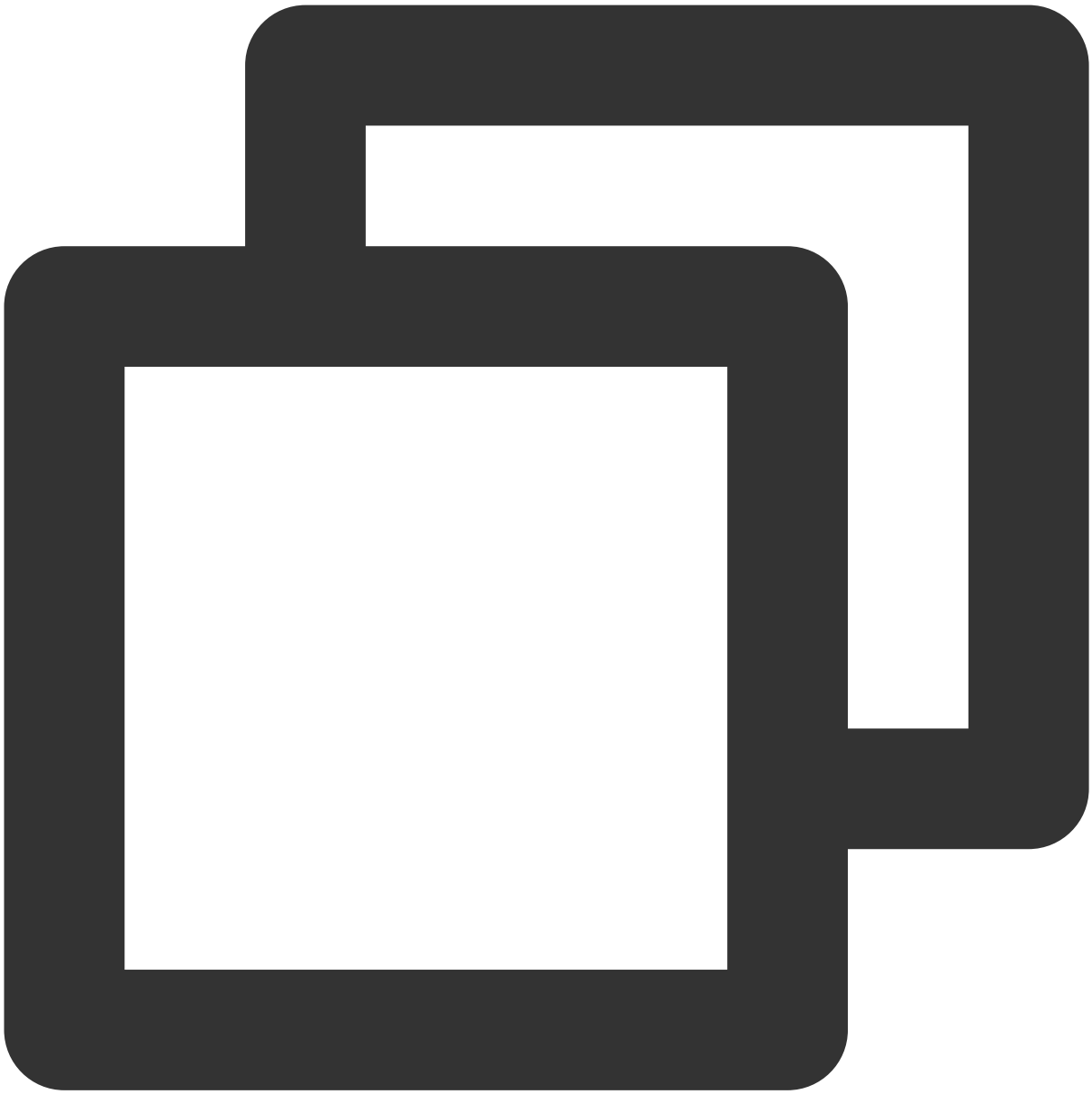
```
public abstract void muteAllRemoteAudio(boolean mute);
```

パラメータは下表に示すとおりです：

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

setVoiceEarMonitorEnable

インイヤーモニタリングのオン/オフ。



```
public abstract void setVoiceEarMonitorEnable(boolean enable);
```

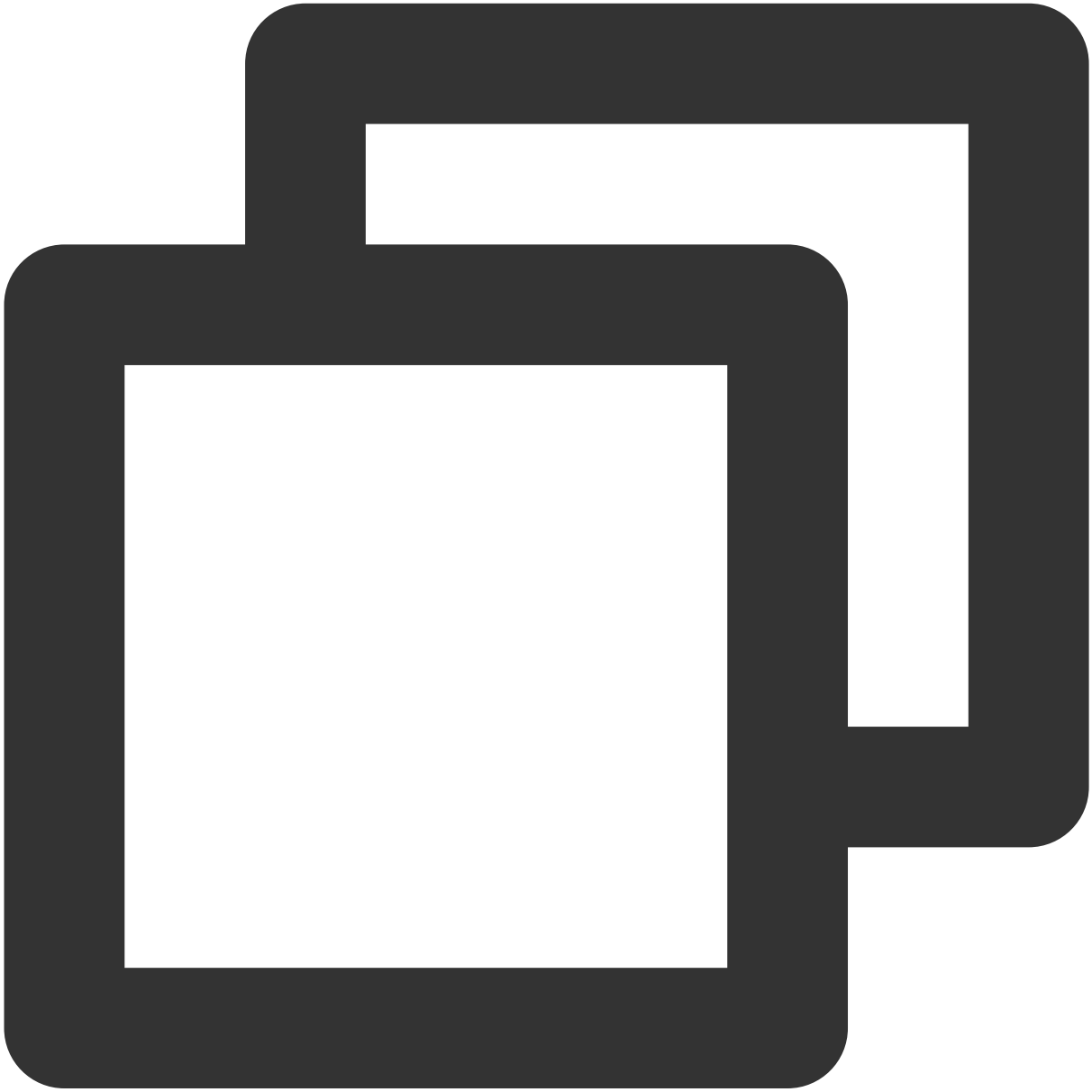
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
enable	boolean	true：インイヤーモニタリングをオン。false：インイヤーモニタリングをオフ。

BGMサウンドエフェクト関連インターフェース関数

getAudioEffectManager

BGMサウンドエフェクト管理オブジェクト TXAudioEffectManagerを取得します。

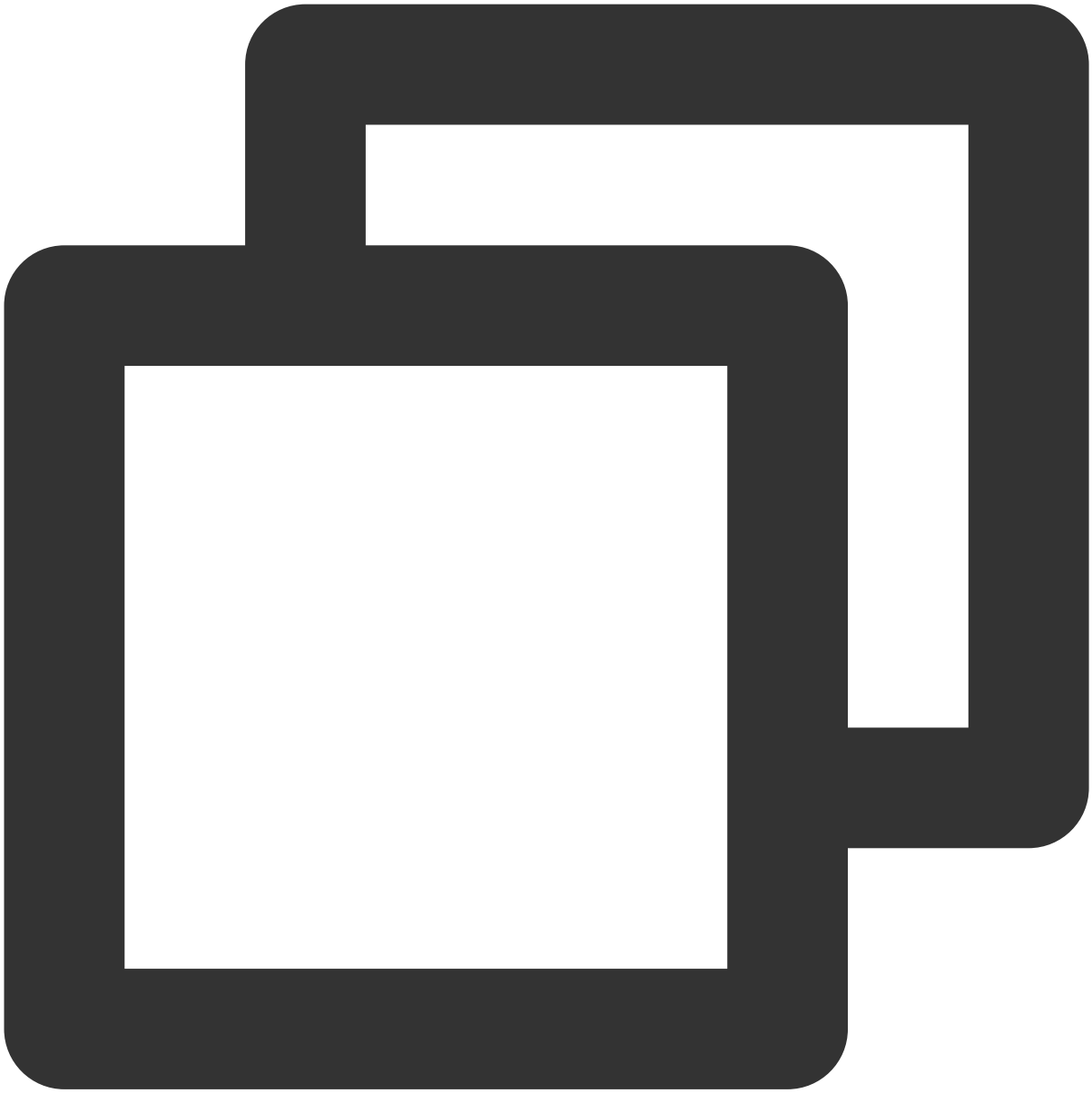


```
public abstract TXAudioEffectManager getAudioEffectManager();
```

メッセージ送信関連インターフェース関数

sendRoomTextMsg

ルーム内でテキストメッセージをブロードキャストします。通常、弾幕によるチャットに使用します。



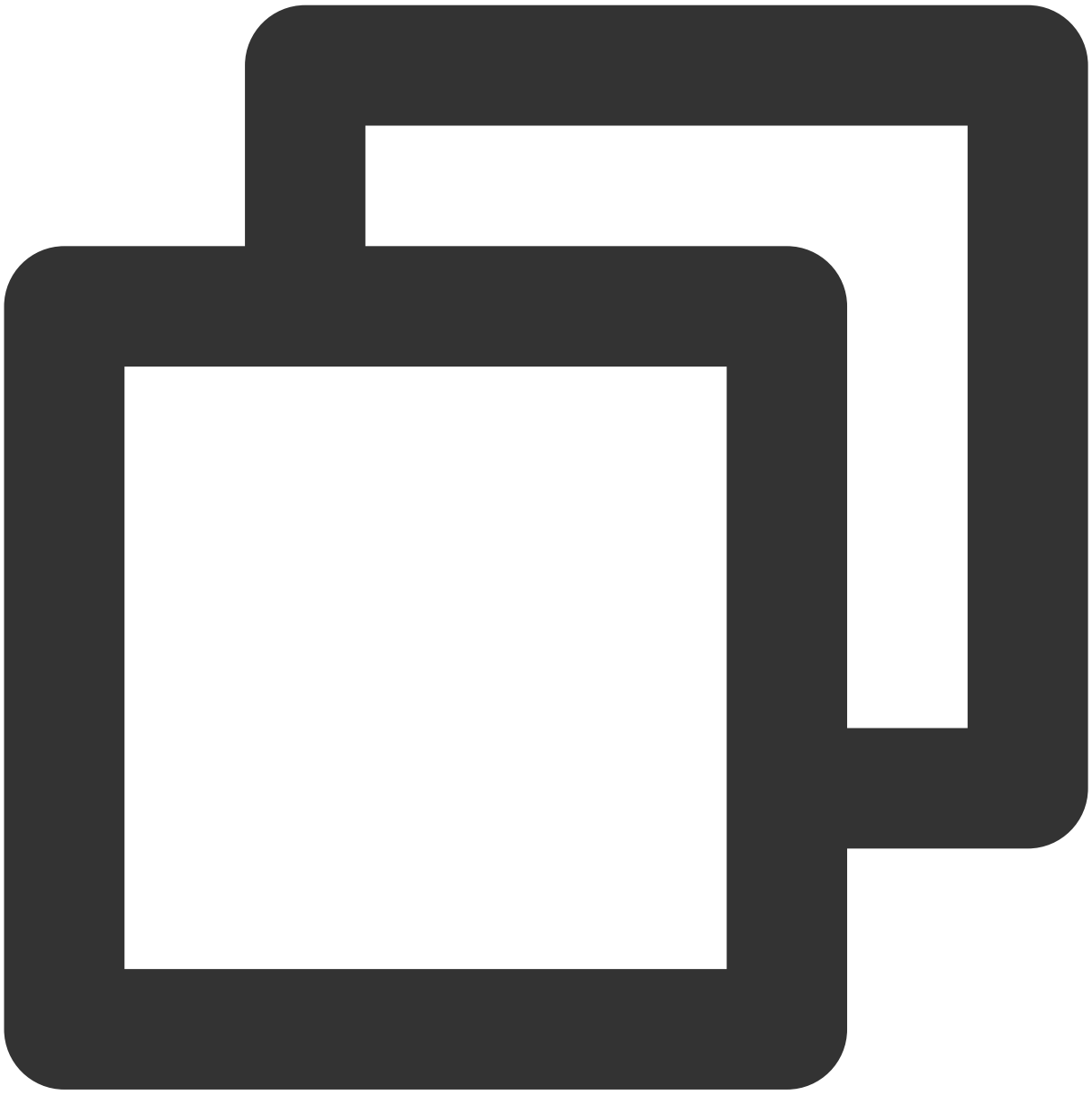
```
public abstract void sendRoomTextMsg(String message, TRTCKaraokeRoomCallback.Action
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

sendRoomCustomMsg

カスタマイズしたテキストメッセージを送信します。



```
public abstract void sendRoomCustomMsg(String cmd, String message, TRTCKaraokeRoomC
```

パラメータは下表に示すとおりです：

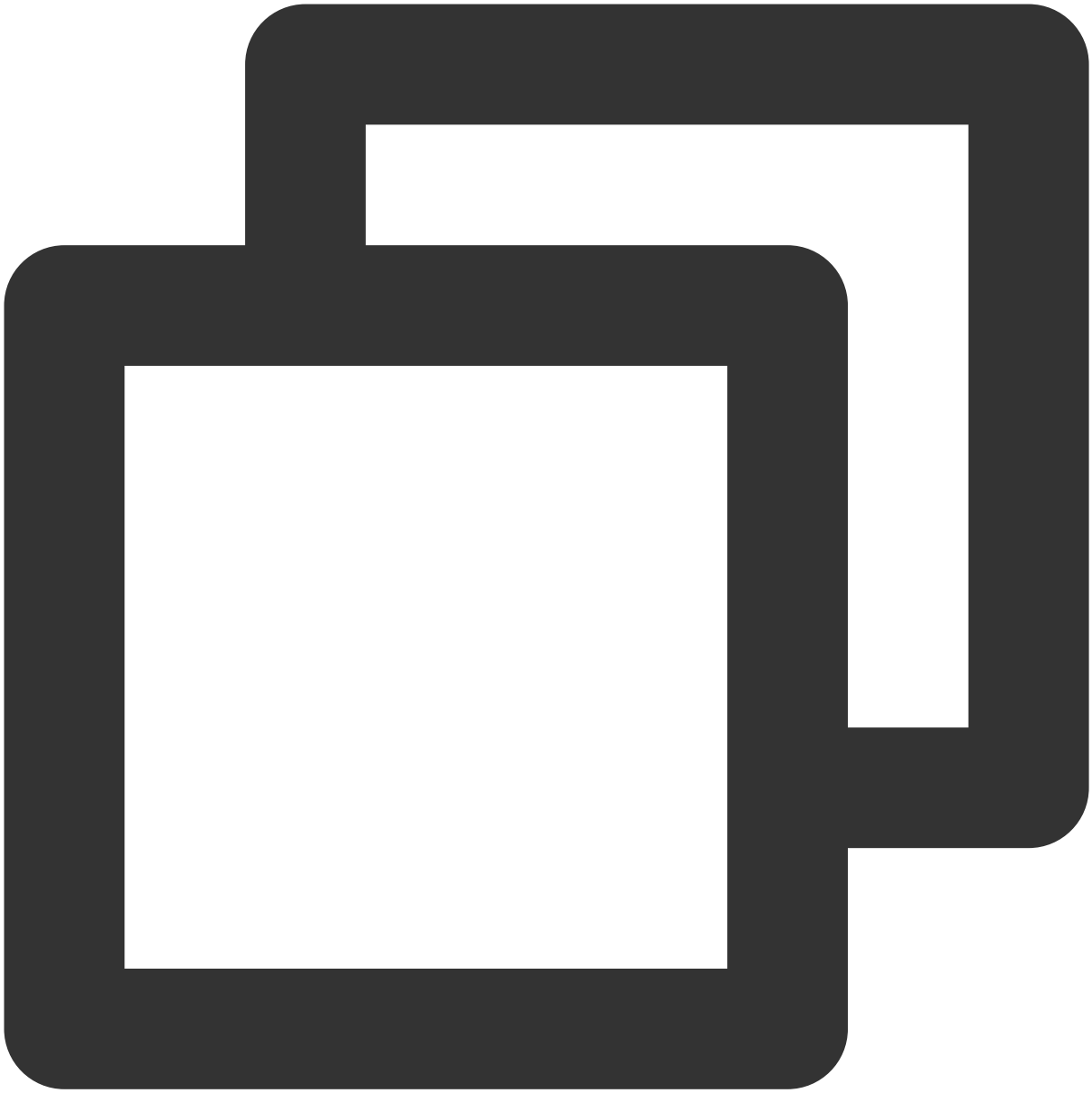
パラメータ	タイプ	意味

cmd	String	コマンドワードです。開発者がカスタマイズするものであり、主にさまざまなメッセージタイプを区別するために使用されます。
message	String	テキストメッセージ。
callback	ActionCallback	送信結果のコールバック。

招待シグナリング関連インターフェース

sendInvitation

ユーザーに招待を送信。



```
public abstract String sendInvitation(String cmd, String userId, String content, TR
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
cmd	String	業務カスタマイズコマンド。
userId	String	招待ユーザーID。
content	String	招待コンテンツ。

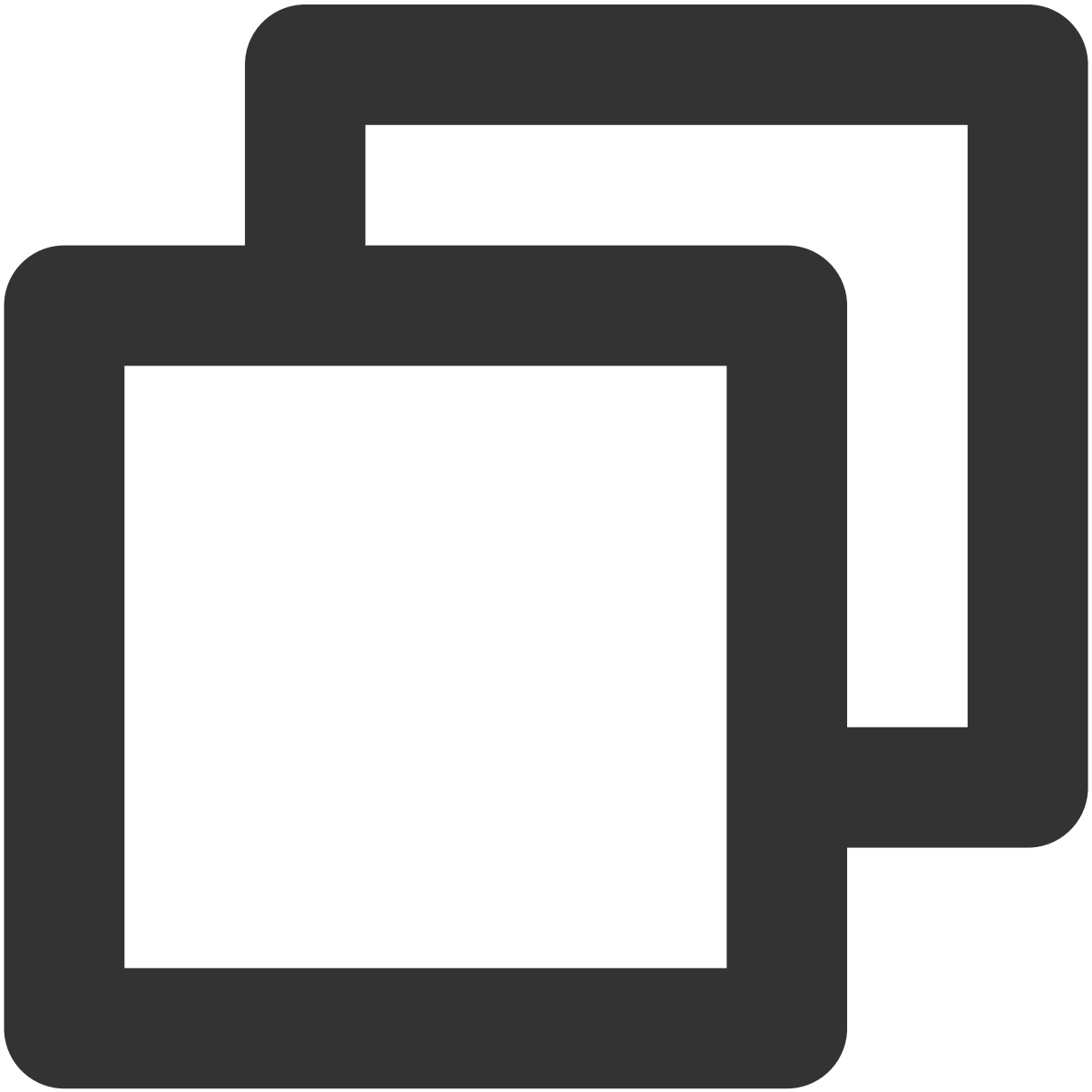
callback	ActionCallback	送信結果のコールバック。

戻り値：

戻り値	タイプ	意味
inviteId	String	今回の招待IDの識別に使用。

acceptInvitation

招待の同意。



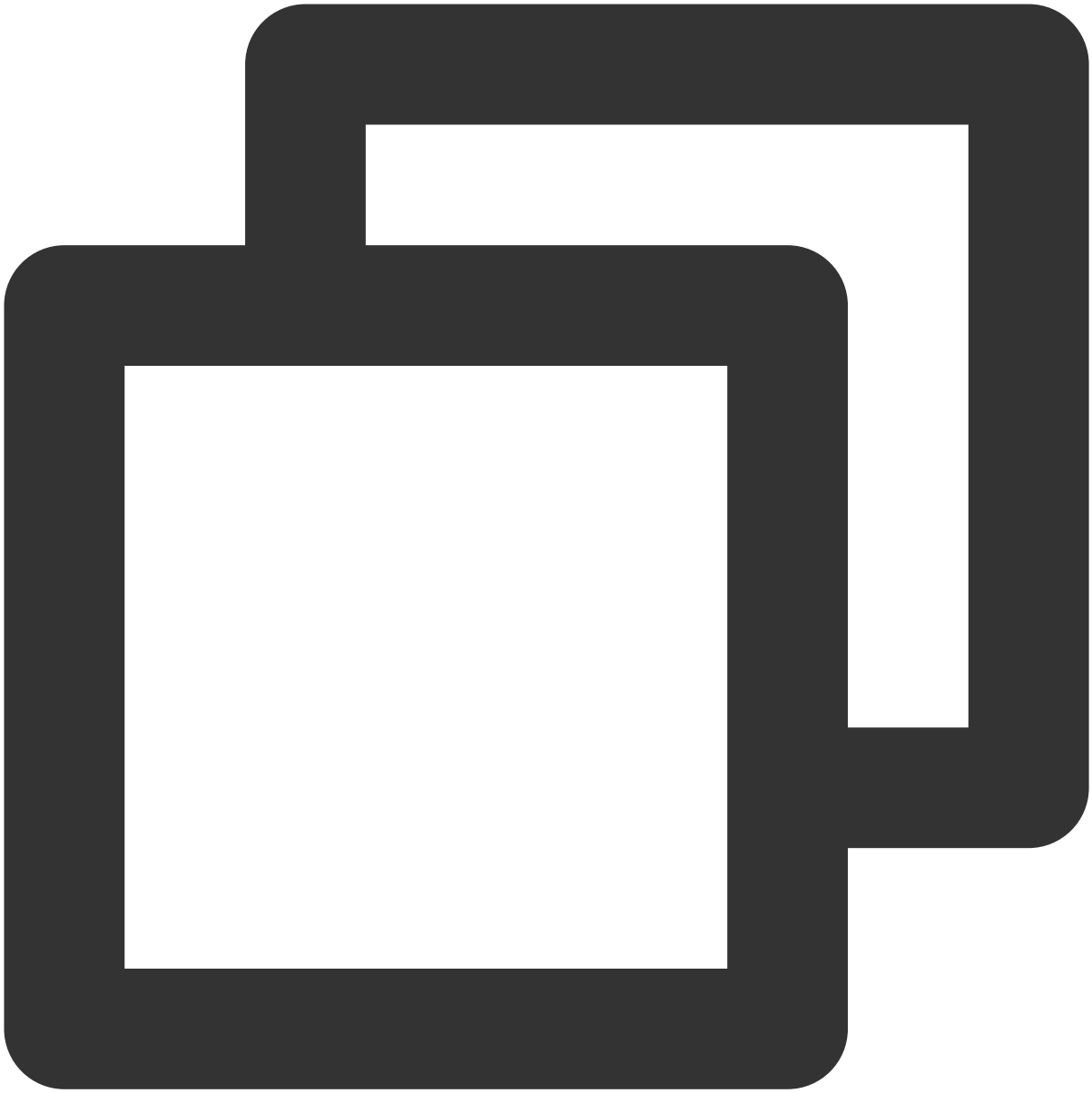
```
public abstract void acceptInvitation(String id, TRTCKaraokeRoomCallback.ActionCall
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

rejectInvitation

招待の拒否。



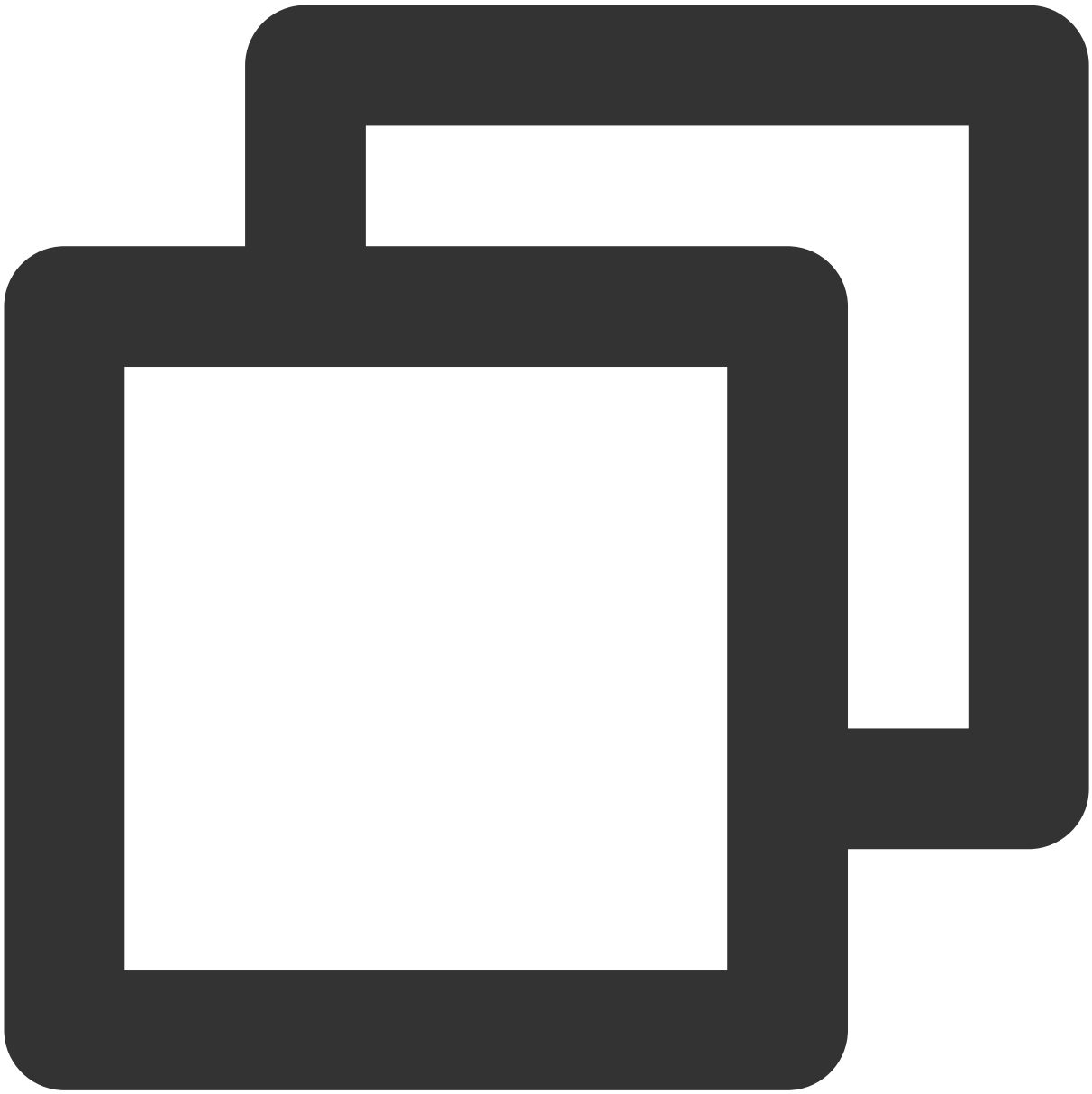
```
public abstract void rejectInvitation(String id, TRTCKaraokeRoomCallback.ActionCall
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

cancelInvitation

招待の取り消し。



```
public abstract void cancelInvitation(String id, TRTCKaraokeRoomCallback.ActionCall
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
callback	ActionCallback	送信結果のコールバック。

TRTCKaraokeRoomDelegateイベントコールバック

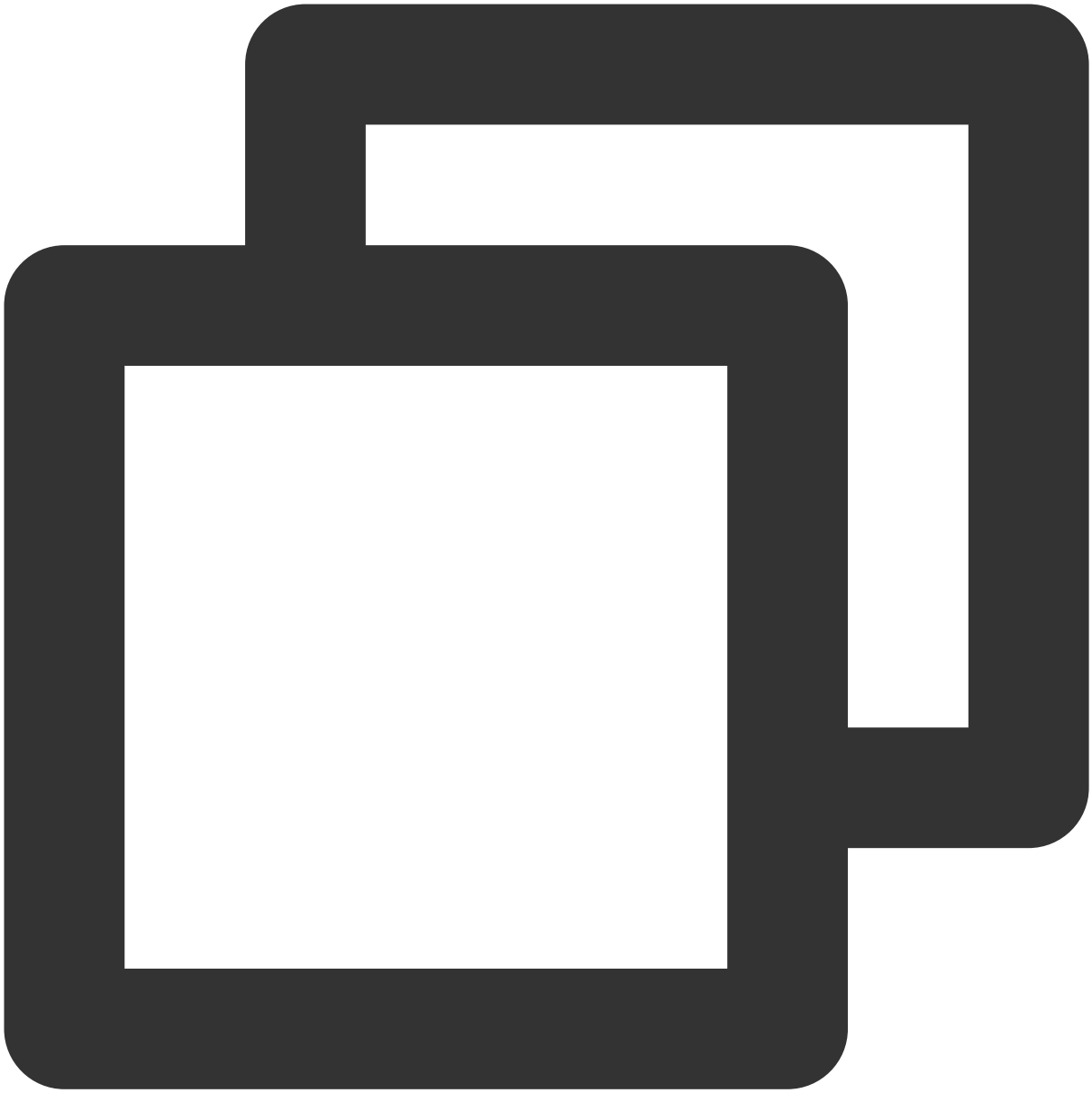
一般的なイベントコールバック

onError

エラーのコールバック。

説明：

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



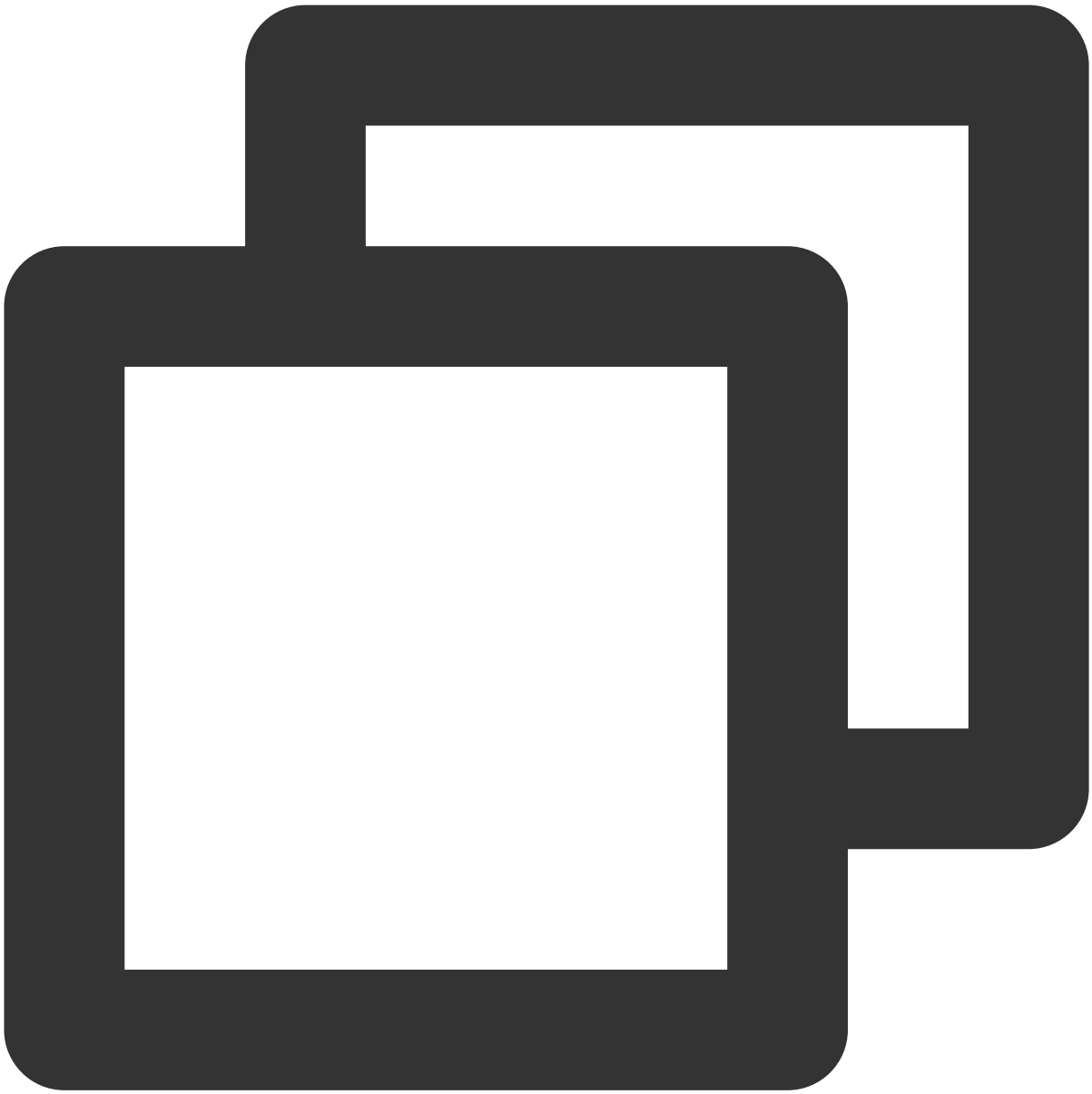
```
void onError(int code, String message);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	エラーメッセージ。

onWarning

警告のコールバック。



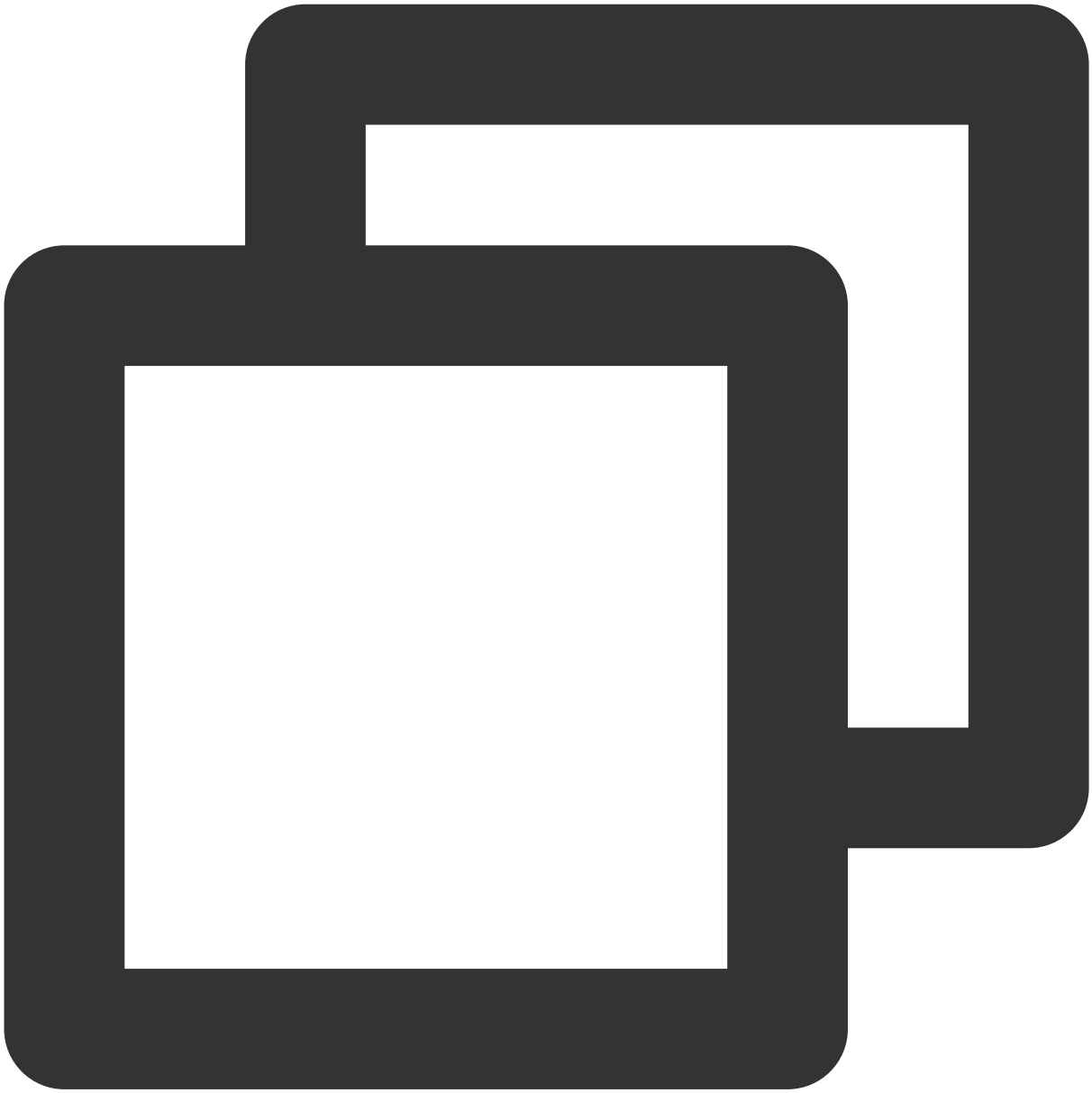
```
void onWarning(int code, String message);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
code	int	エラーコード。
message	String	警告メッセージ。

onDebugLog

Logコールバック。



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

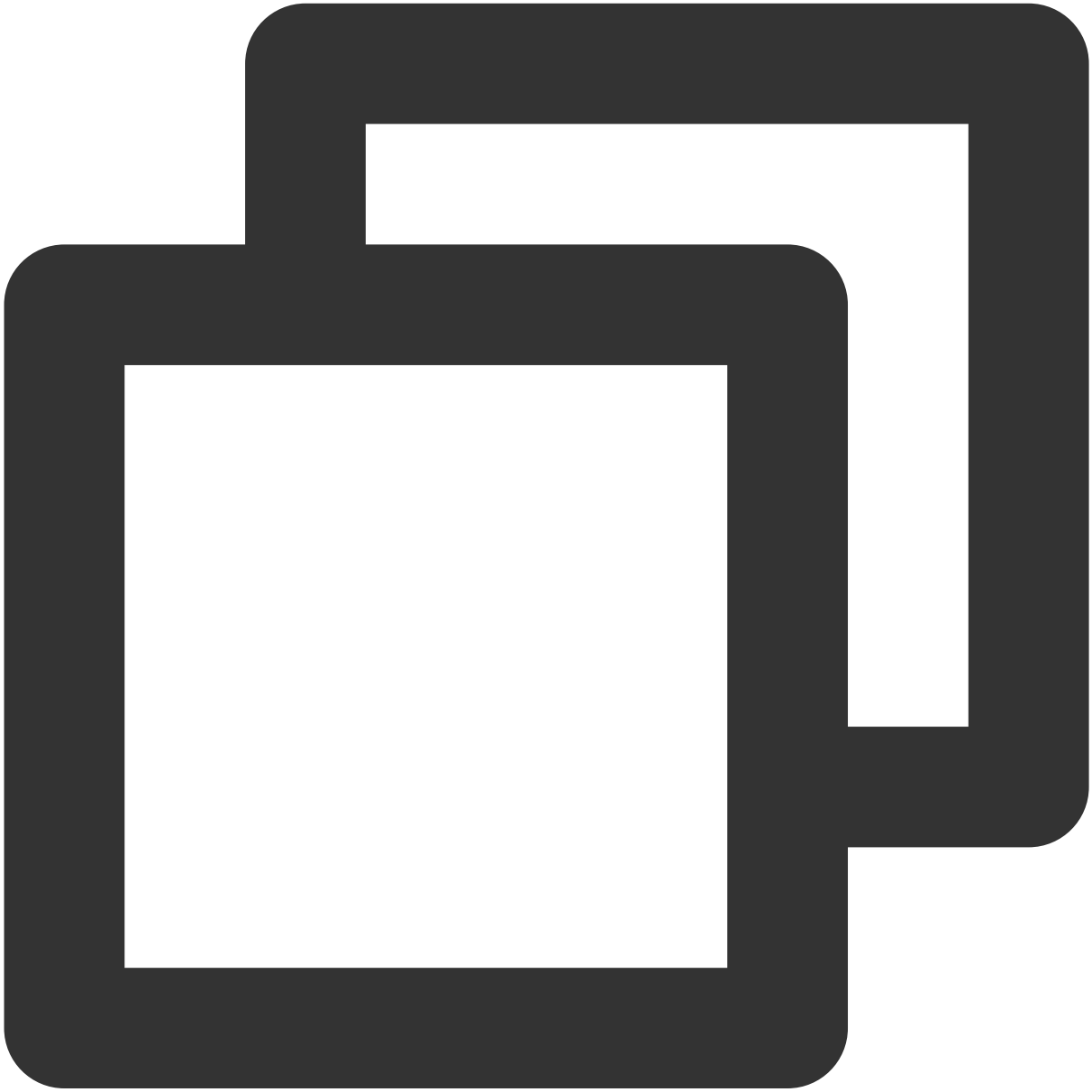
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
message	String	ログ情報。

ルームイベントのコールバック

onRoomDestroy

ルーム破棄のコールバック。管理者がルームを解散するとき、ルーム内の全ユーザーはこの通知を受信します。



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

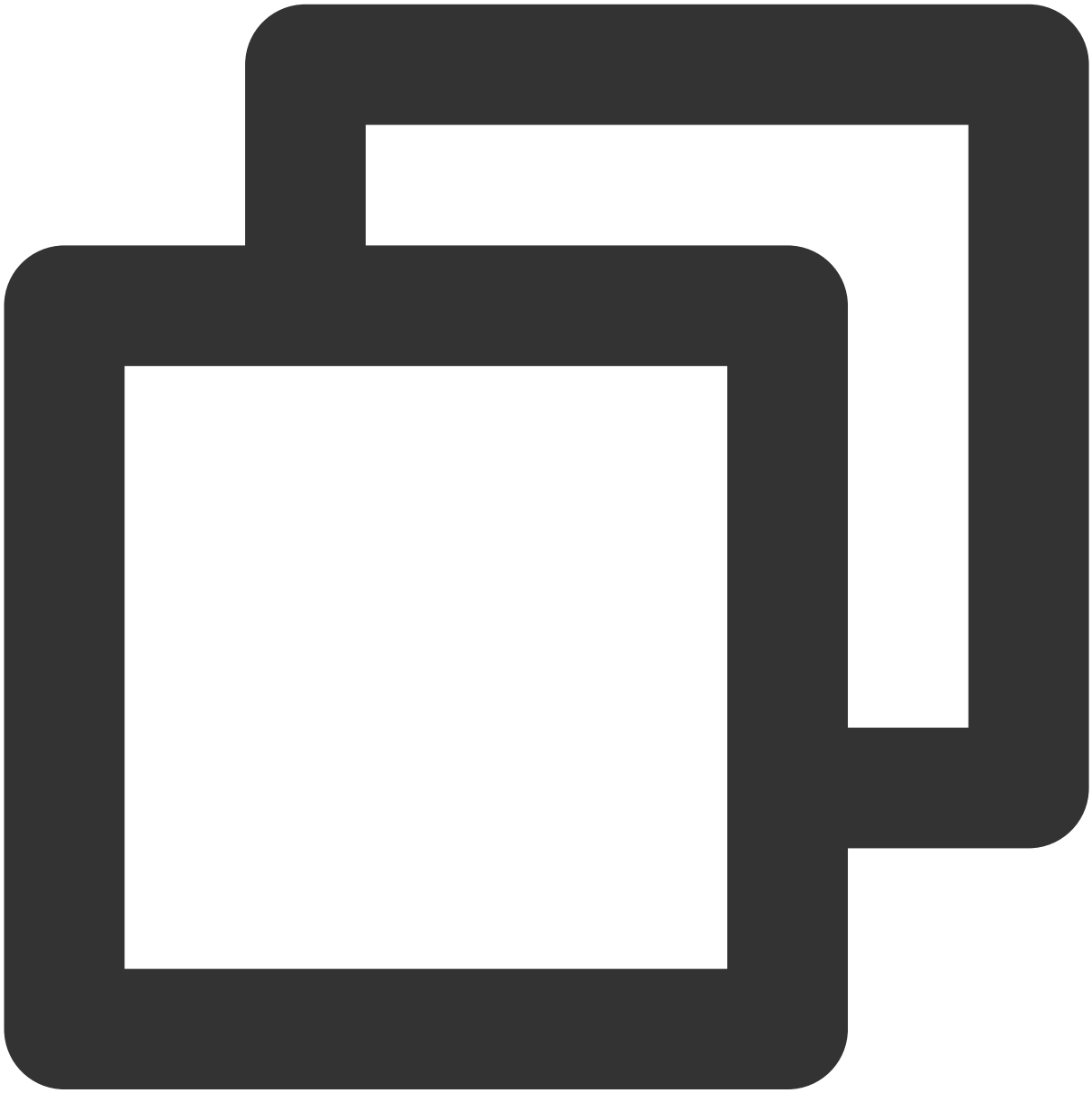
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味

roomId	String	ルームID。

onRoomInfoChange

入室に成功後、このインターフェースをコールバックします。roomInfoの情報は、管理者がルームを作成するときに渡されます。



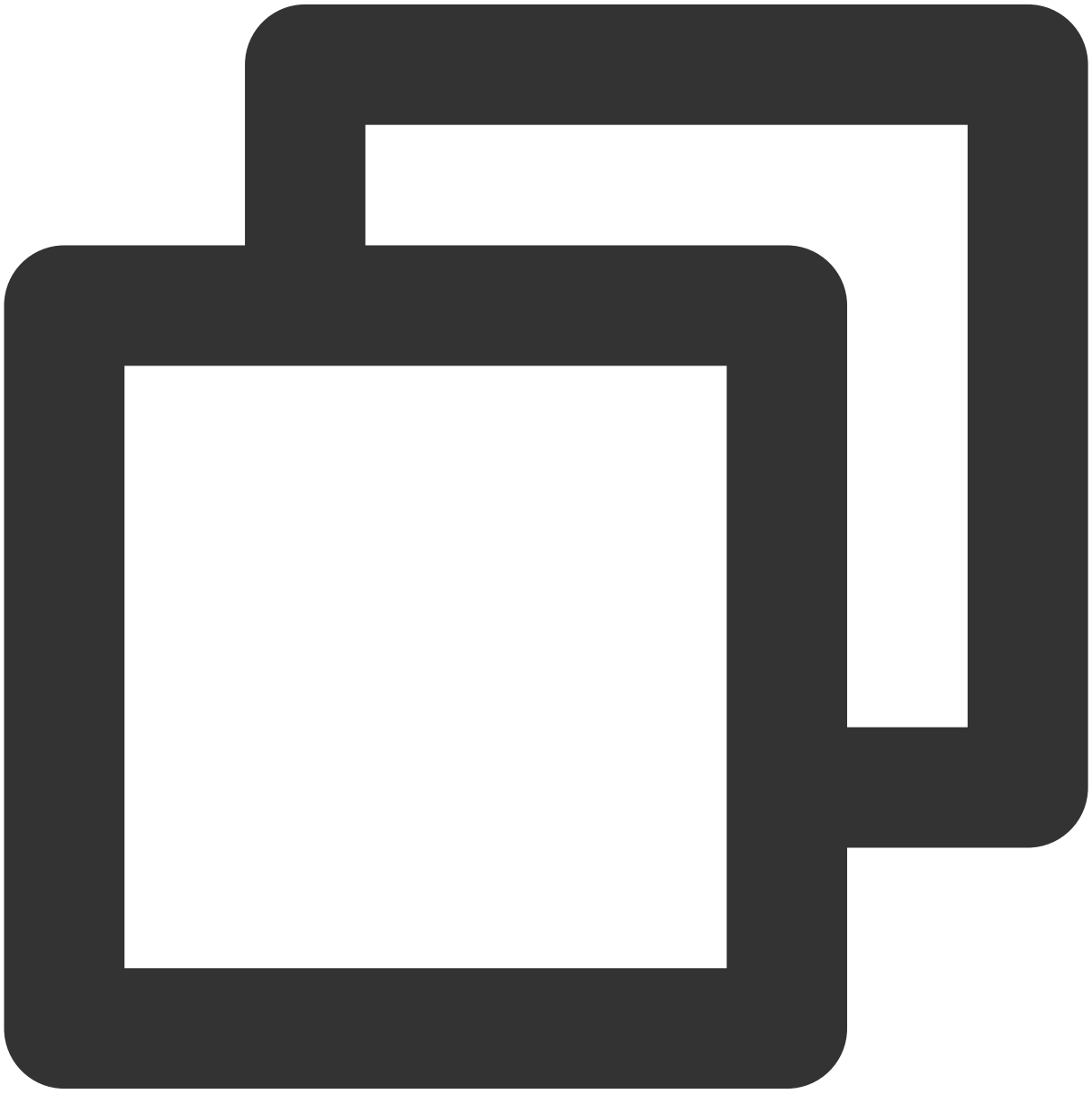
```
void onRoomInfoChange(TRTCKaraokeRoomDef.RoomInfo roomInfo);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
roomInfo	RoomInfo	ルーム情報。

onUserMicrophoneMute

ユーザーのマイクがミュートになっているかどうかについて、ユーザーがmuteLocalAudioを呼び出すと、ルームの他のユーザーがこの通知を受信します。



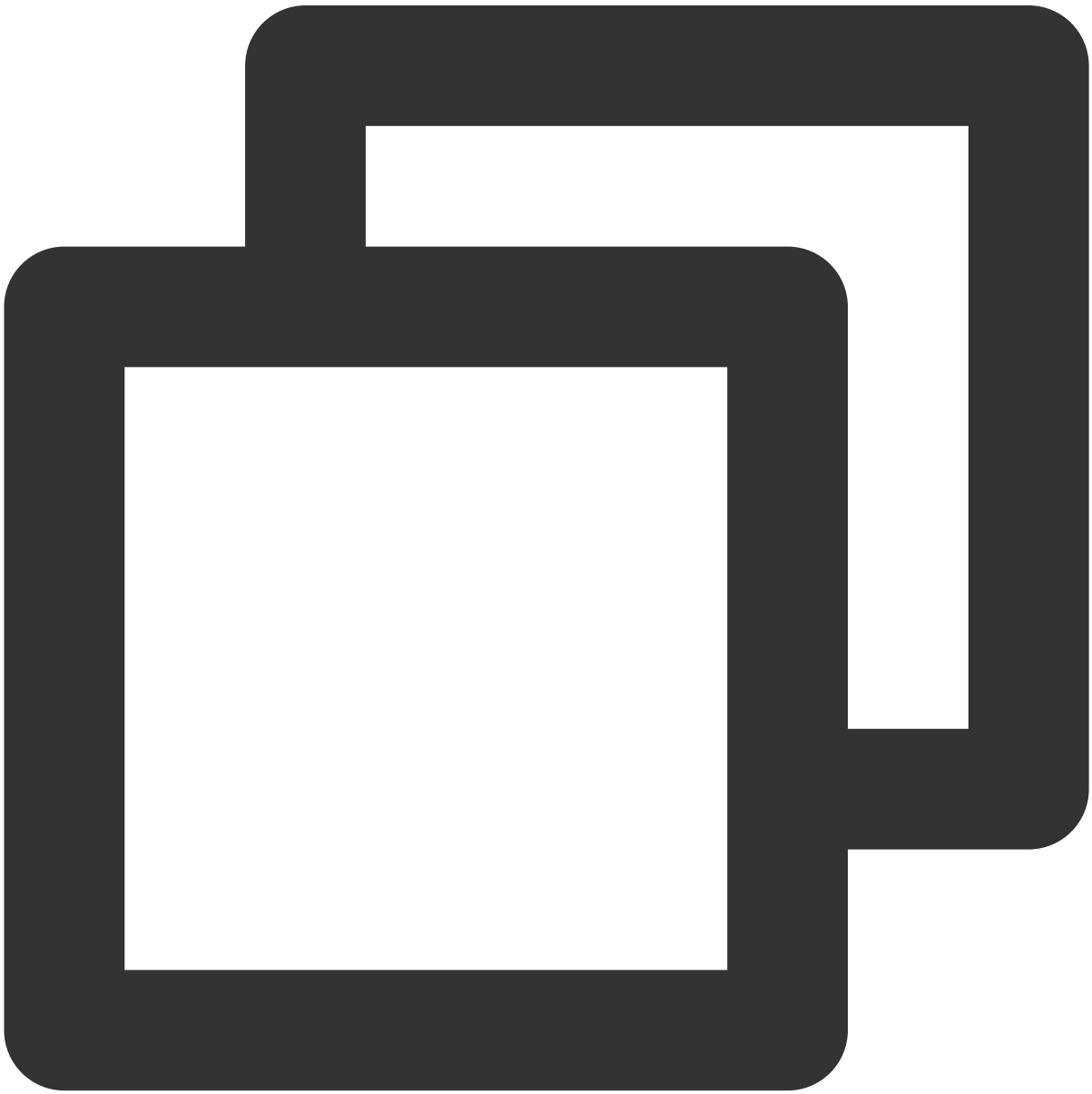
```
void onUserMicrophoneMute(String userId, boolean mute);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userId	String	ユーザーID。
mute	boolean	音量の大きさ。値：0～100。

onUserVolumeUpdate

音量レベルリマインダを有効にすると、各メンバーの音量を通知します。



```
void onUserVolumeUpdate(List<TRTCCloudDef.TRTCVolumeInfo> userVolumes, int totalVol
```

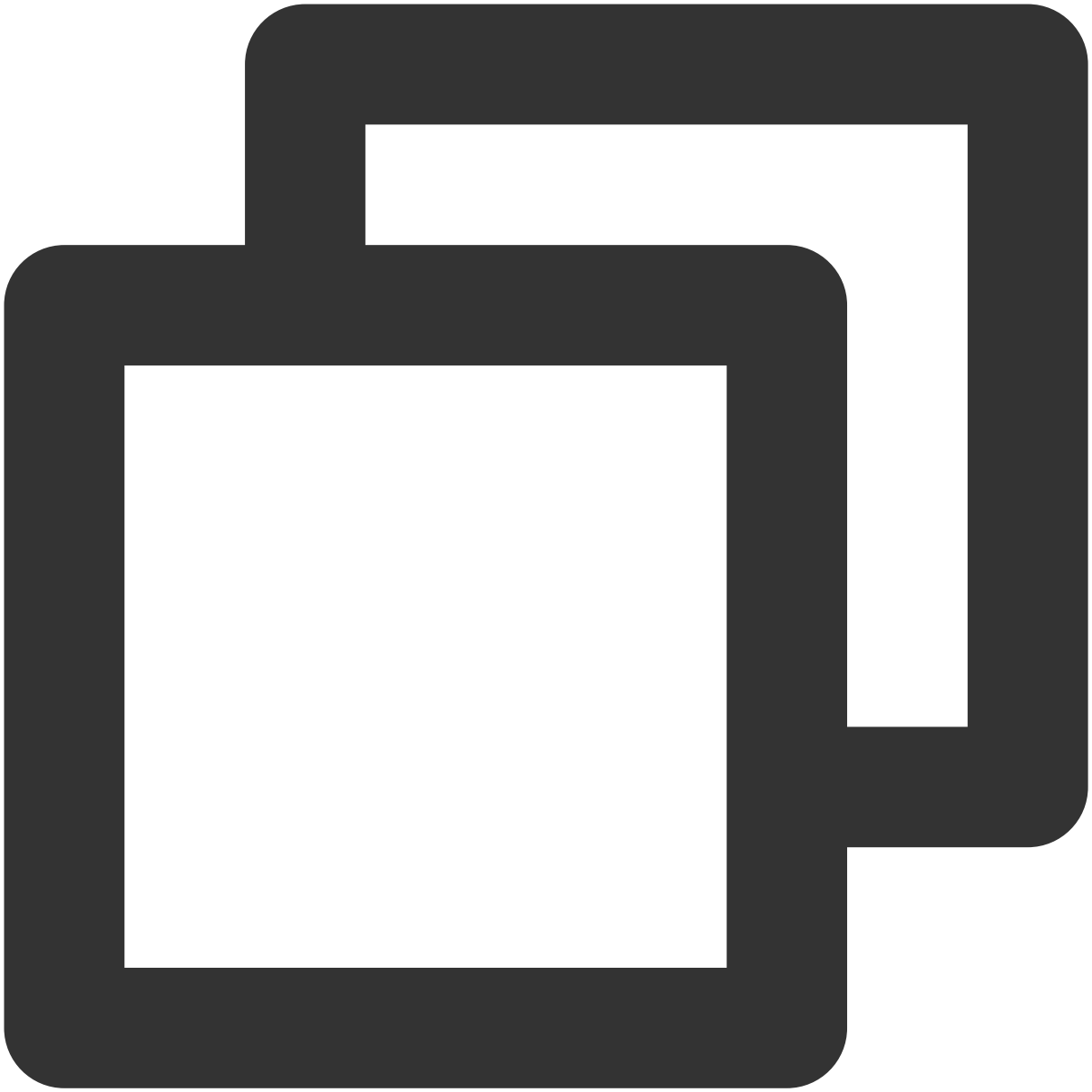
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userVolumes	List	ユーザーリスト。
totalVolume	int	音量の大きさ。値：0～100。

マイクコールバック

onSeatListChange

全量のマイクリストの変更です。全てのマイクリストを含みます。



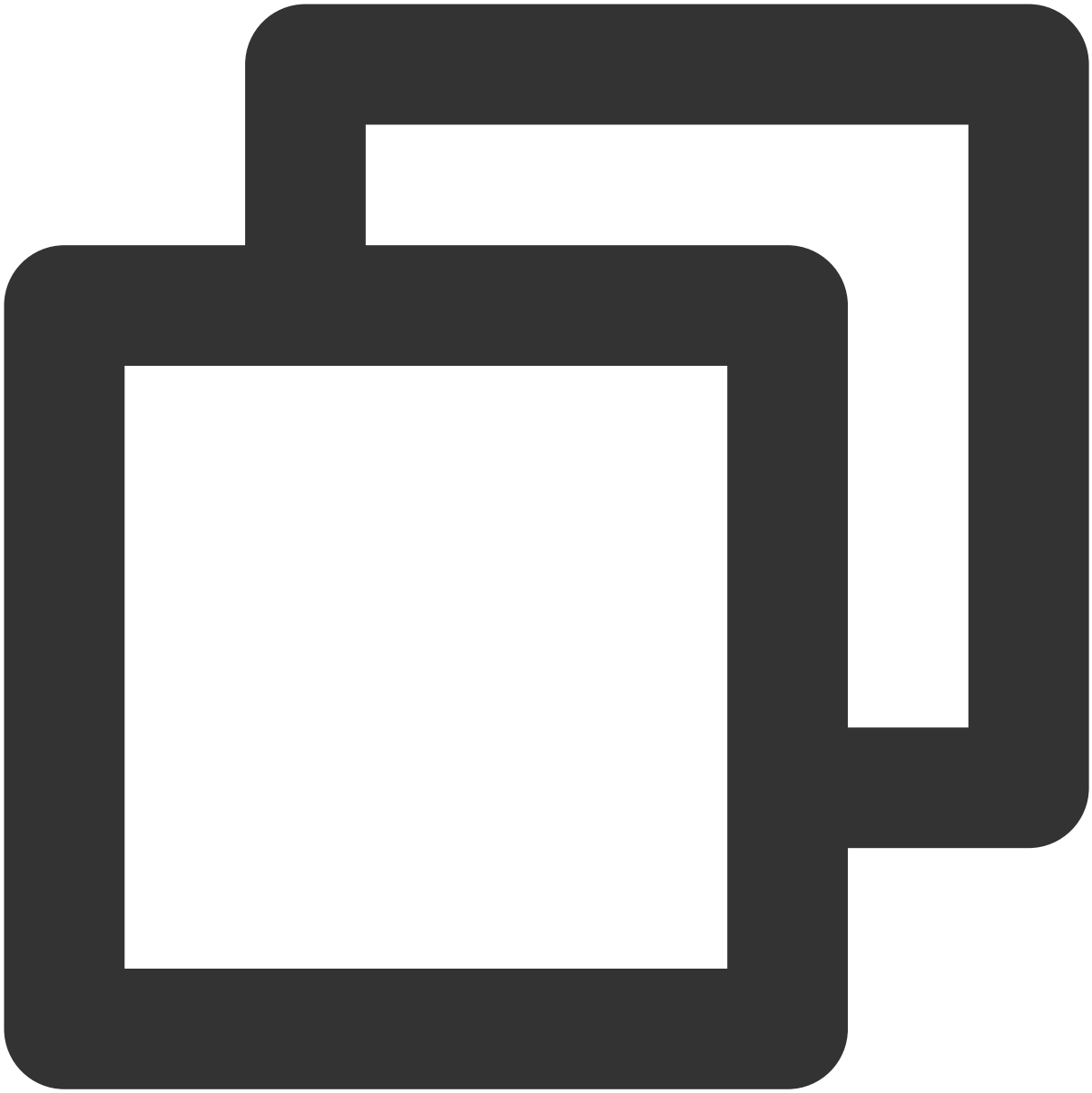
```
void onSeatListChange(List<SeatInfo> seatInfoList);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
seatInfoList	List<SeatInfo>	全量のマイクリスト。

onAnchorEnterSeat

発言者のメンバーがいます（ユーザーが発言者になる/管理者が視聴者を発言できるように招待）。



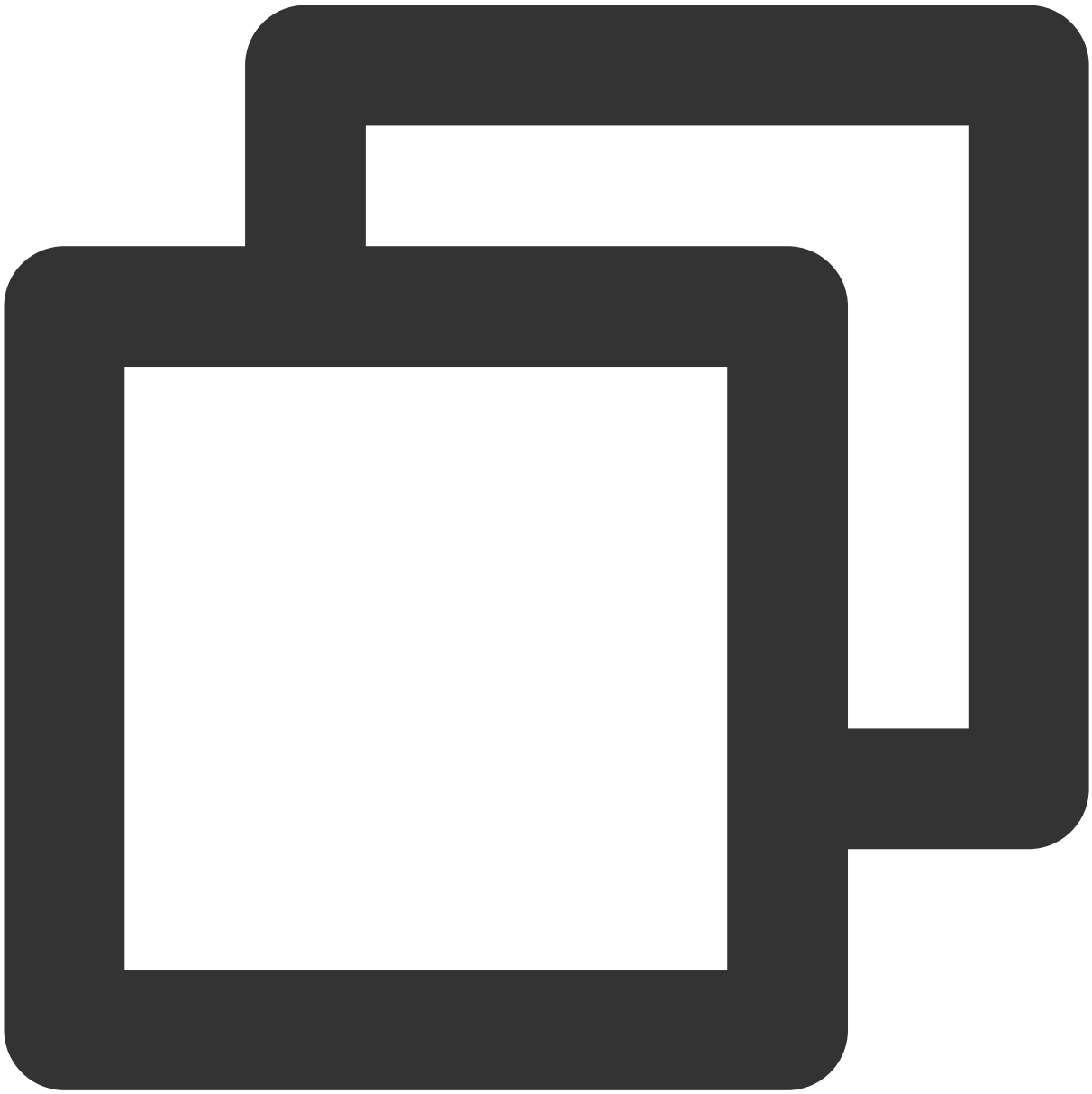
```
void onAnchorEnterSeat(int index, TRTCKaraokeRoomDef.UserInfo user);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
index	int	メンバーがマイク・オンのマイク。
user	UserInfo	マイク・オンのユーザーの詳細情報。

onAnchorLeaveSeat

視聴者のメンバーがいます（ユーザーが視聴者になる/管理者がキックアウトしてマイク・オフ）。



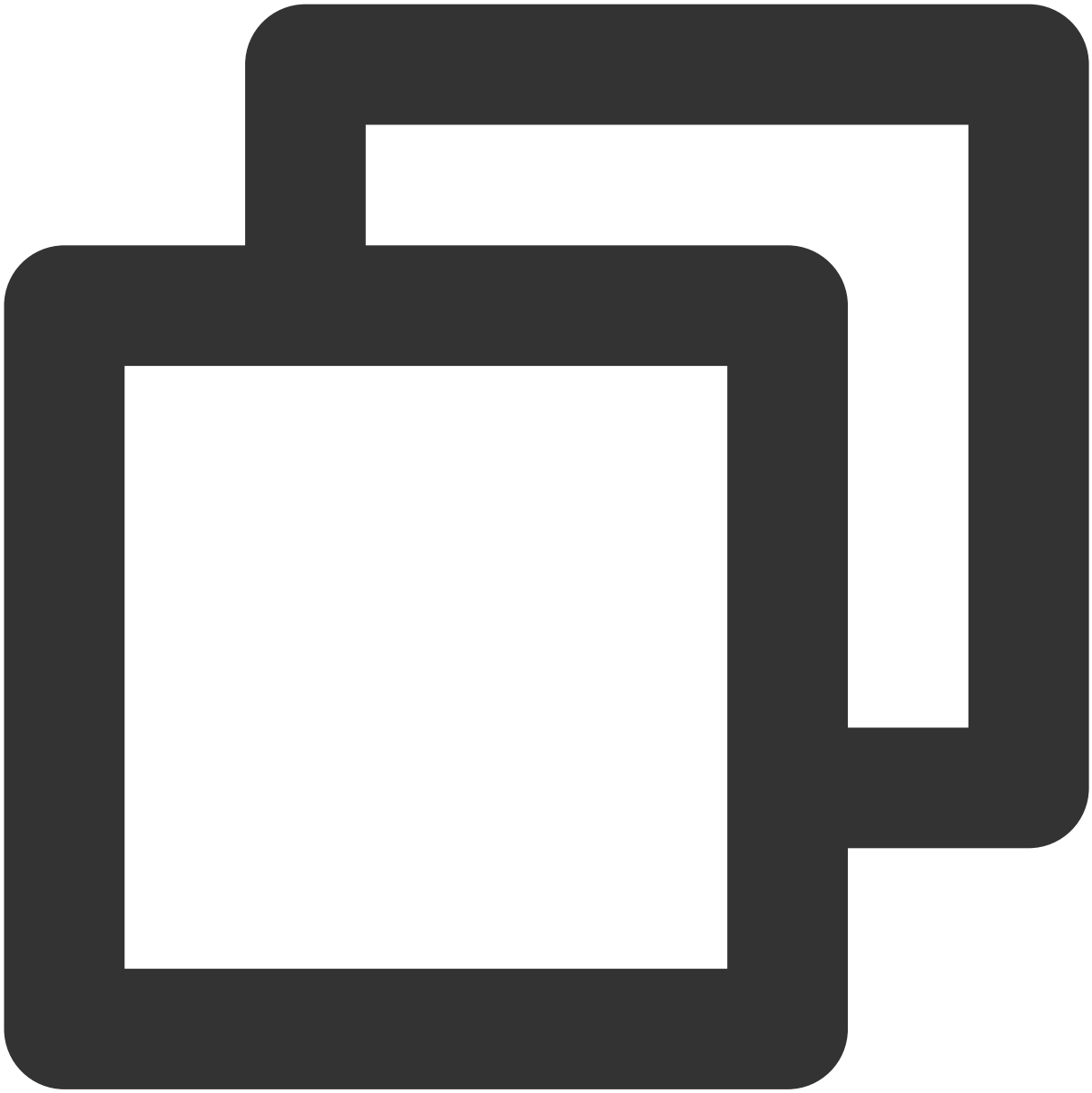
```
void onAnchorLeaveSeat(int index, TRTCKaraokeRoomDef.UserInfo user);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
index	int	マイク・オフのマイク。
user	UserInfo	マイク・オフのユーザーの詳細情報。

onSeatMute

管理者のマイクミュート。



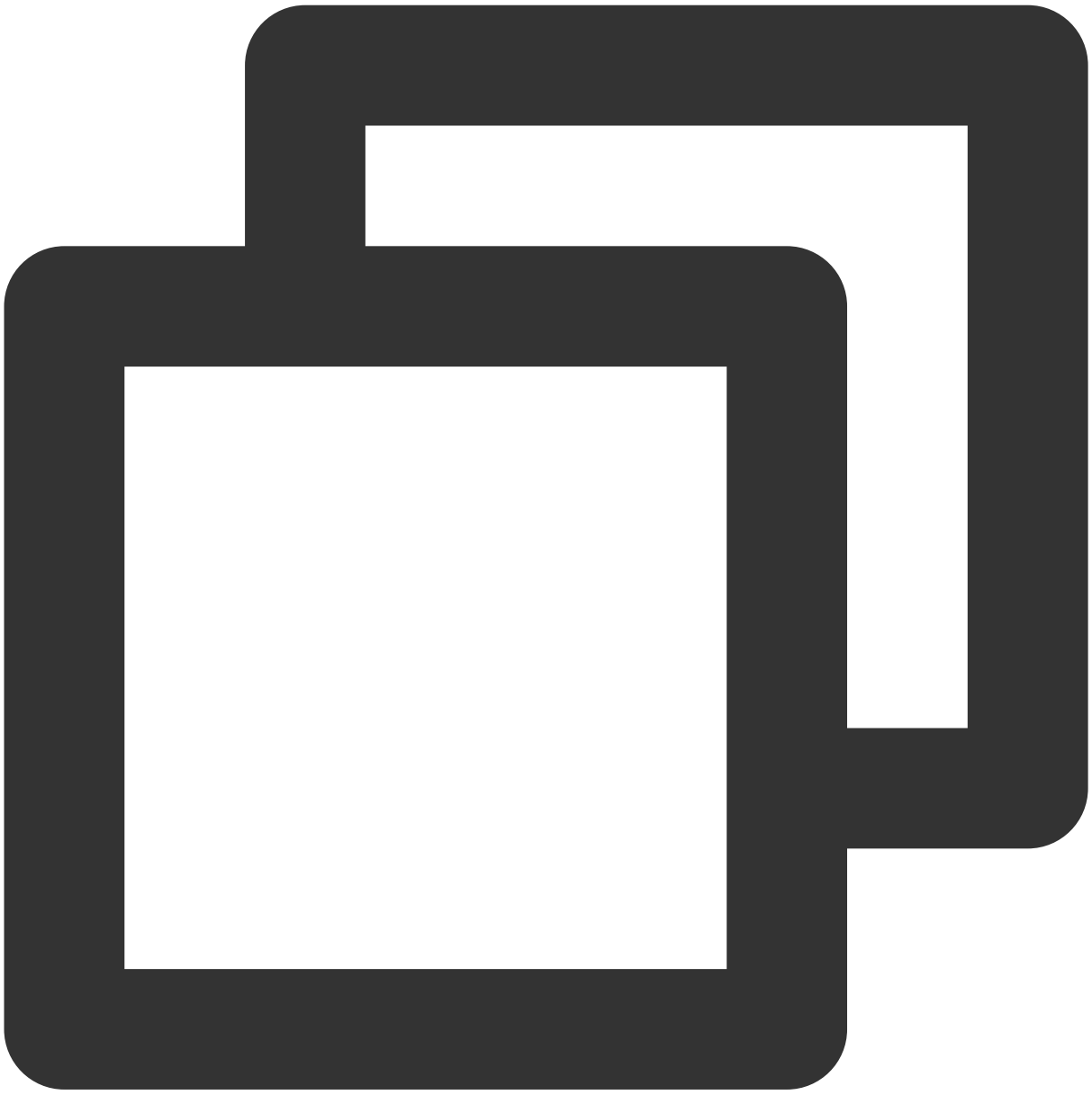
```
void onSeatMute(int index, boolean isMute);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
index	int	操作するマイク。
isMute	boolean	true：マイクミュート； false：ミュート解除。

onSeatClose

管理者のマイククローズ。



```
void onSeatClose(int index, boolean isClose);
```

パラメータは下表に示すとおりです：

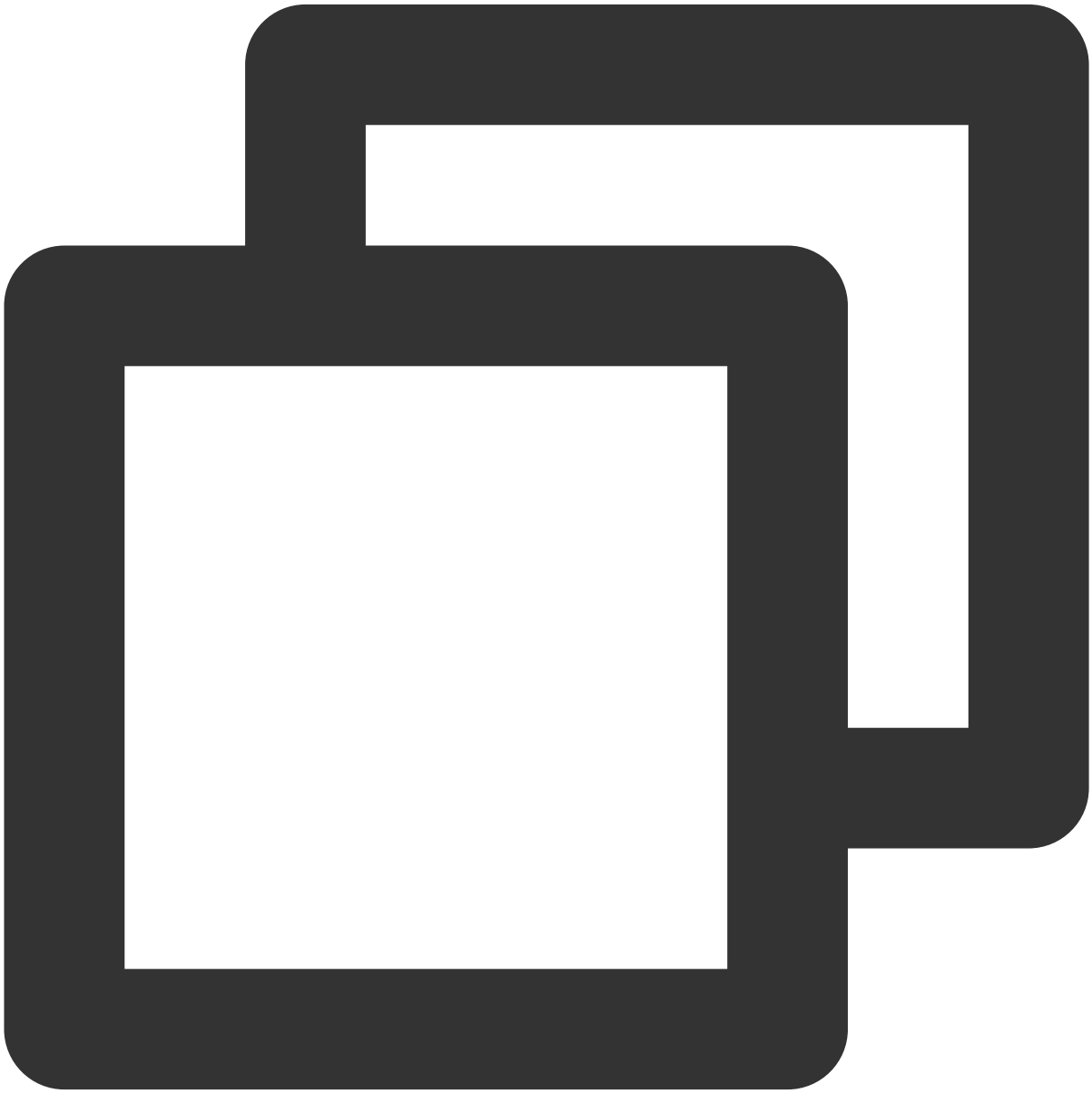
パラメータ	タイプ	意味
index	int	操作するマイク。

isClose	boolean	true：マイクのクローズ； false：マイクのクローズ解除。

リスナーの入退室イベントのコールバック

onAudienceEnter

リスナー入室通知の受信。



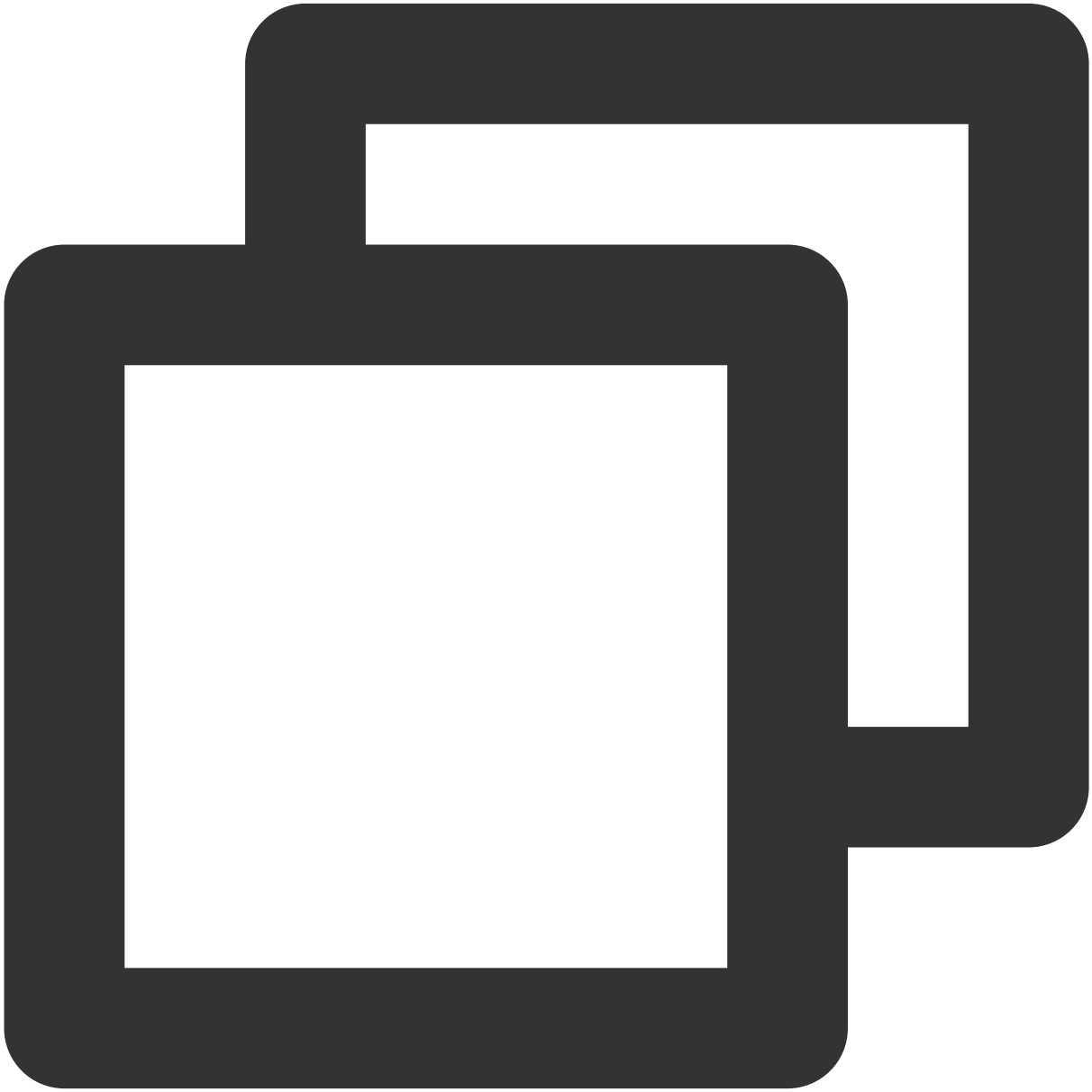
```
void onAudienceEnter(TRTCKaraokeRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userInfo	UserInfo	入室したリスナーの情報。

onAudienceExit

リスナー退室通知の受信。



```
void onAudienceExit(TRTCKaraokeRoomDef.UserInfo userInfo);
```

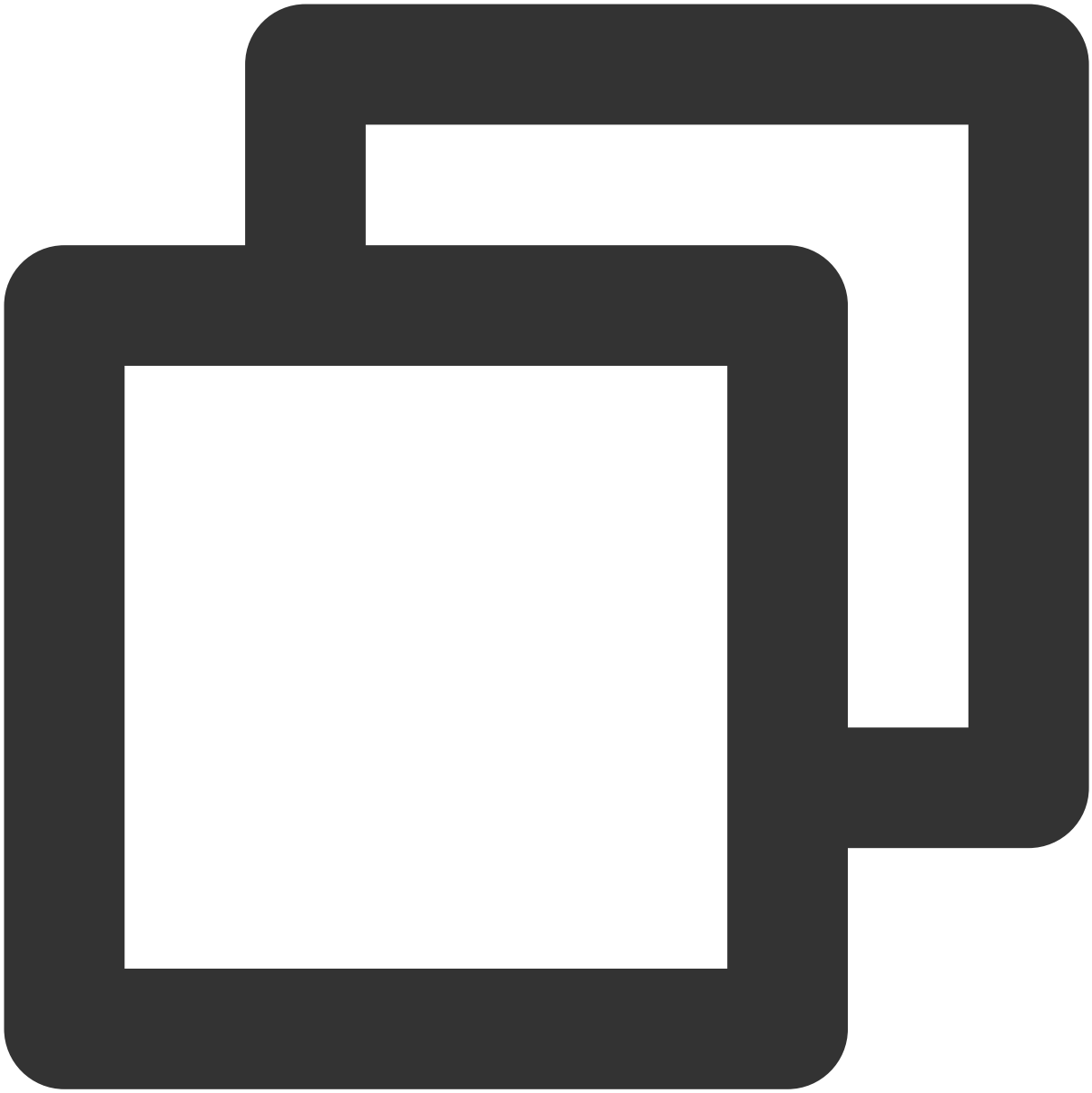
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
userInfo	UserInfo	退室したリスナーの情報。

メッセージイベントのコールバック

onRecvRoomTextMsg

テキストメッセージを受信します。



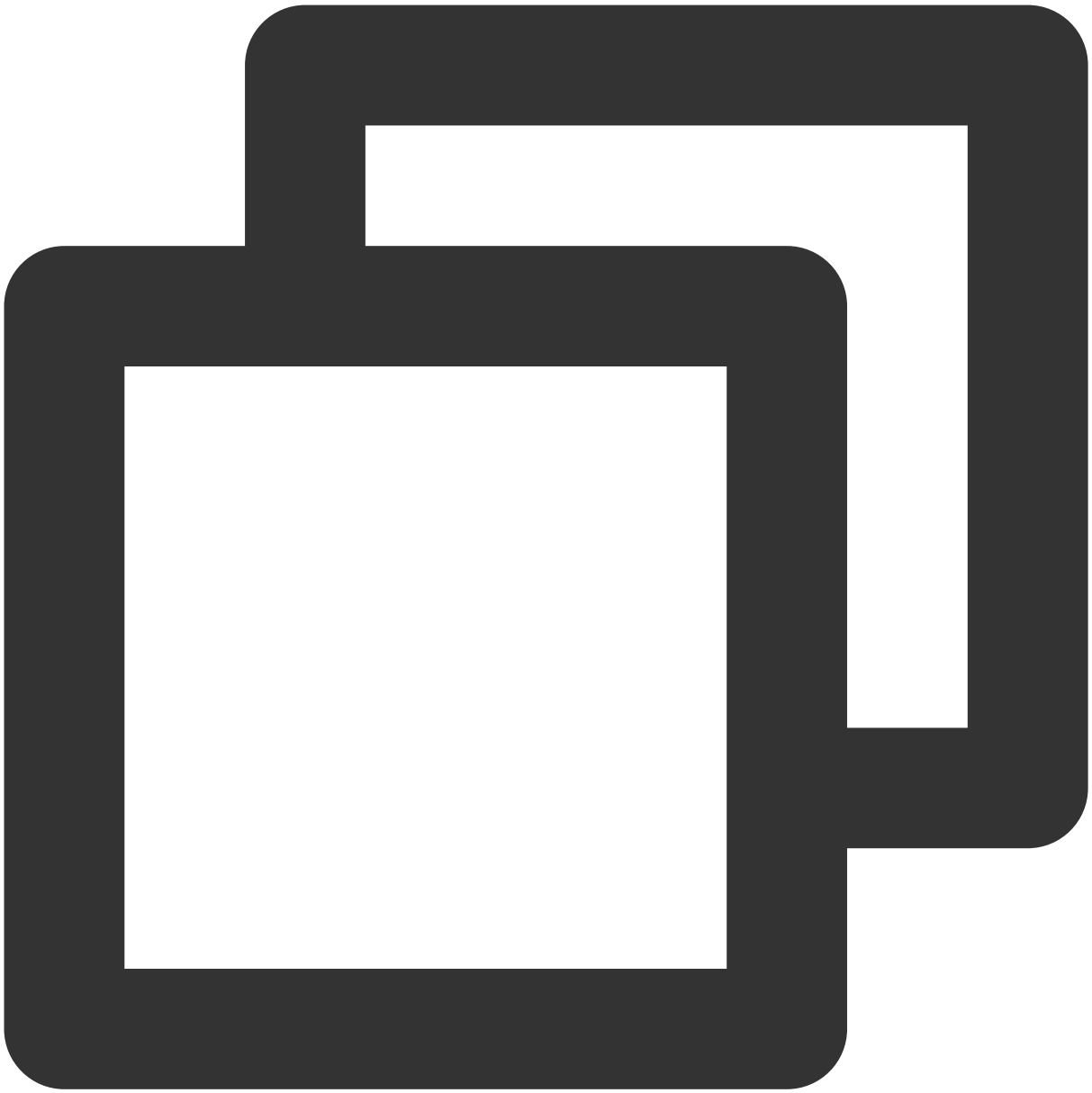
```
void onRecvRoomTextMsg(String message, TRTCKaraokeRoomDef.UserInfo userInfo);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

onRecvRoomCustomMsg

カスタムメッセージを受信します。



```
void onRecvRoomCustomMsg(String cmd, String message, TRTCKaraokeRoomDef.UserInfo us
```

パラメータは下表に示すとおりです：

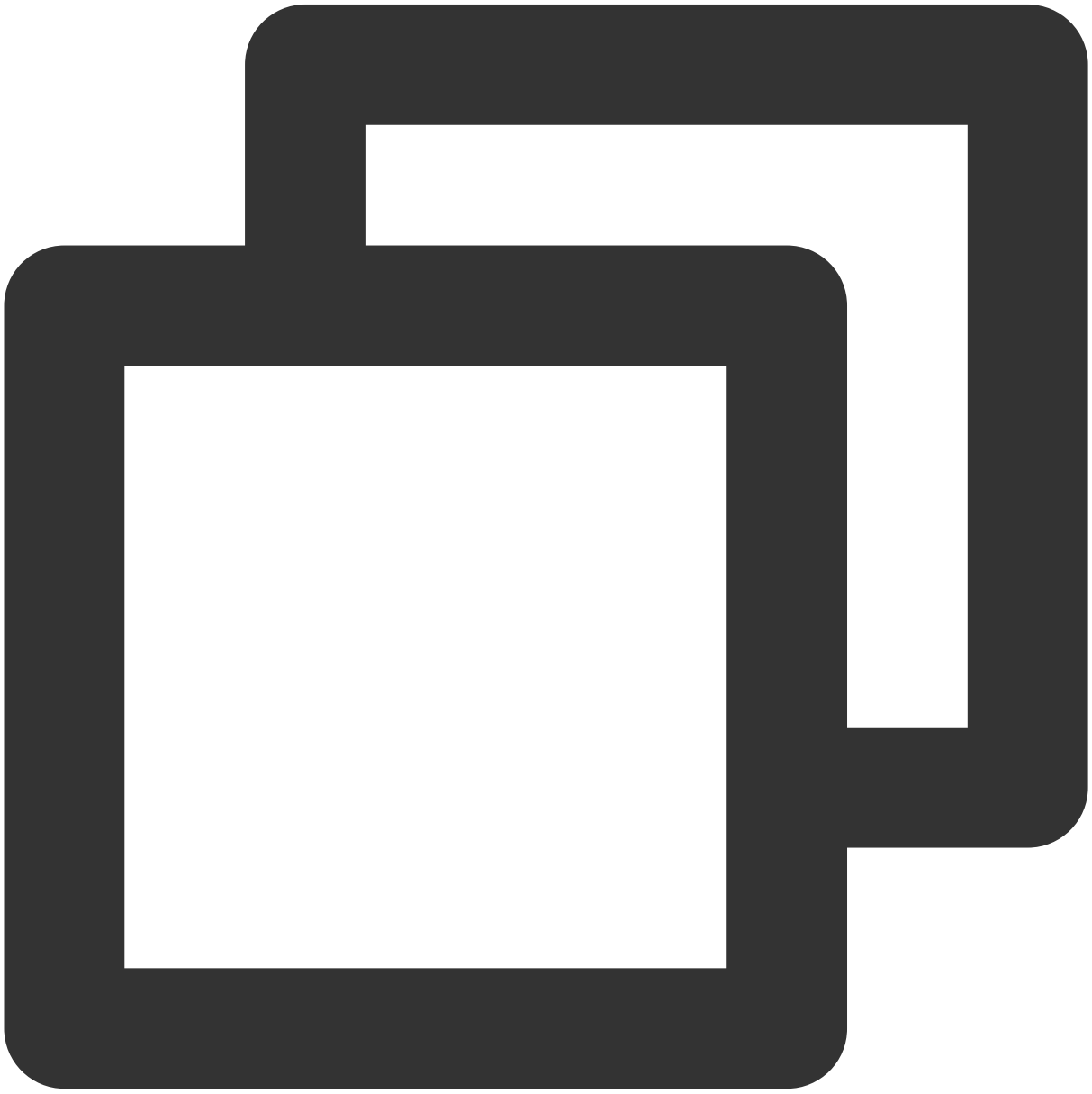
パラメータ	タイプ	意味
command	String	コマンドワードです。開発者がカスタマイズするもので、主にさまざまなメッセージタイプを区別するために使用されます。

message	String	テキストメッセージ。
userInfo	UserInfo	送信者のユーザー情報。

招待シグナリングイベントのコールバック

onReceiveNewInvitation

新規招待リクエストの受信。



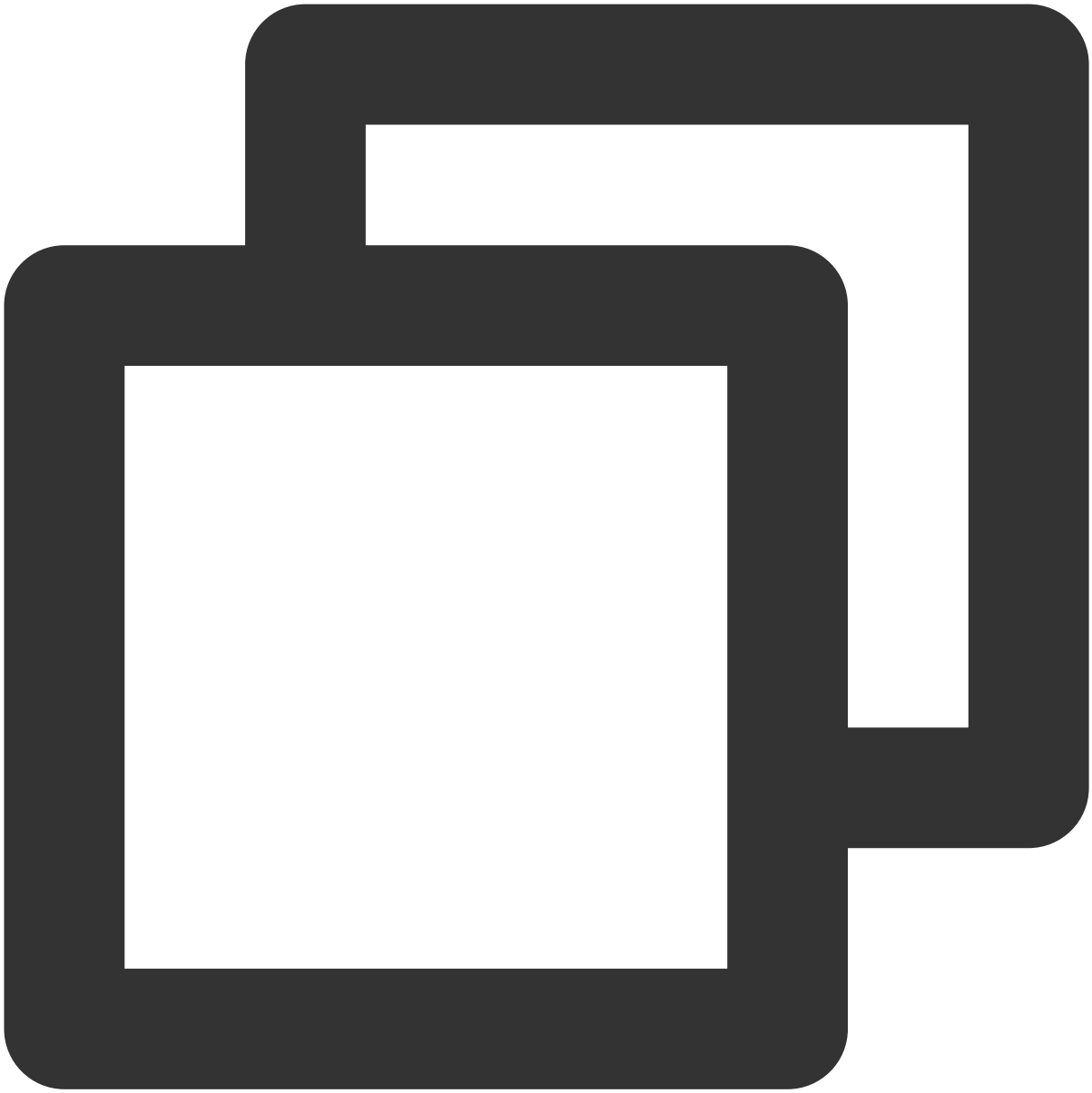
```
void onReceiveNewInvitation(String id, String inviter, String cmd, String content);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
inviter	String	招待者のユーザーID。
cmd	String	開発者がカスタマイズする業務指定のコマンドワードです。
content	String	業務指定のコンテンツ。

onInviteeAccepted

被招待者が招待に同意。



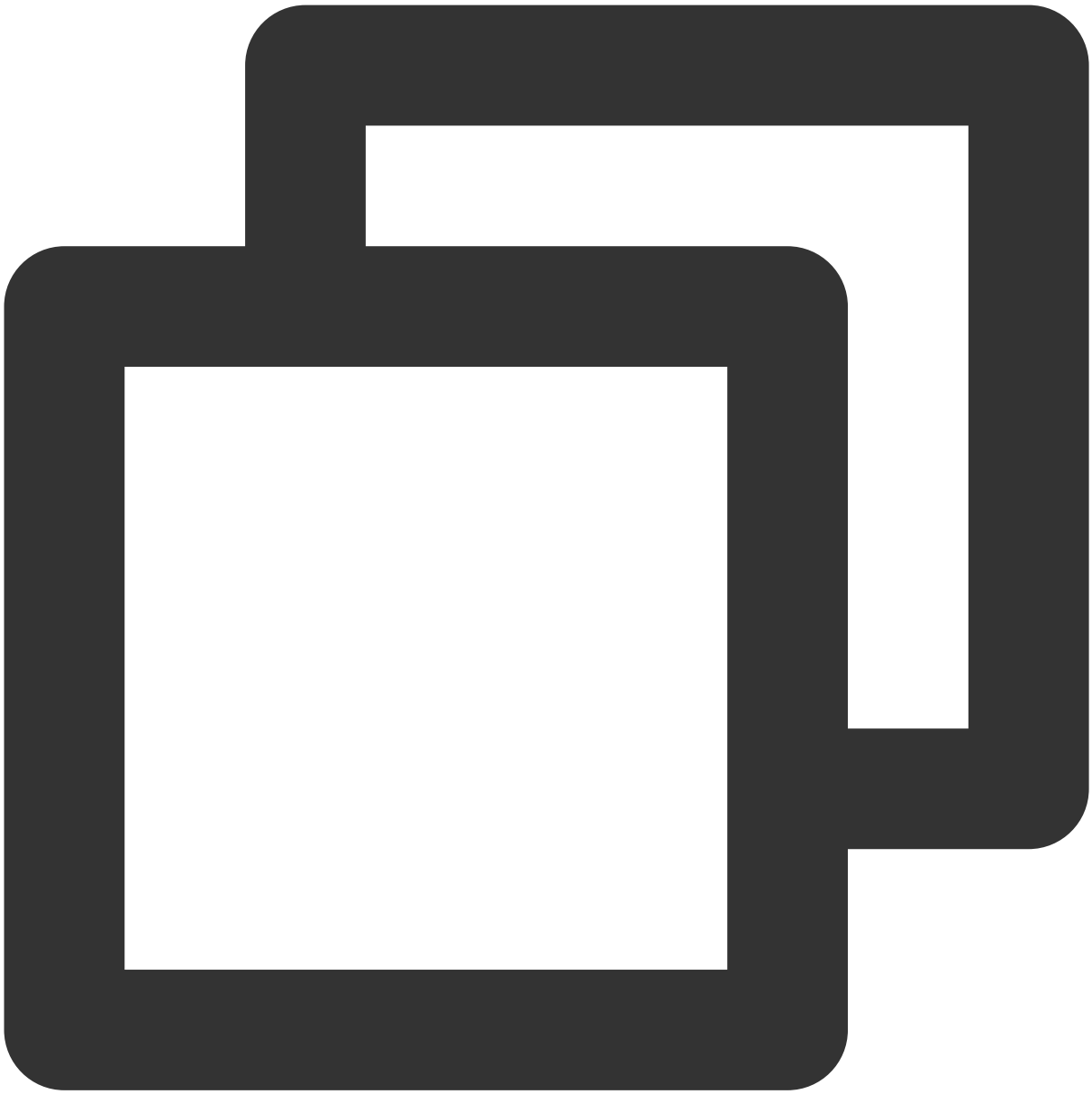
```
void onInviteeAccepted(String id, String invitee);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInviteeRejected

被招待者による招待の拒否。



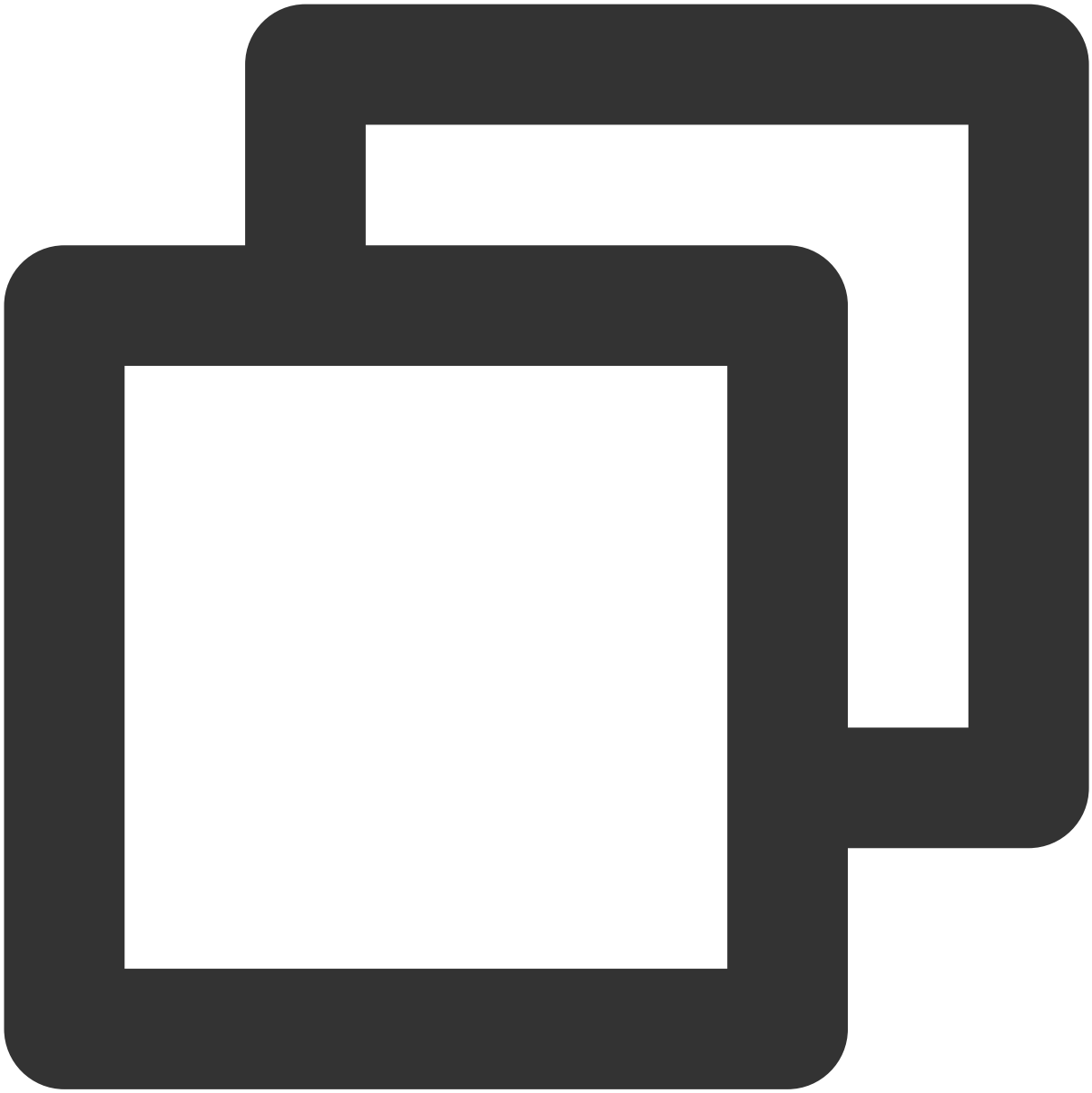
```
void onInviteeRejected(String id, String invitee);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
invitee	String	被招待者のユーザーID。

onInvitationCancelled

招待者が招待を取り消し。



```
void onInvitationCancelled(String id, String inviter);
```

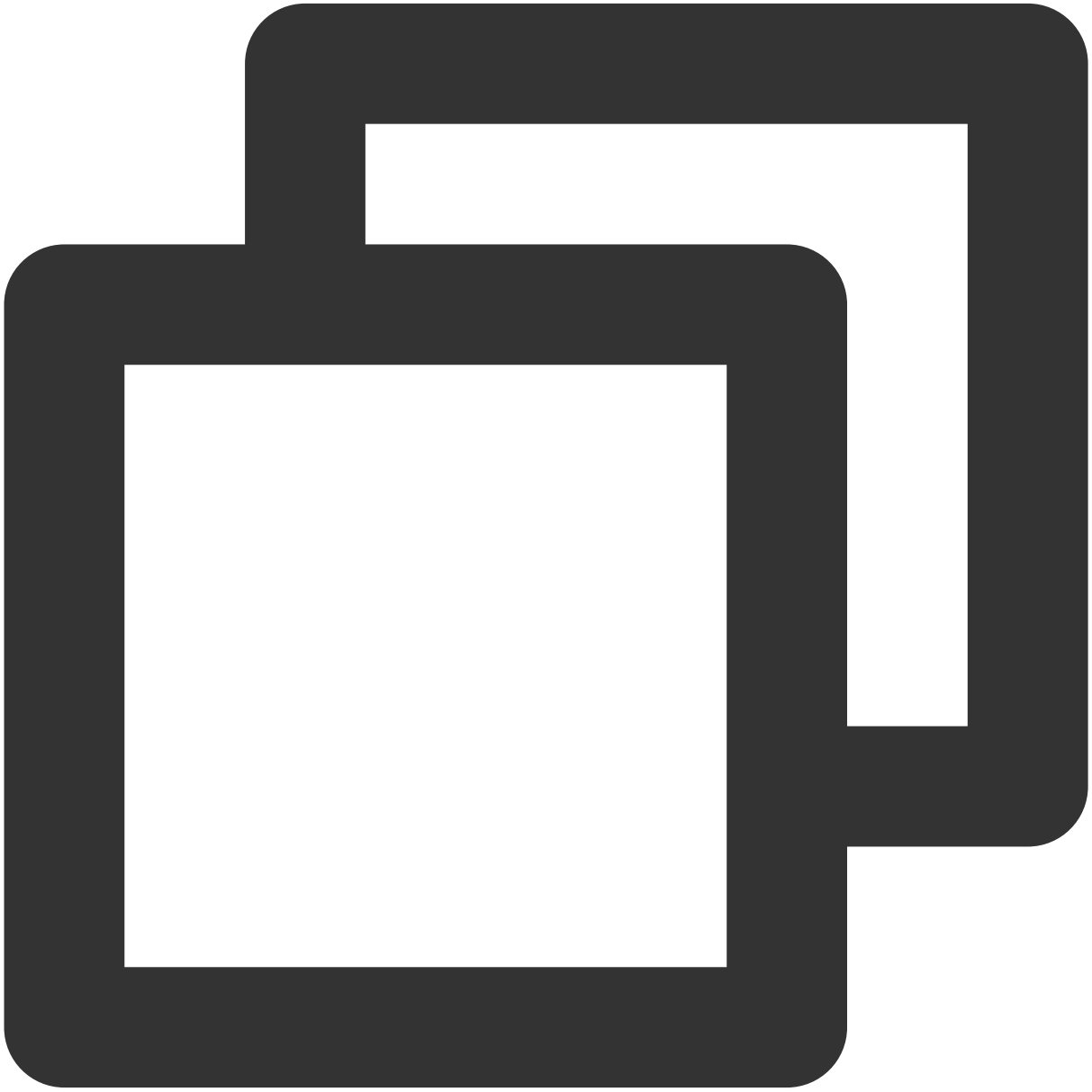
パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
id	String	招待ID。
inviter	String	招待者のユーザーID。

音楽再生ステータスコールバック

onMusicPrepareToPlay

音楽再生準備のコールバック



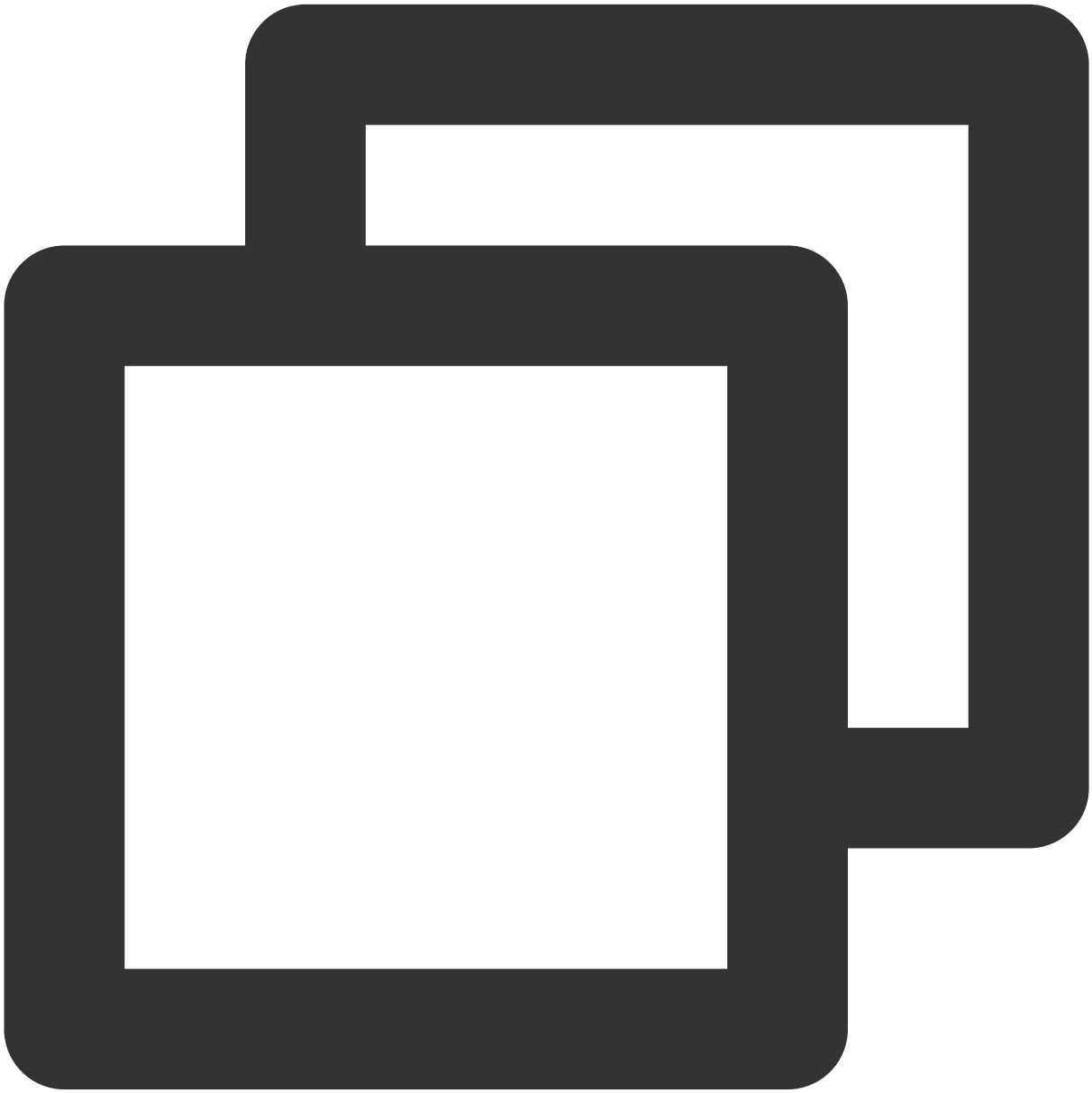
```
void onMusicPrepareToPlay(int musicID);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int	再生時に渡されたmusicID。

onMusicProgressUpdate

楽曲再生進捗度のコールバック



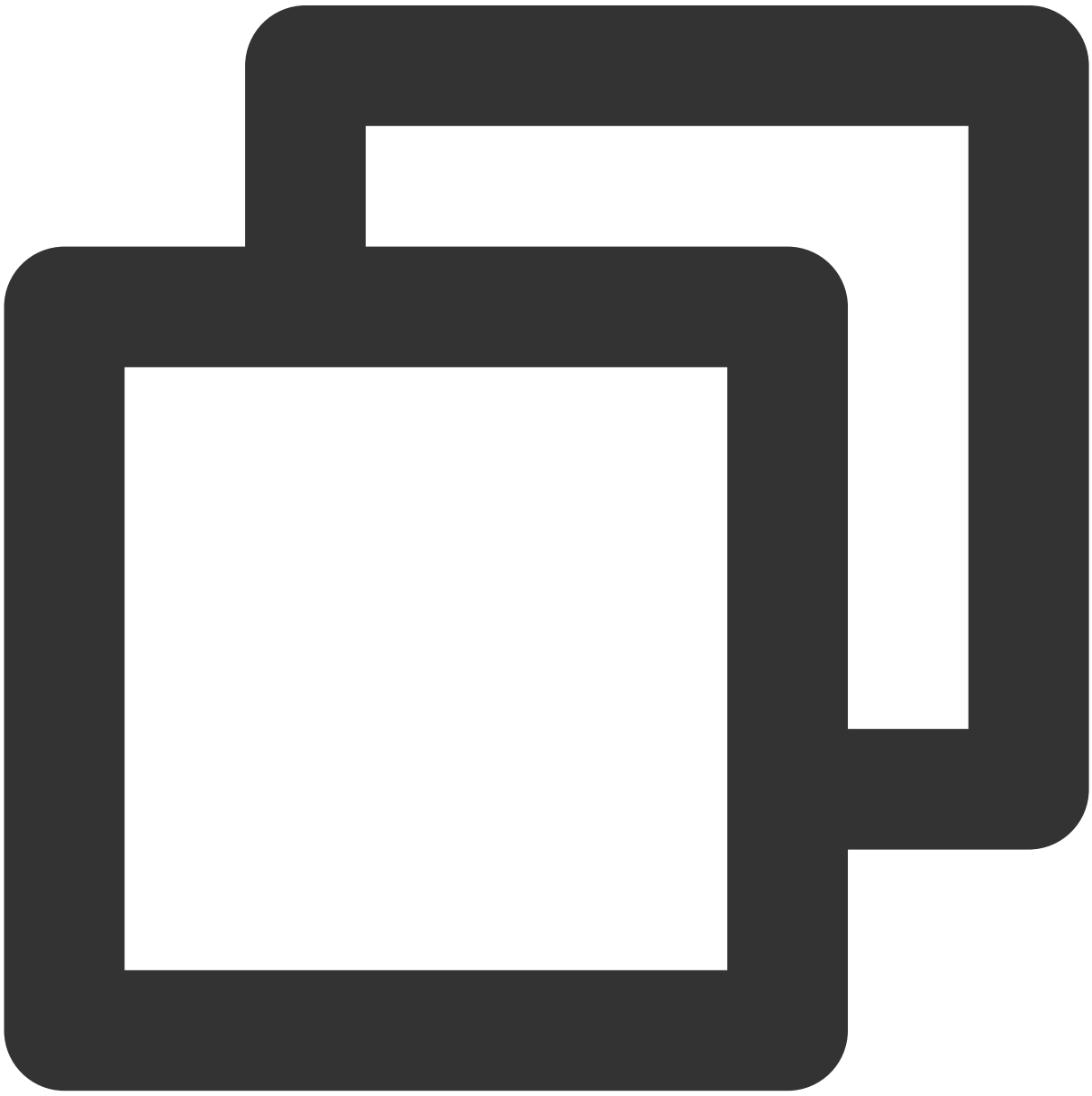
```
void onMusicProgressUpdate(int musicID, long progress, long total);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int	再生時に渡されたmusicID。
progress	long	現在の再生時間。単位： ms。
total	long	合計時間。単位： ms。

onMusicCompletePlaying

音楽再生完了のコールバック



```
void onMusicCompletePlaying(int musicID);
```

パラメータは下表に示すとおりです：

パラメータ	タイプ	意味
musicID	int	再生時に渡されたmusicID。