Game Multimedia Engine

クイックスタート

製品ドキュメント

カタログ：

クイックスタート
Quick Integration of SDK
Native SDKのクイック導入

最終更新日：：2024-01-18 15:42:47

開発者がGME製品APIのデバッグ・アクセスを行いやすいように、ここで、Unity開発に適用されるクイックアクセス技術ドキュメントを説明します。
GMEクイックスタート文書は、ユーザーのアクセスを助けるための最も主要なアクセスインターフェースのみを提供しています。

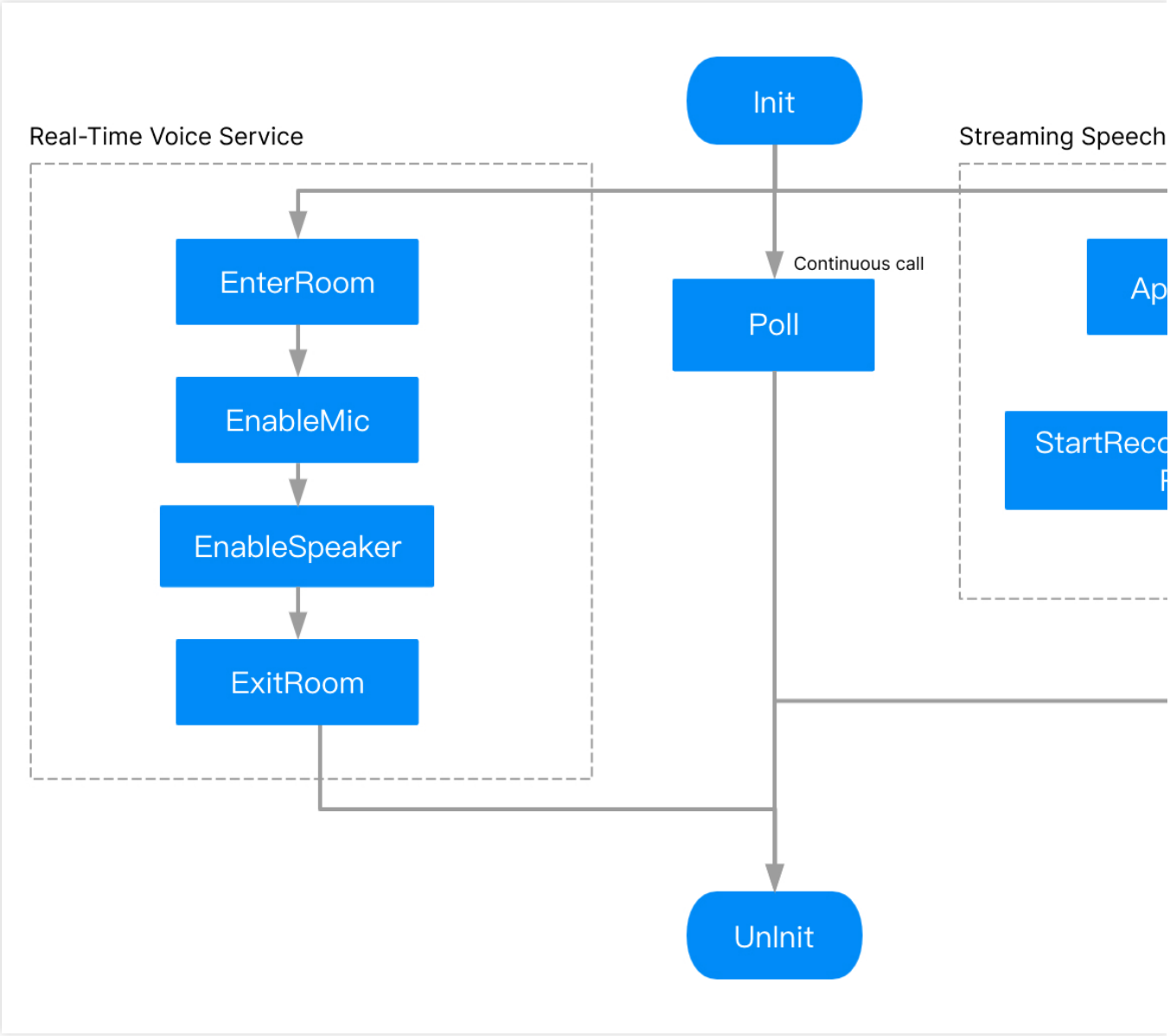
GME利用上の重要事項

GMEは2つの部分に分かれます。リアルタイム音声サービス、音声メッセージおよびテキスト変換サービスを提供しており、これらのサービスの利用はInitやPollなどのコアインターフェースに依存しています。

关于 Init 接口

例えば、リアルタイムの音声サービスを使用する同時に音声メッセージ・サービスも使用する場合、**Init初期化インターフェースを1回だけ呼び出す必要があります**。

インターフェース呼び出しのフローチャート



統合の手順

コアインターフェース

GMEの初期化接口：Init

定期的なPoll呼び出しによるコールバックのトリガー接口：Poll

リアルタイム音声

1. リアルタイム音声ルームへの参加，接口：EnterRoom
2. マイクのオン/オフ，接口：EnableMic
3. スピーカーのオン/オフ，接口：EnableSpeaker
4. 音声ルーム退出，接口：ExitRoom

音声メッセージ

1. 認証の初期化，接口：ApplyPTTAuthbuffer
2. ストリーミング音声認識の起動，接口：StartRecordingWithStreamingRecognition
3. レコーディング停止，接口：StopRecording

GMEの未初期化，接口：UnInit

コアインターフェースのアクセス

1. SDKのダウンロード

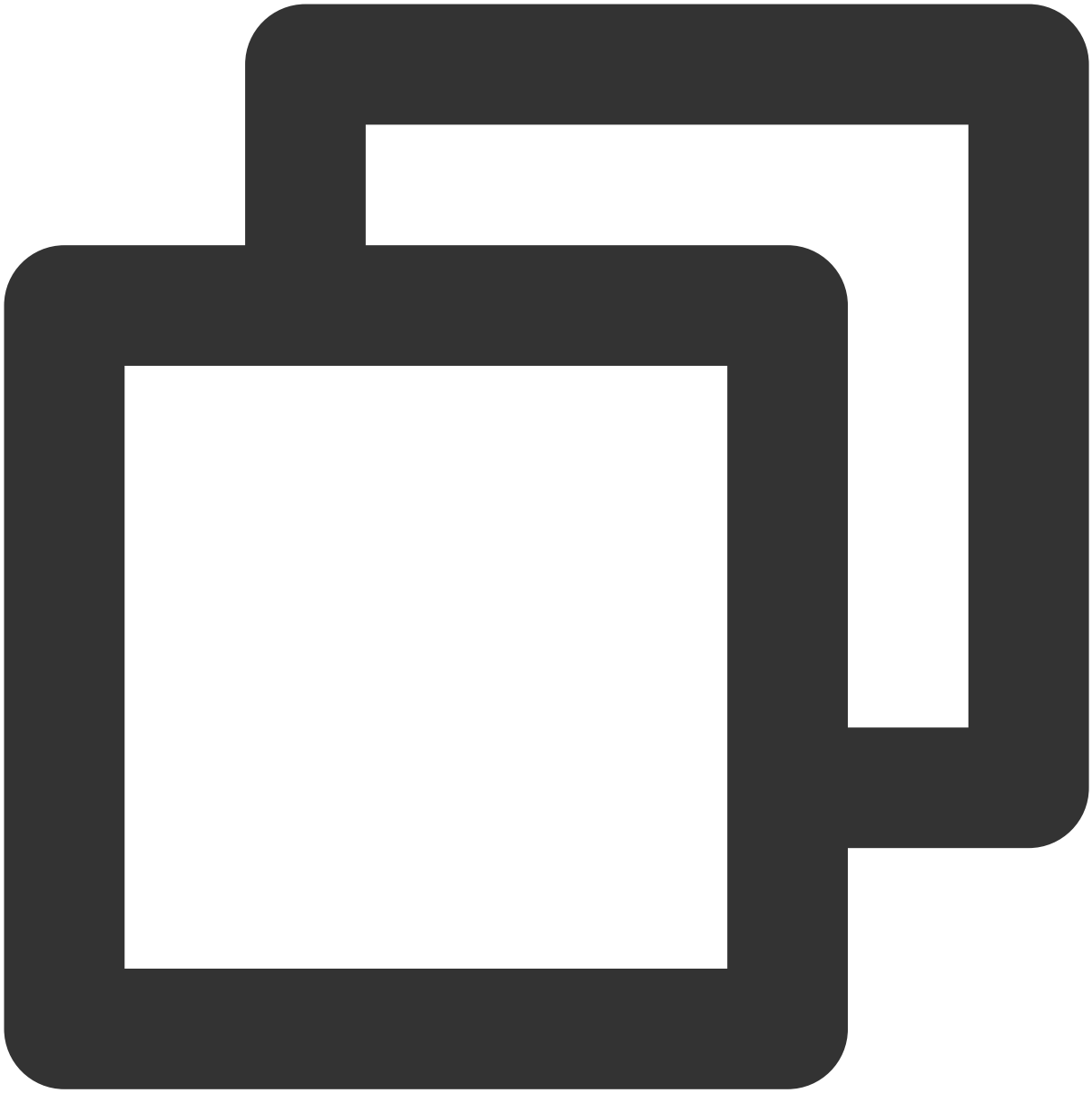
ダウンロード案内ページにアクセスして、必要な クライアンSDKをダウンロードします。

2. ヘッダーファイルを取り込む

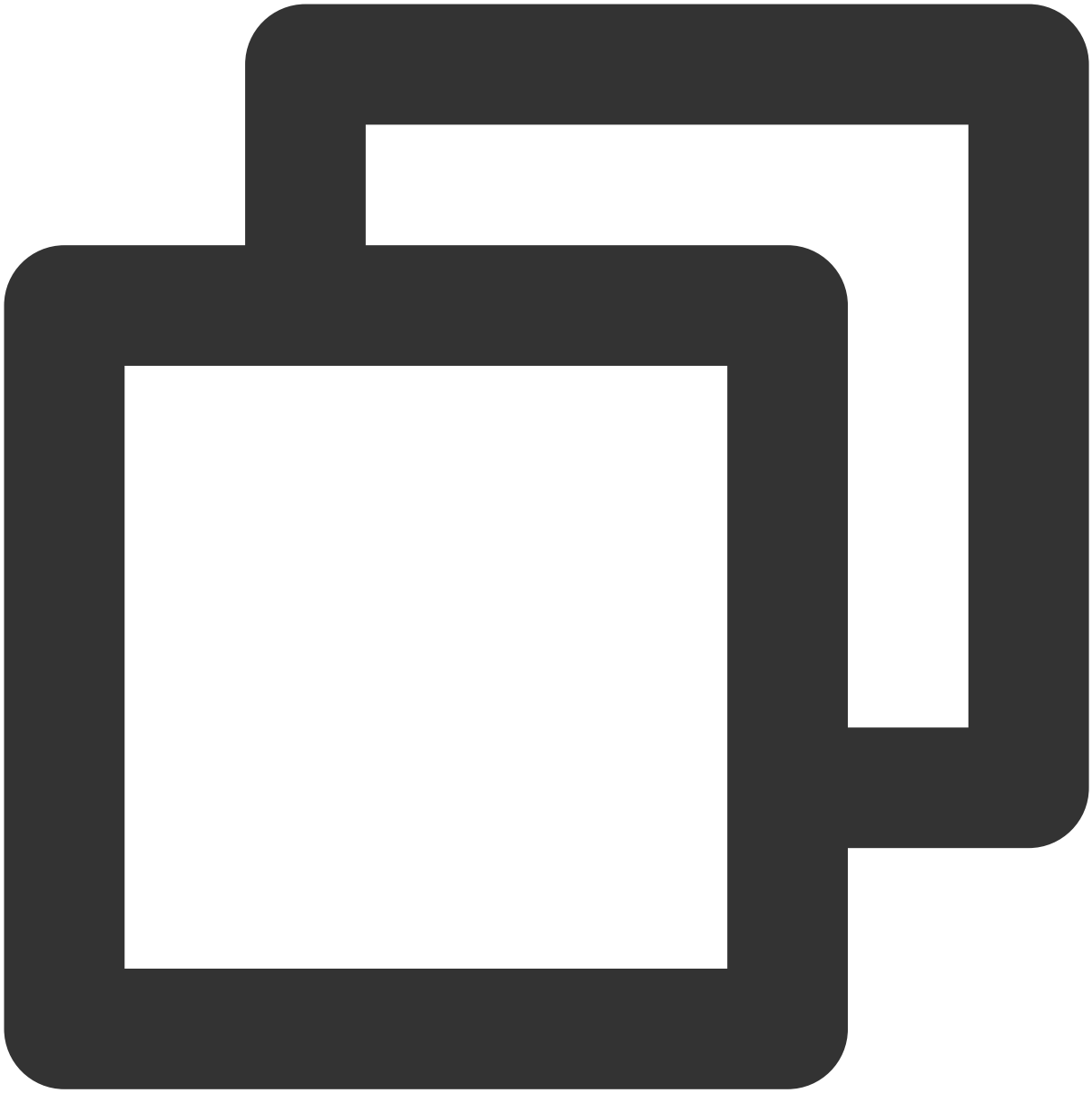
Java

Object-C

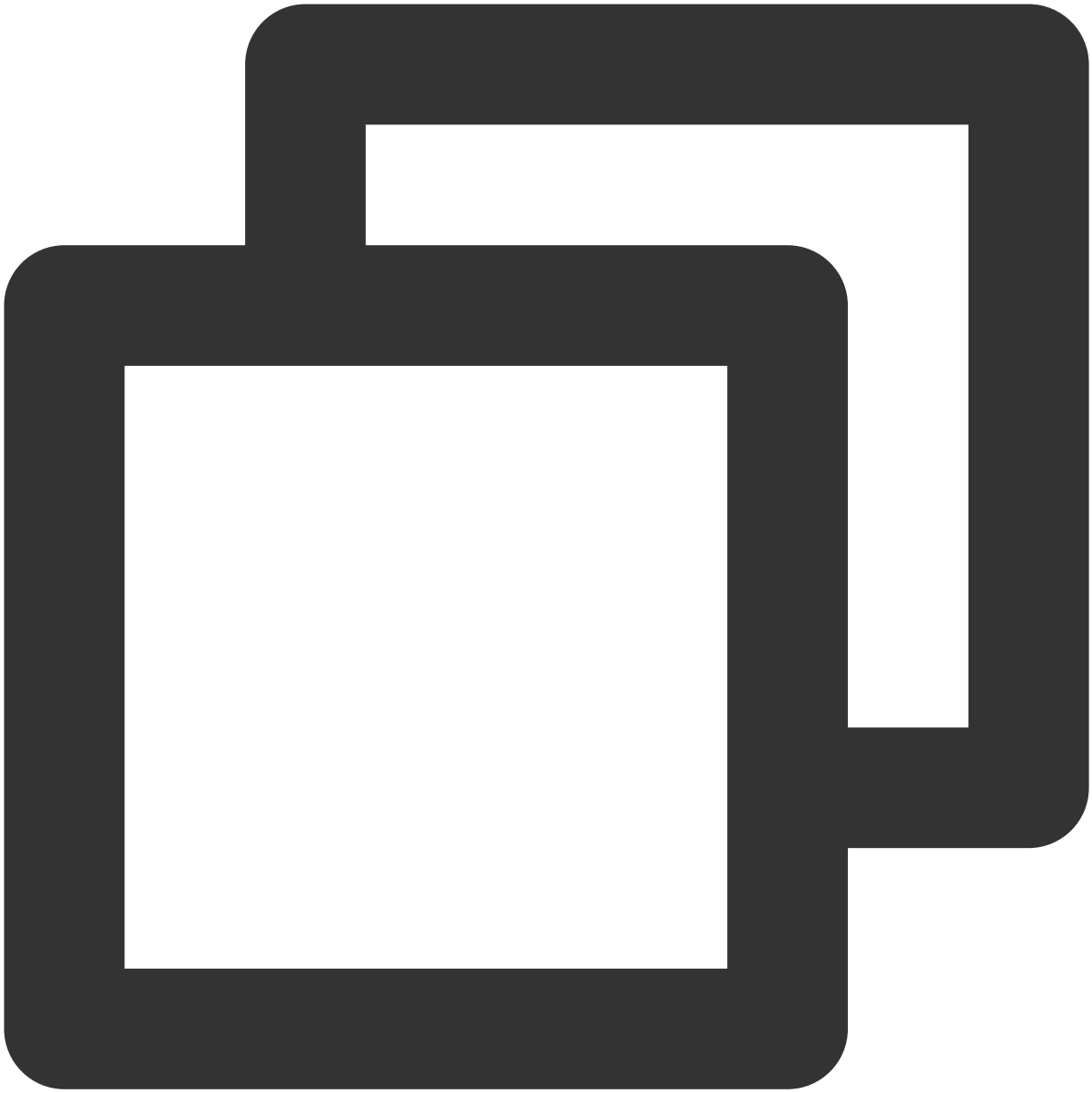
C++



```
import com.tencent.TMG.ITMGContext;
import com.tencent.av.sig.AuthBuffer;
import com.tencent.bugly.crashreport.CrashReport;
```



```
#import "GMESDK/TMGEngine.h"
#import "GMESDK/QAVAuthBuffer.h"
```



```
#include "auth_buffer.h"
#include "tmg_sdk.h"
#include "AdvanceHeaders/tmg_sdk_adv.h"
#include <vector>
```

3. シングルトンの取得

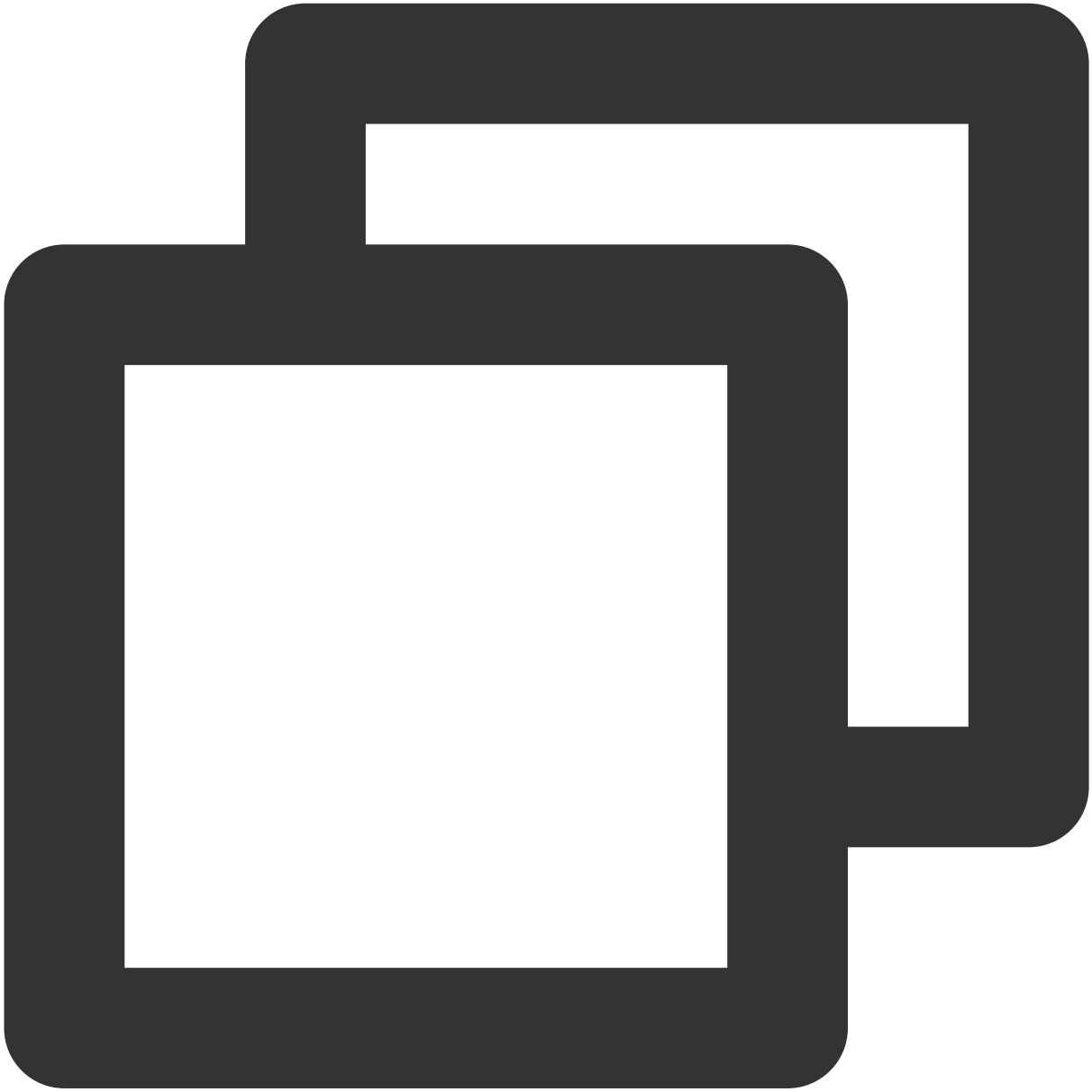
音声機能を使用する場合、先にITMGContextオブジェクトを取得する必要があります。

関数のプロトタイプ

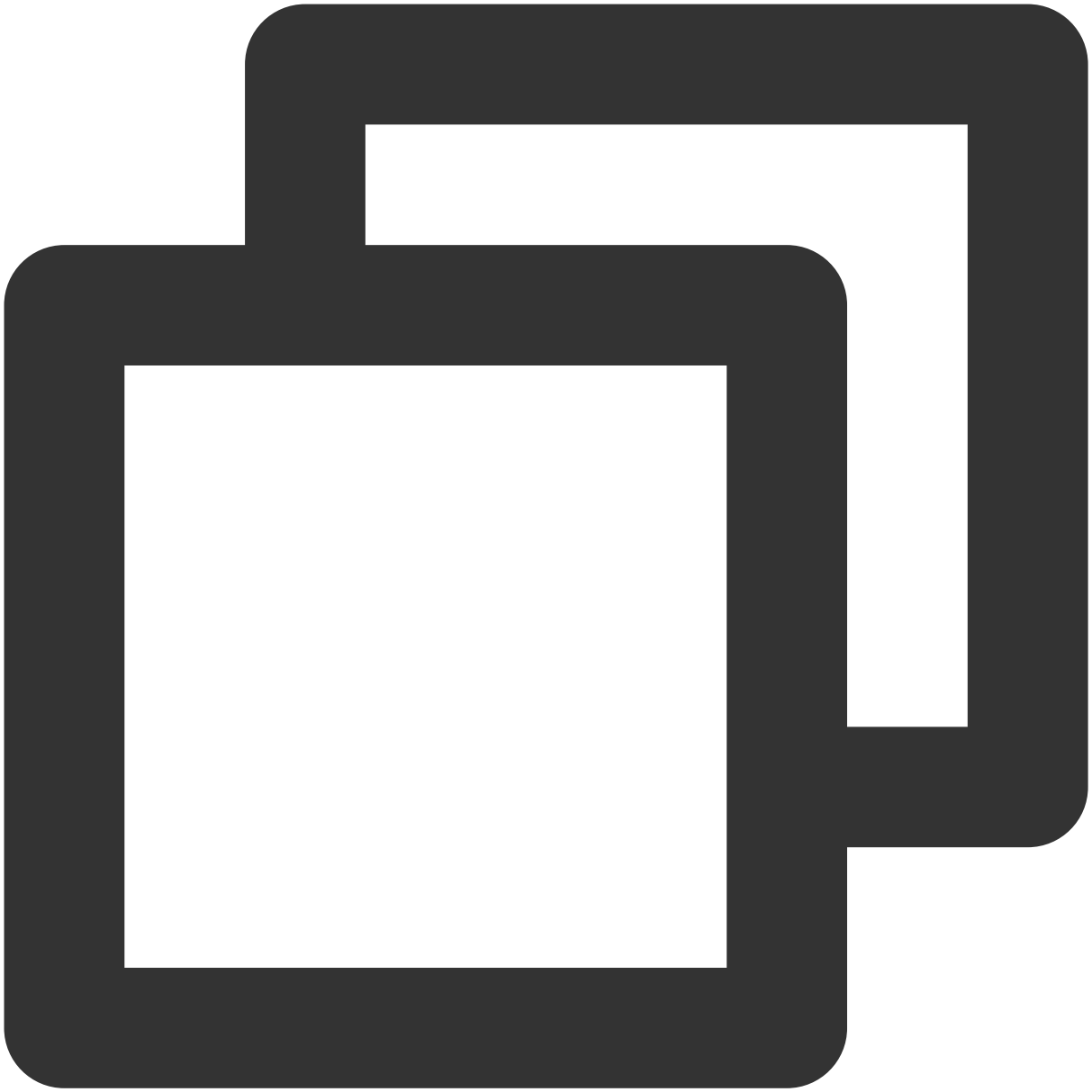
Java

Object-C

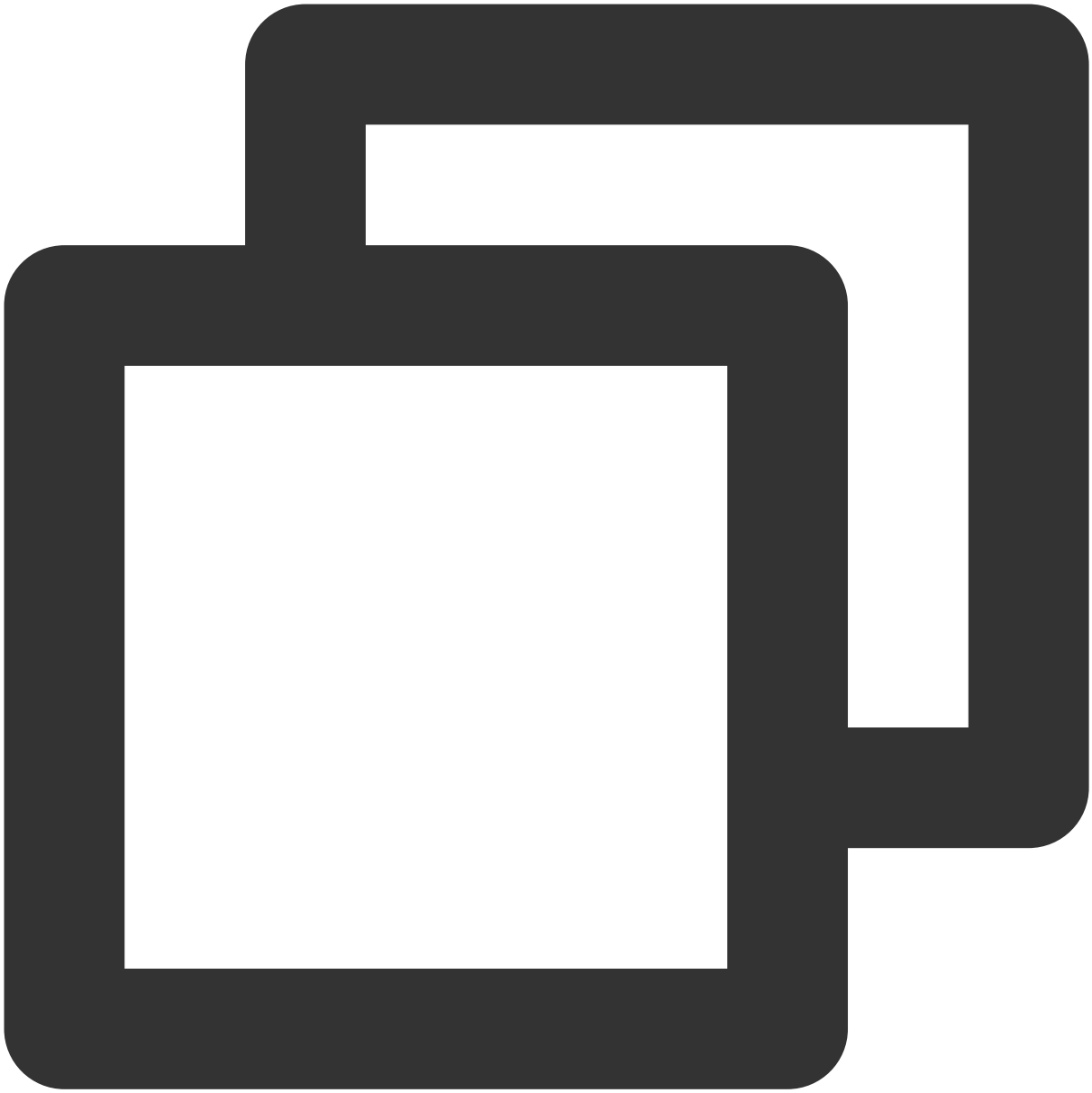
C++



```
public static ITMGContext GetInstance(Context context)
```



```
+ (ITMGContext*) GetInstance;
```



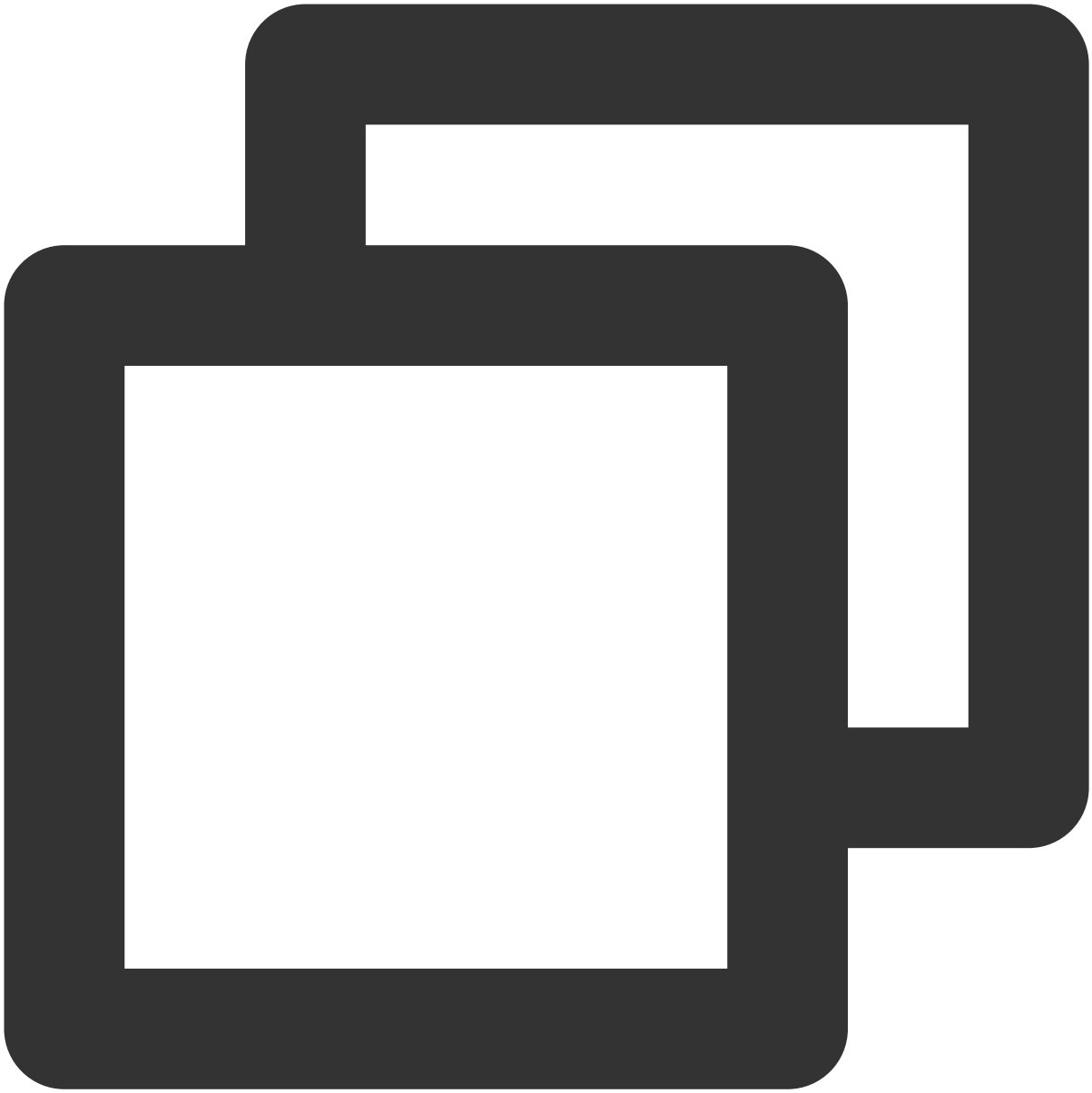
```
__UNUSED static ITMGContext* ITMGContextGetInstance(){
return ITMGContextGetInstanceInner(TMG_SDK_VERSION);
}
```

サンプルコード

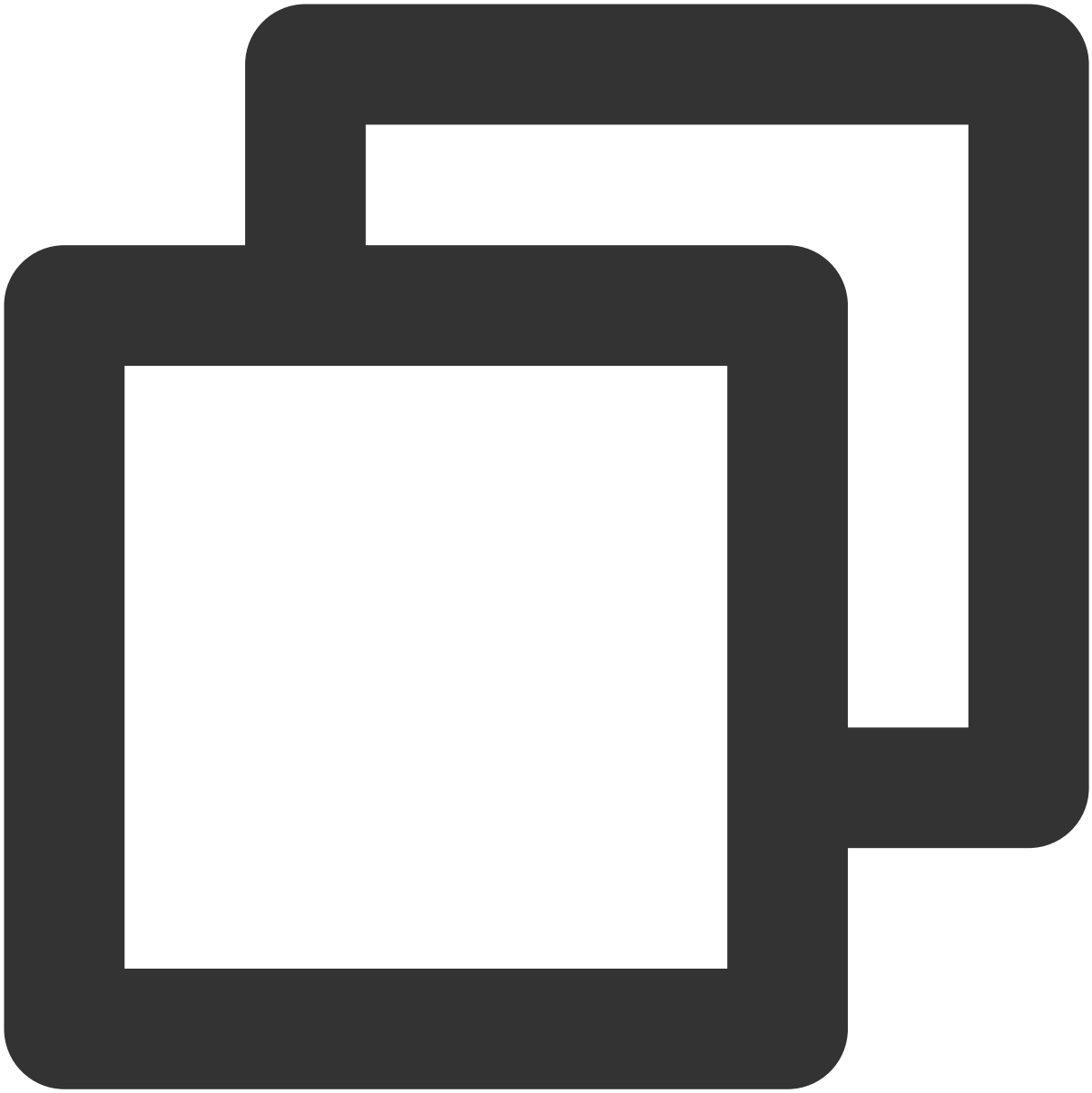
Java

Object-C

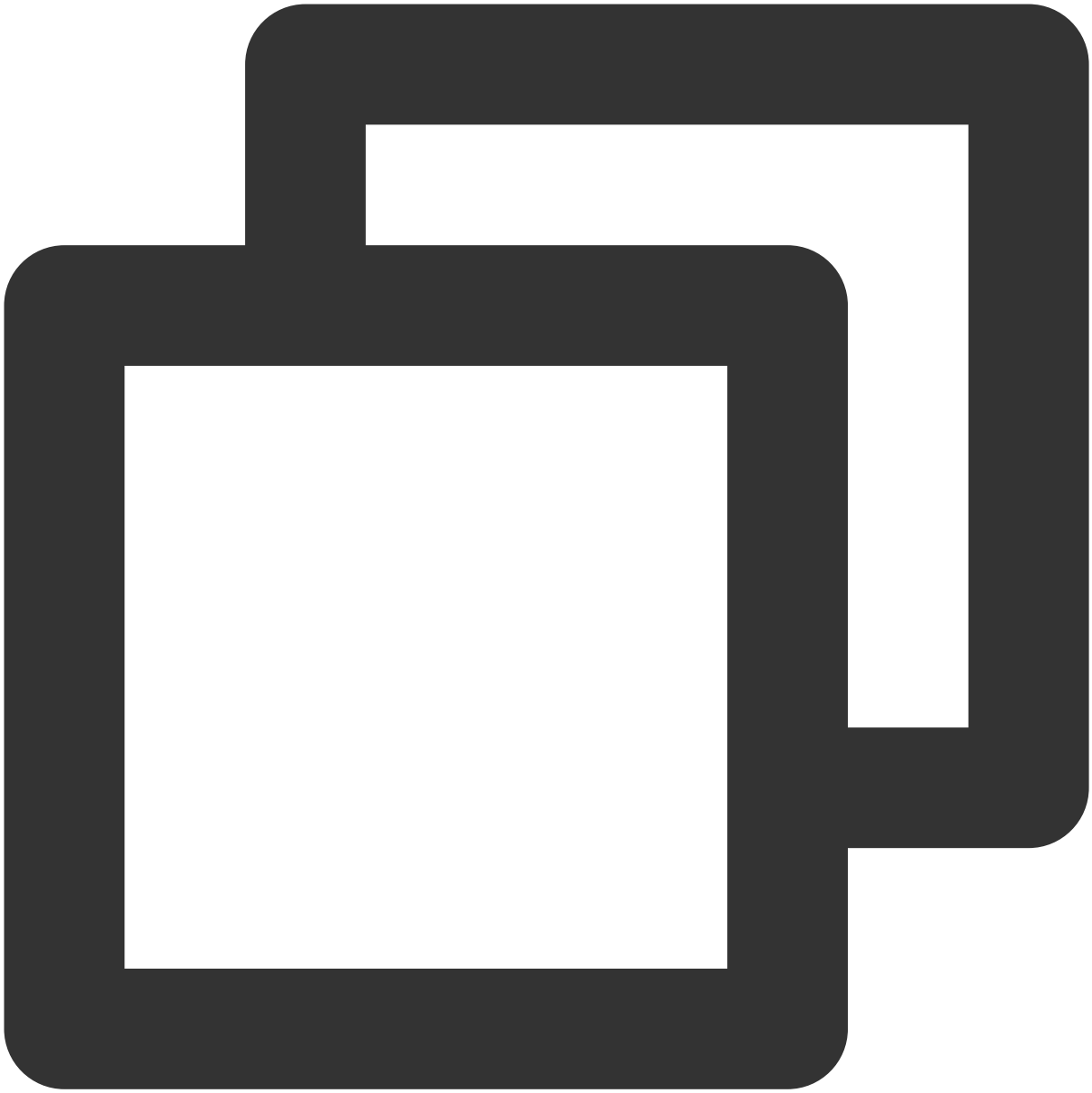
C++



```
//MainActivity.java
import com.tencent.TMG.ITMGContext;
ITMGContext tmgContext = ITMGContext.GetInstance(this);
```



```
//TMGSampleViewController.m
ITMGContext* _context = [ITMGContext GetInstance];
```



```
ITMGContext* context = ITMGContextGetInstance();
```

4. コールバックの設定

インターフェースクラスは、Delegateメソッドを使用して、アプリケーションにコールバック通知を送信します。コールバック関数をSDKに登録して、コールバック情報を受信します。ルームに参加する前に設定してください。

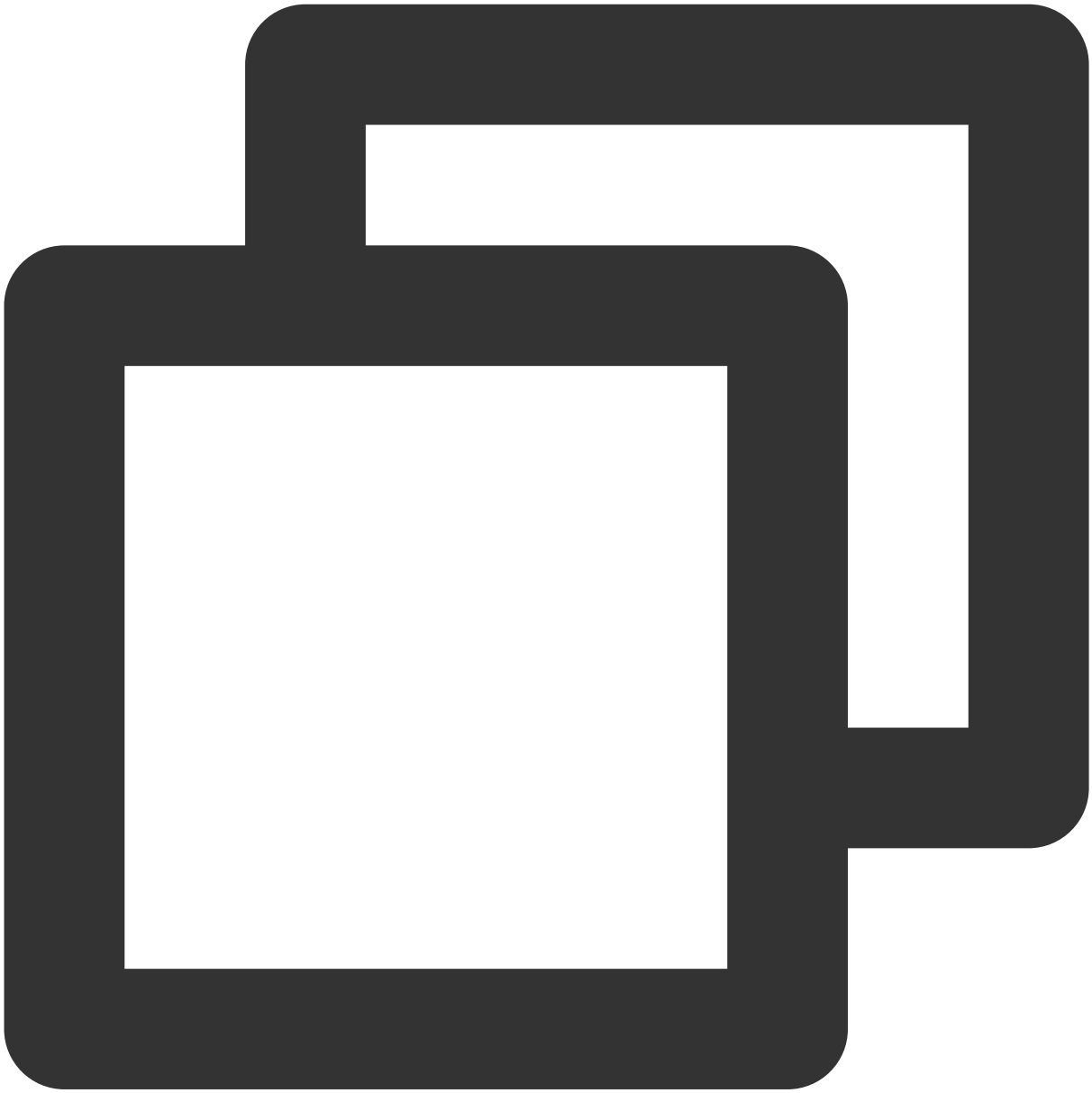
関数のプロトタイプとサンプルコード

コールバックの情報を受信するためのコールバックを設定します。ルームに入る前に設定してください。

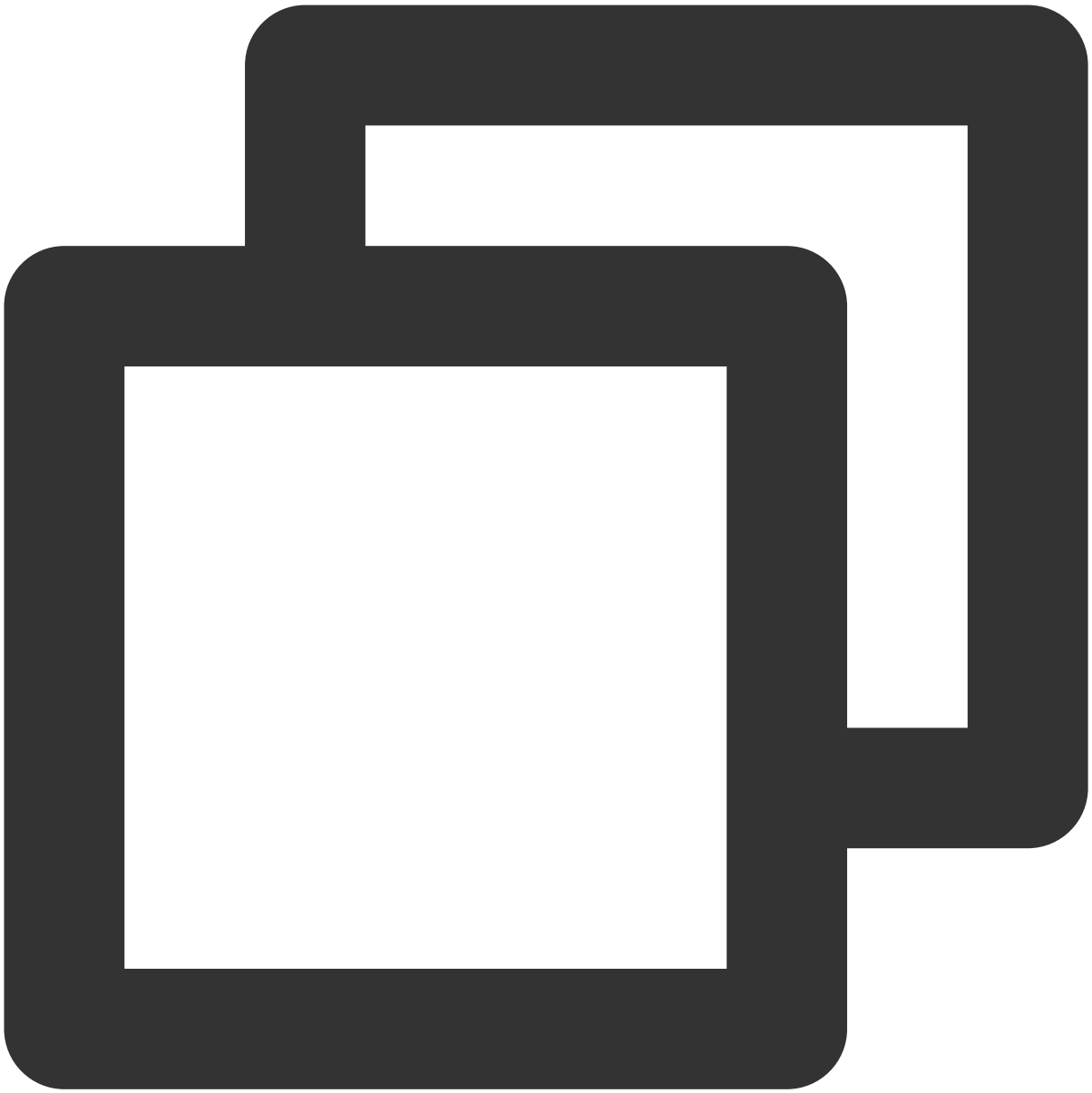
Java

Object-C

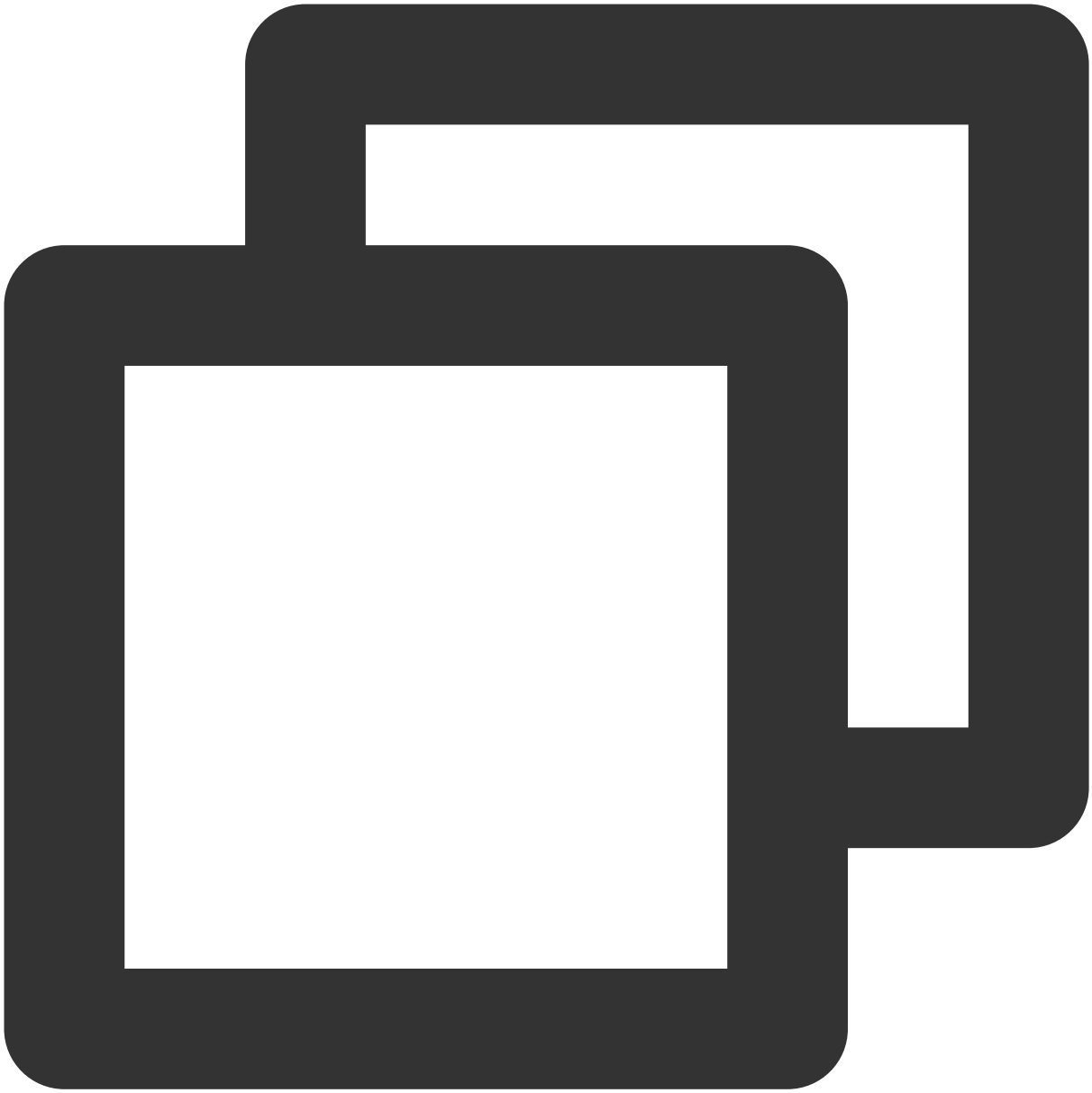
C++



```
//ITMGContext
public abstract int SetTMGDelegate(ITMGDelegate delegate);

//MainActivity.java
tmgContext.SetTMGDelegate(TMGCallbackDispatcher.getInstance());
```



```
ITMGDelegate < NSObject >

//TMGSampleViewController.m
ITMGContext* _context = [ITMGContext GetInstance];
_context.TMGDelegate = [DispatchCenter getInstance];
```



```
//SDKの初期化時
m_pTmgContext = ITMGContextGetInstance();
m_pTmgContext->SetTMGDelegate(this);
//デストラクタ内
CTMGSDK_For_AudioDlg::~CTMGSDK_For_AudioDlg()
{
    ITMGContextGetInstance()->SetTMGDelegate(NULL);
}
```

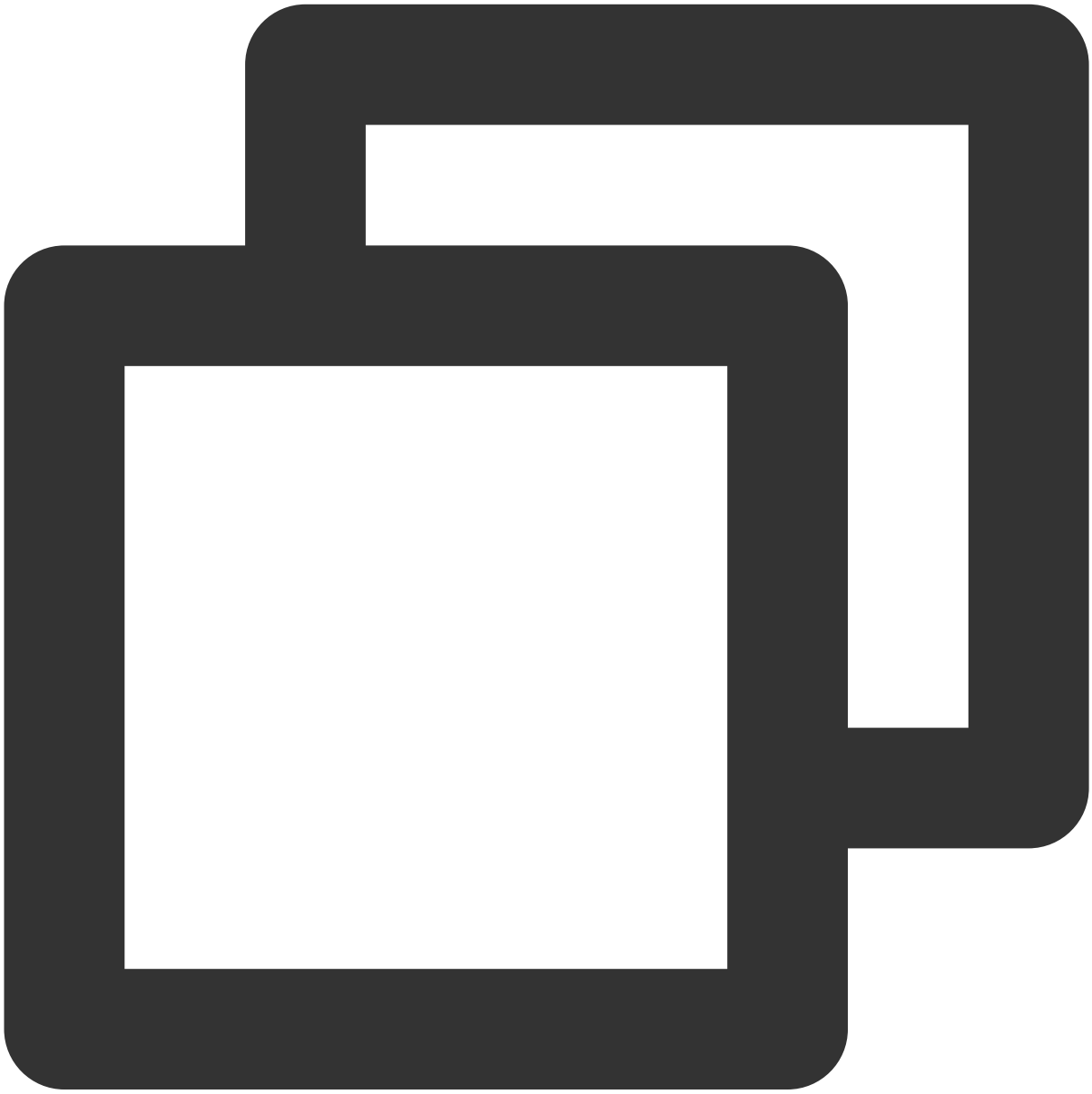
コールバックの例

コンストラクタでこのコールバック関数をオーバーライドして、コールバックパラメータを処理させます。

Java

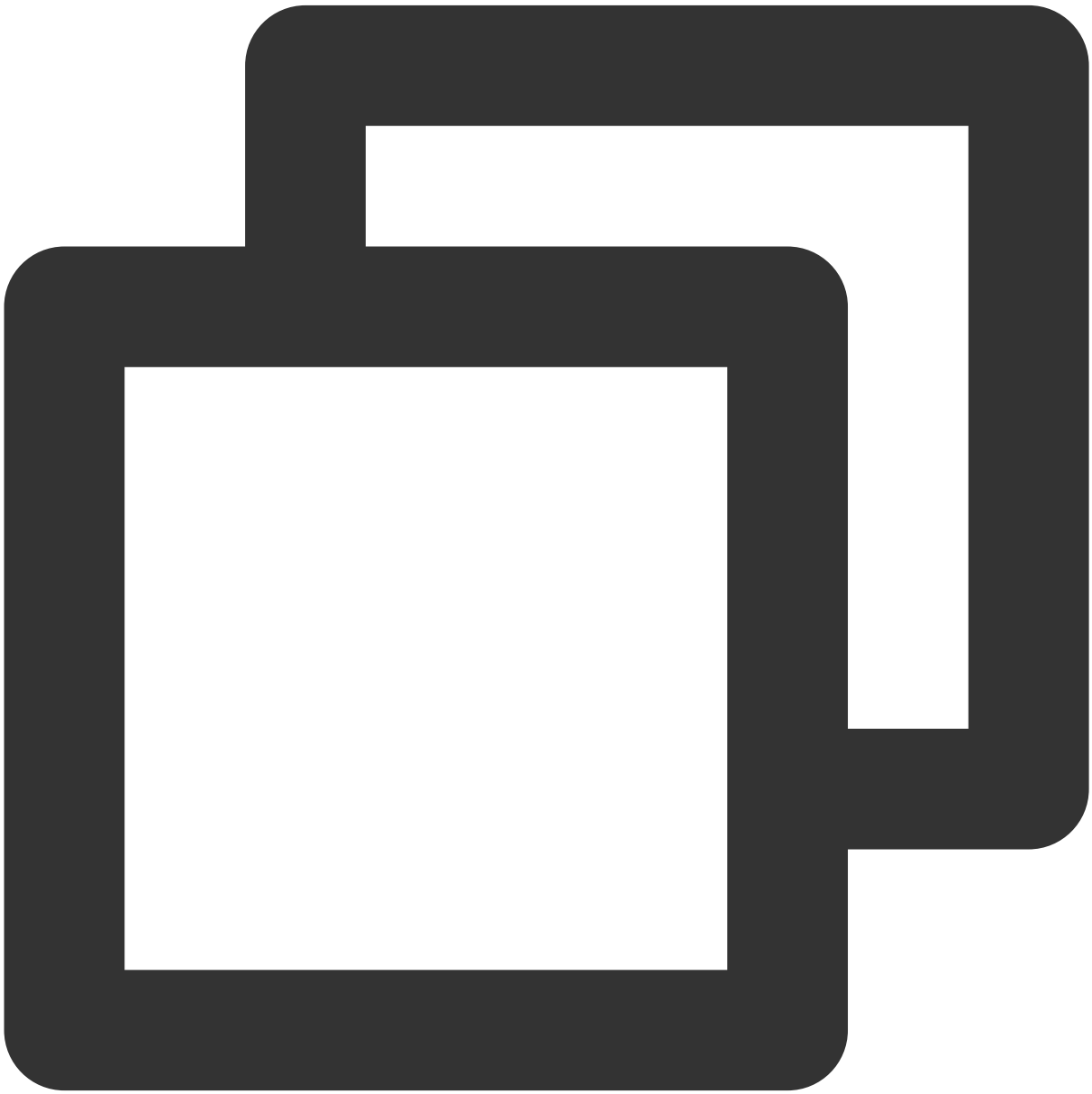
Object-C

C++



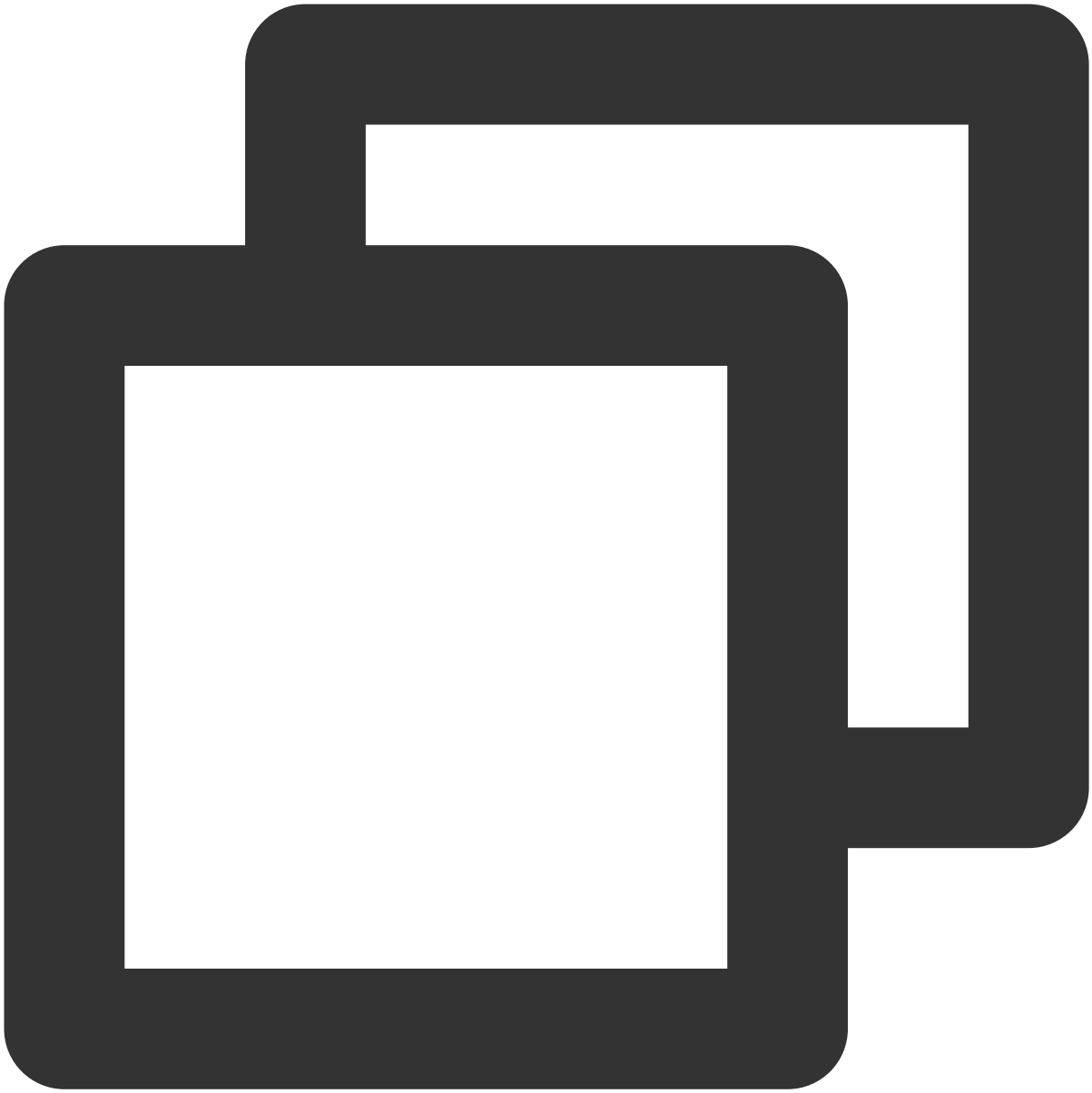
```
//MainActivity.java
tmgContext.SetTMGDelegate(TMGCallbackDispatcher.getInstance());

//RealTimeVoiceActivity.java
```

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
        if (type == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM)
        {
            //コールバック処理
        }
}

//TMGCallbackDispatcher.java、TMGCallbackHelper.javaおよびTMGDispatcherBase.javaを参考
```



```
//TMGRealTimeViewController.m
```

```
TMGRealTimeViewController ()< ITMGDelegate >


- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data {
  NSString *log = [NSString stringWithFormat:@"OnEvent:%d,data:%@", (int)eventType,
  [self showLog:log];
  NSLog(@"====%@====", log);
  switch (eventType) {
      // Step 6/11 : Perform the enter room event
      case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM: {
          int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
          NSString* error_info = [data objectForKey:@"error_info"];

          [self showLog:[NSString stringWithFormat:@"OnEnterRoomComplete:%d msg:(%@

          if (result == 0) {
              [self updateStatusEnterRoom:YES];
          }
      }
      break;

  }
  }

//DispatchCenter.h、DispatchCenter.mを参考にしてください
```



```
//ヘッダファイルにおける宣言
virtual void OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data);
//サンプルコード
void CTMGSDK_For_AudioDlg::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char* data
{
    switch(eventType)
    {
    case ITMG_MAIN_EVENT_TYPE_XXXX_XXXX:
        {
            //コールバックの処理
        }
```

```
            break;
        }
    }
```

パラメータ	タイプ	意味
type	ITMGContext.ITMG_MAIN_EVENT_TYPE	コールバックのイベントタイプ
data	Intent メッセージタイプ	コールバックの関連情報、イベントデータ

5. SDKを初期化する

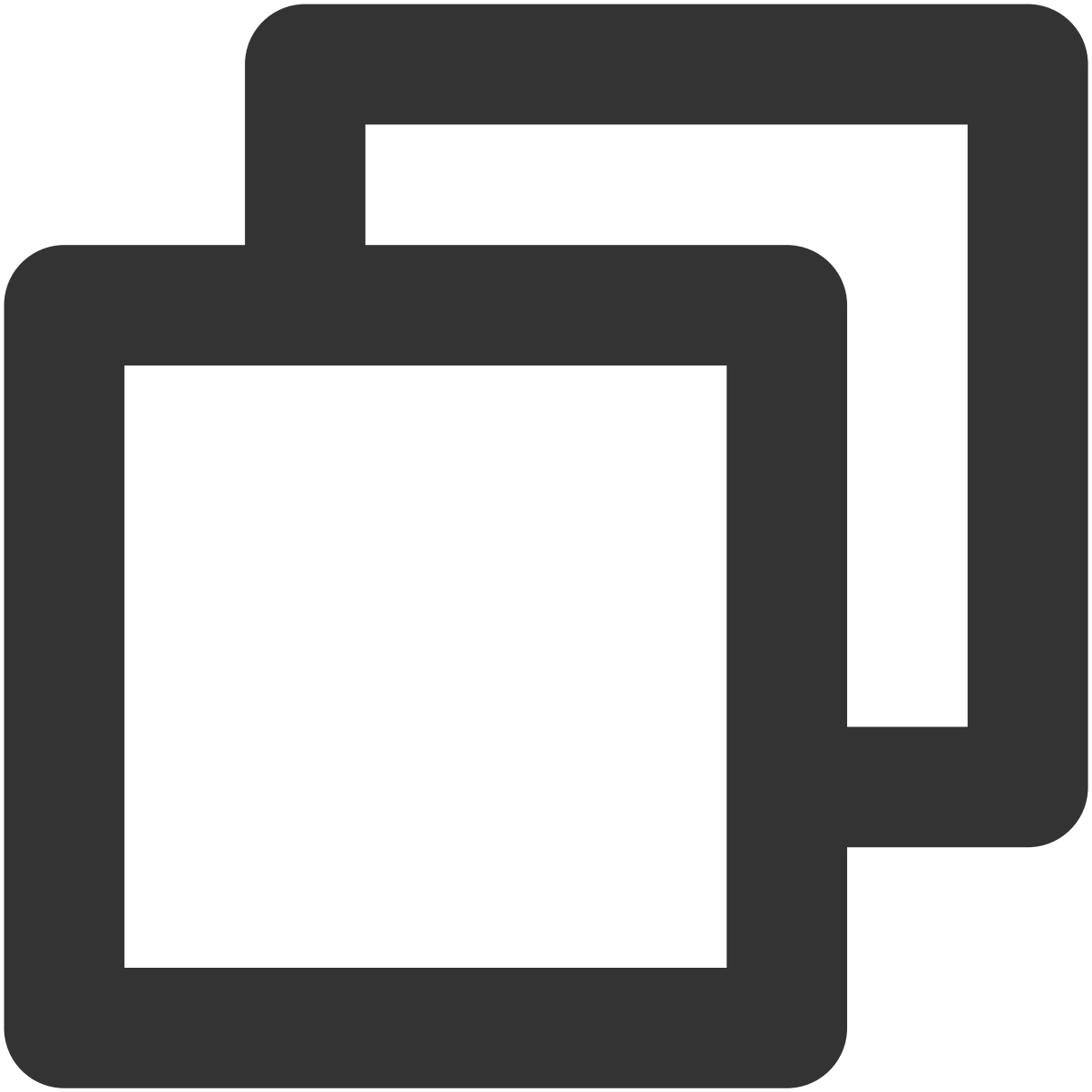
初期化前のSDKは初期化されていない状態です。リアルタイム音声サービス、音声メッセージサービスおよびボイスツーテキスト変換サービスを使用するには、**インターフェースInitを使用してSDKを初期化する必要があります**。Initインターフェースを呼び出すスレッドは、他のインターフェースと同じスレッドである必要があります。すべてのメインスレッドでインターフェースを呼び出すことをお勧めします。

インターフェースのプロトタイプ

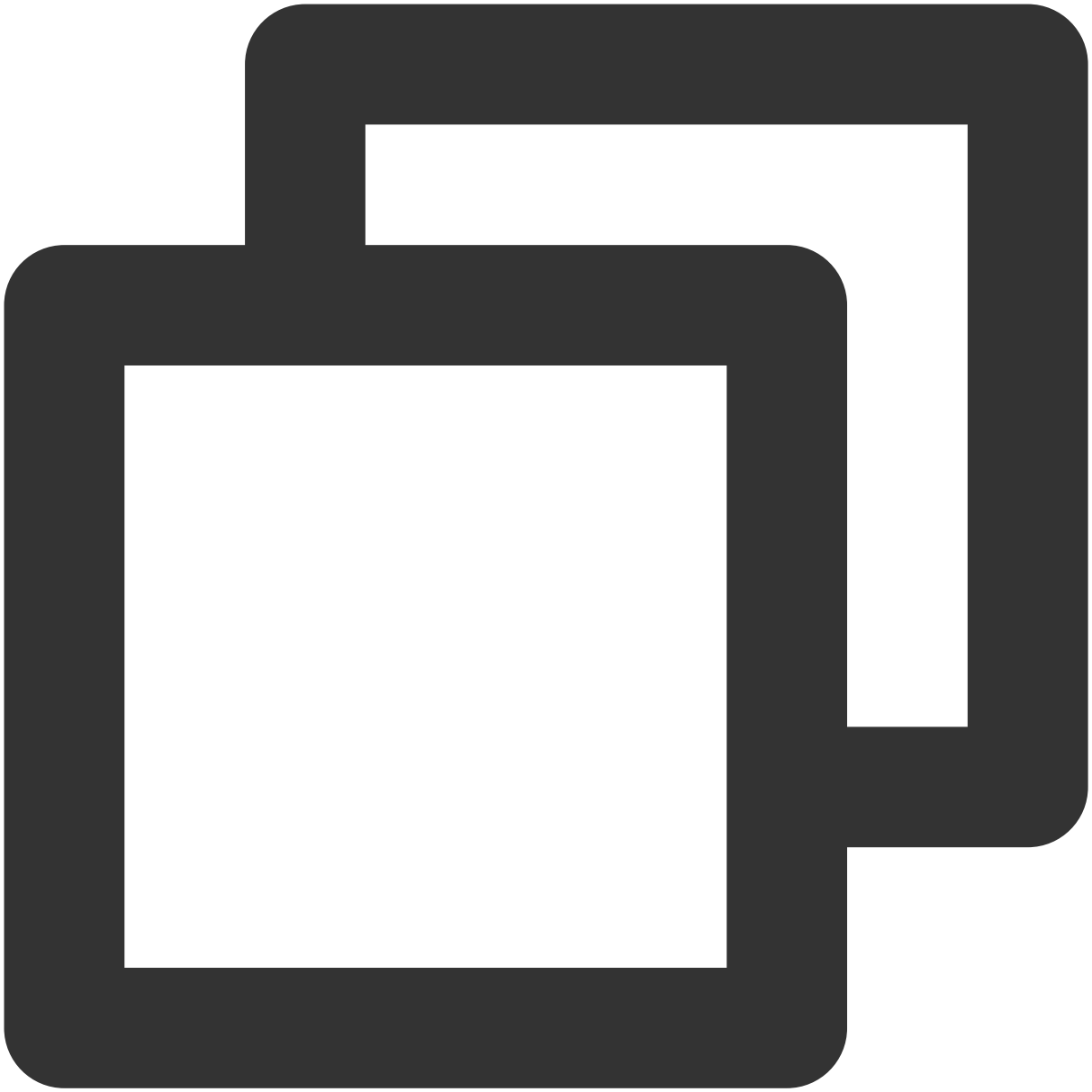
Java

Object-C

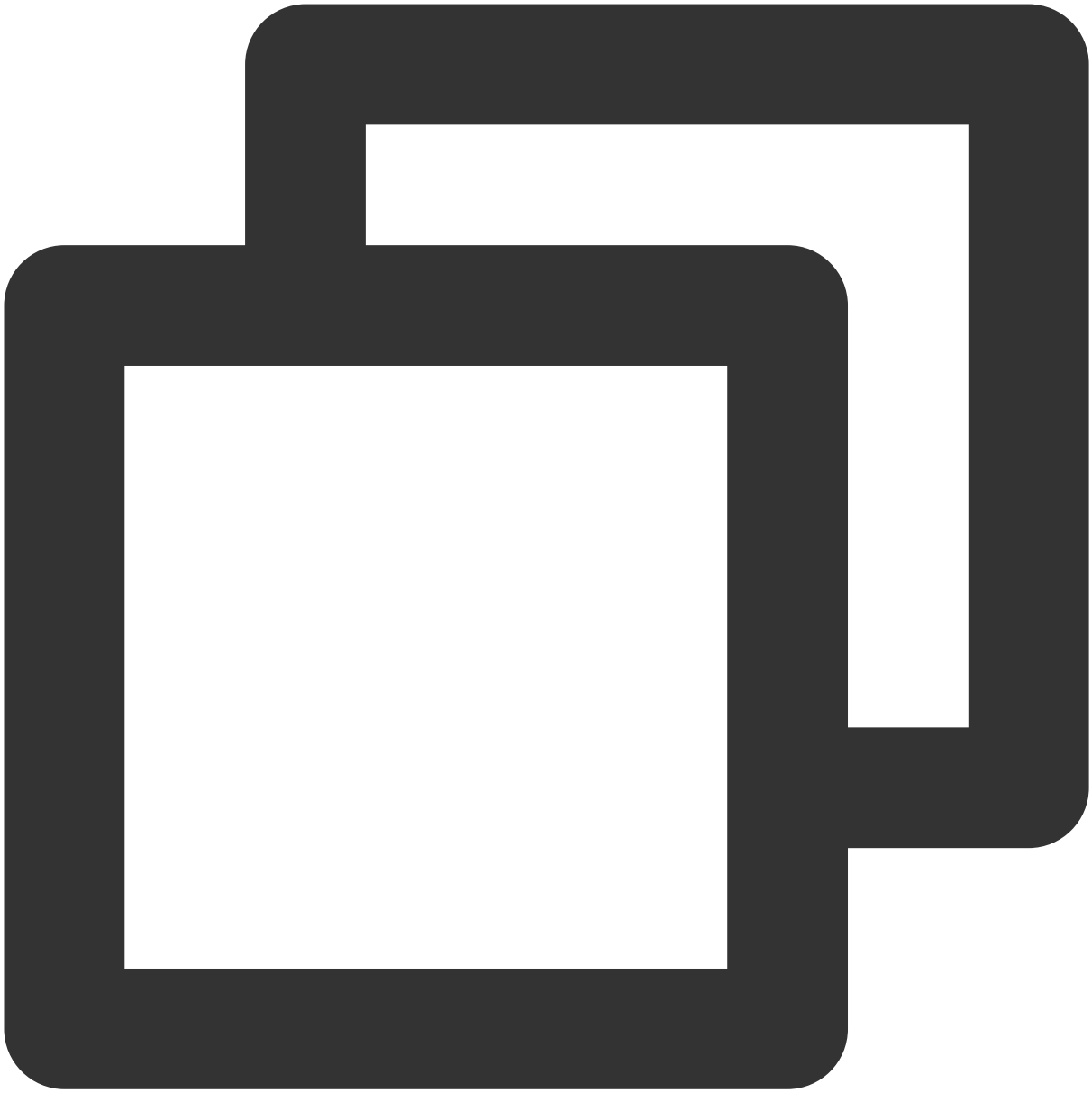
C++



```
public abstract int Init(String sdkAppId, String openId);
```



```
-(int)InitEngine:(NSString*)sdkAppID openID:(NSString*)openID;
```



```
ITMGContext virtual int Init(const char* sdkAppId, const char* openId)
```

パラメータ	タイプ	意味
sdkAppId	string	Tencent Cloud ConsoleのGMEサービスが提供するAppIDです。取得についてはサービス開始ガイドラインをご参照ください。
openID	string	openIDはInt64型（stringに変換して渡す）のみに対応しており、ルールはApp開発

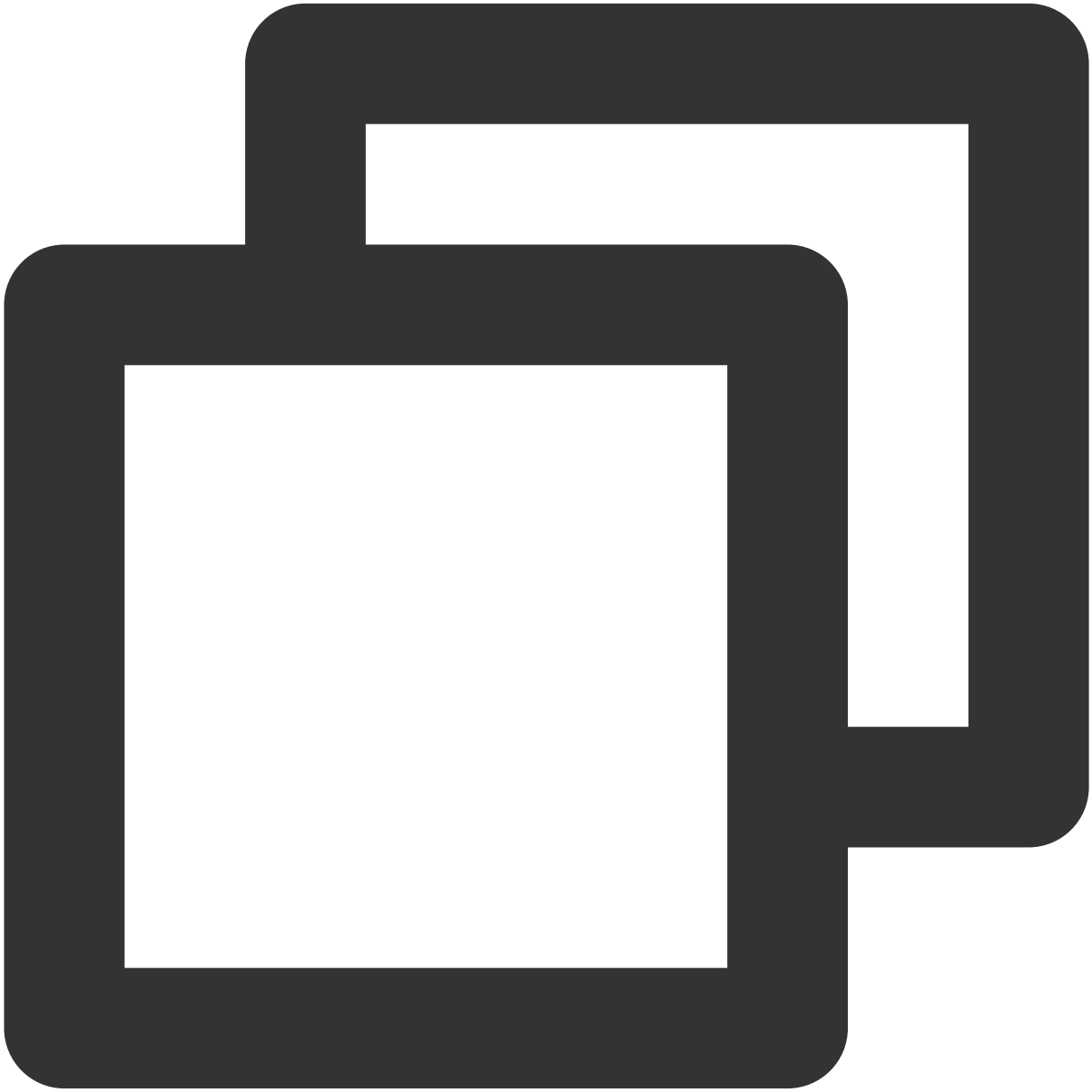
		者が独自に定め、App内で重複しなければよい。文字列をOpenidとして渡す必要がある場合は、チケットを提出をして開発者に連絡してください。

サンプルコード

Java

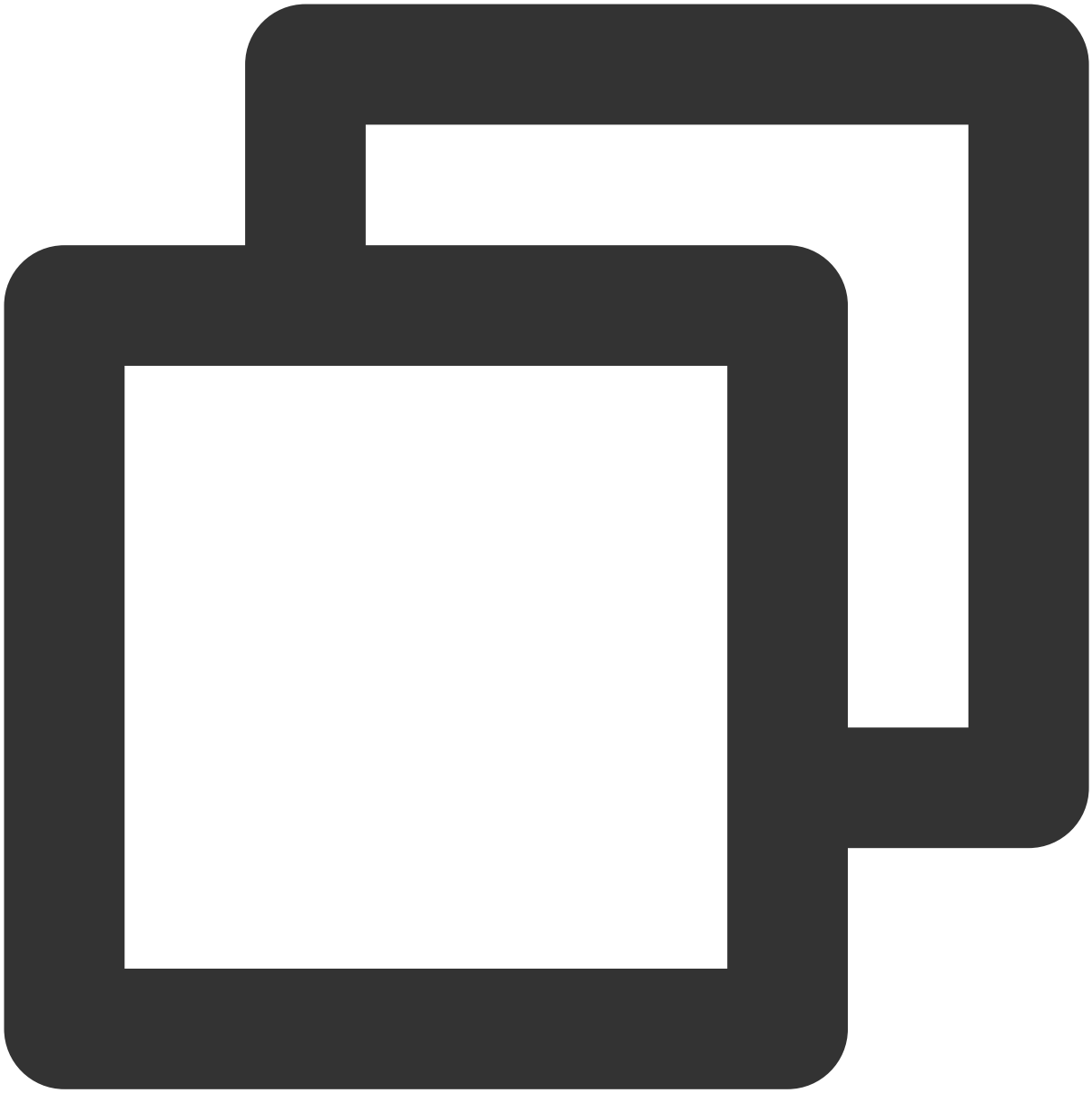
Object-C

C++

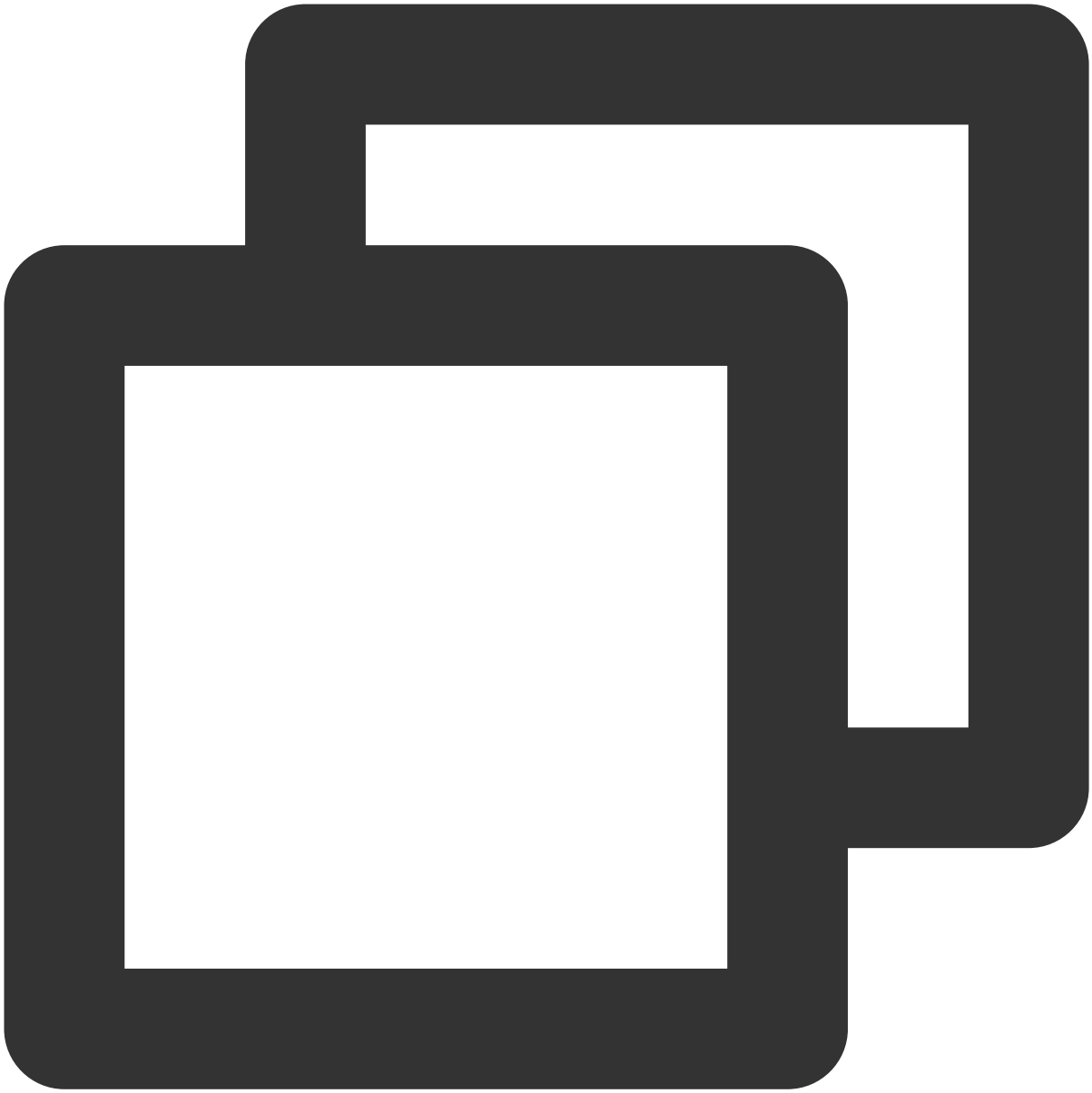


```
//MainActivity.java
int nRet = tmgContext.Init(appId, openId);
```

```
if (nRet == AV_OK )
{
    GMEAuthBufferHelper.getInstance().setGEMParams(appId, key, openId);
    // Step 4/11: Poll to trigger callback
    //https://intl.cloud.tencent.com/document/product/607/40860
    EnginePollHelper.createEnginePollHelper();
    showToast("Init success");
}else if (nRet == AV_ERR_HAS_IN_THE_STATE) // 初期化されました。この操作は成功したと考えら
{
    showToast("Init success");
}else
{
    showToast("Init error errorCode:" + nRet);
}
```



```
//TMGSampleViewController.m
QAVResult result = [_context InitEngine:self.appIDTF.text openID:self.openIDTF.text
if (result == QAV_OK) {
    self.isSDKInit = YES;
}
```



```
#define SDKAPPID3RD "14000xxxxx"
cosnt char* openId="10001";
ITMGContext* context = ITMGContextGetInstance();
context->Init(SDKAPPID3RD, openId);
```

6. イベントコールバックのトリガー

updateで周期的にPollを呼び出すことで、イベントのコールバックをトリガできます。PollはGMEのメッセージポンプであり、GMEはイベントのコールバックをトリガするためにPollインターフェースを定期的に呼び出す必要が

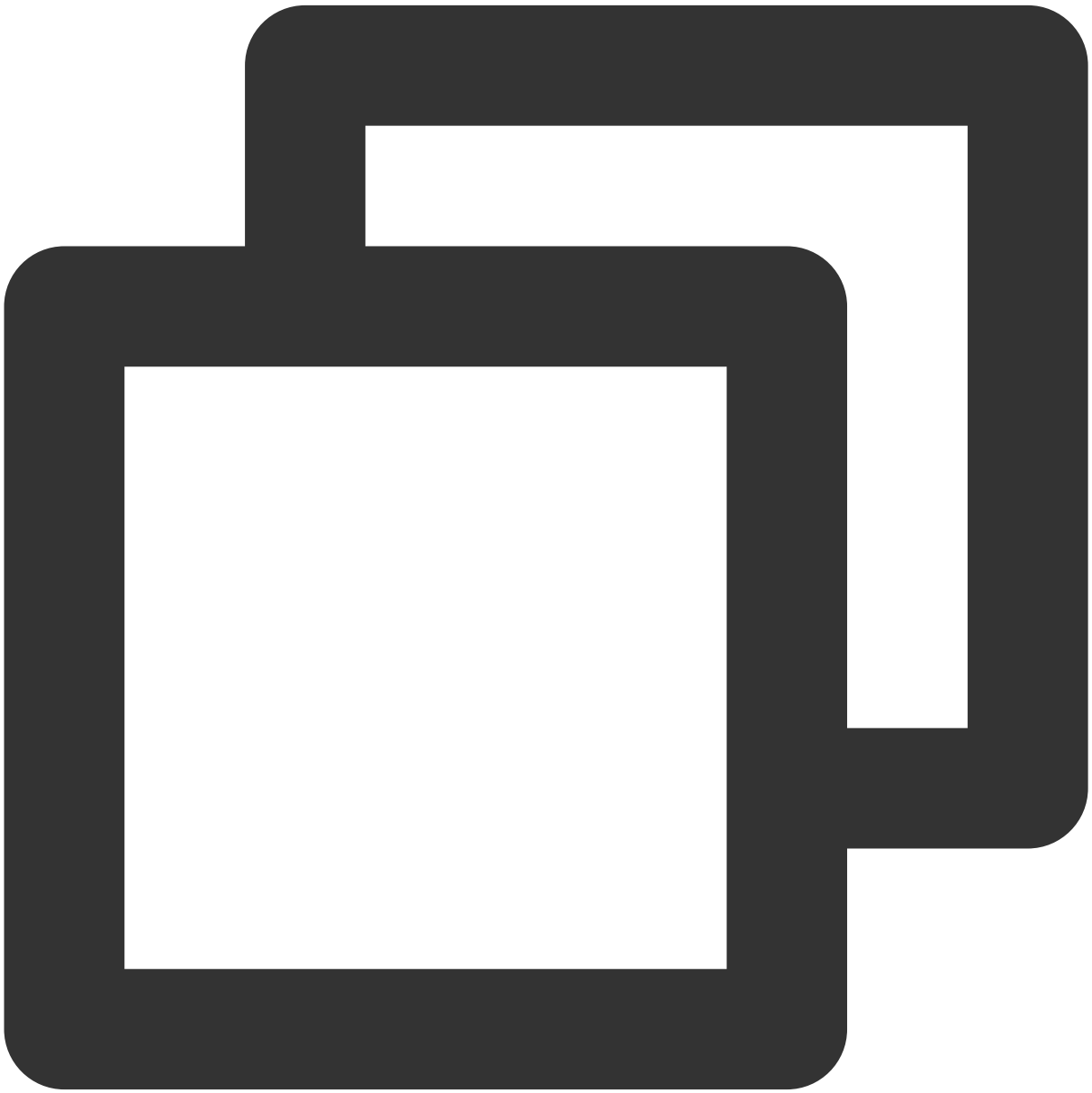
あります。Pollが呼び出されないと、SDKサービス全体が異常に動作します。詳細については、Sample ProjectのEnginePollHelperファイルをご参照ください。

サンプルコード

Java

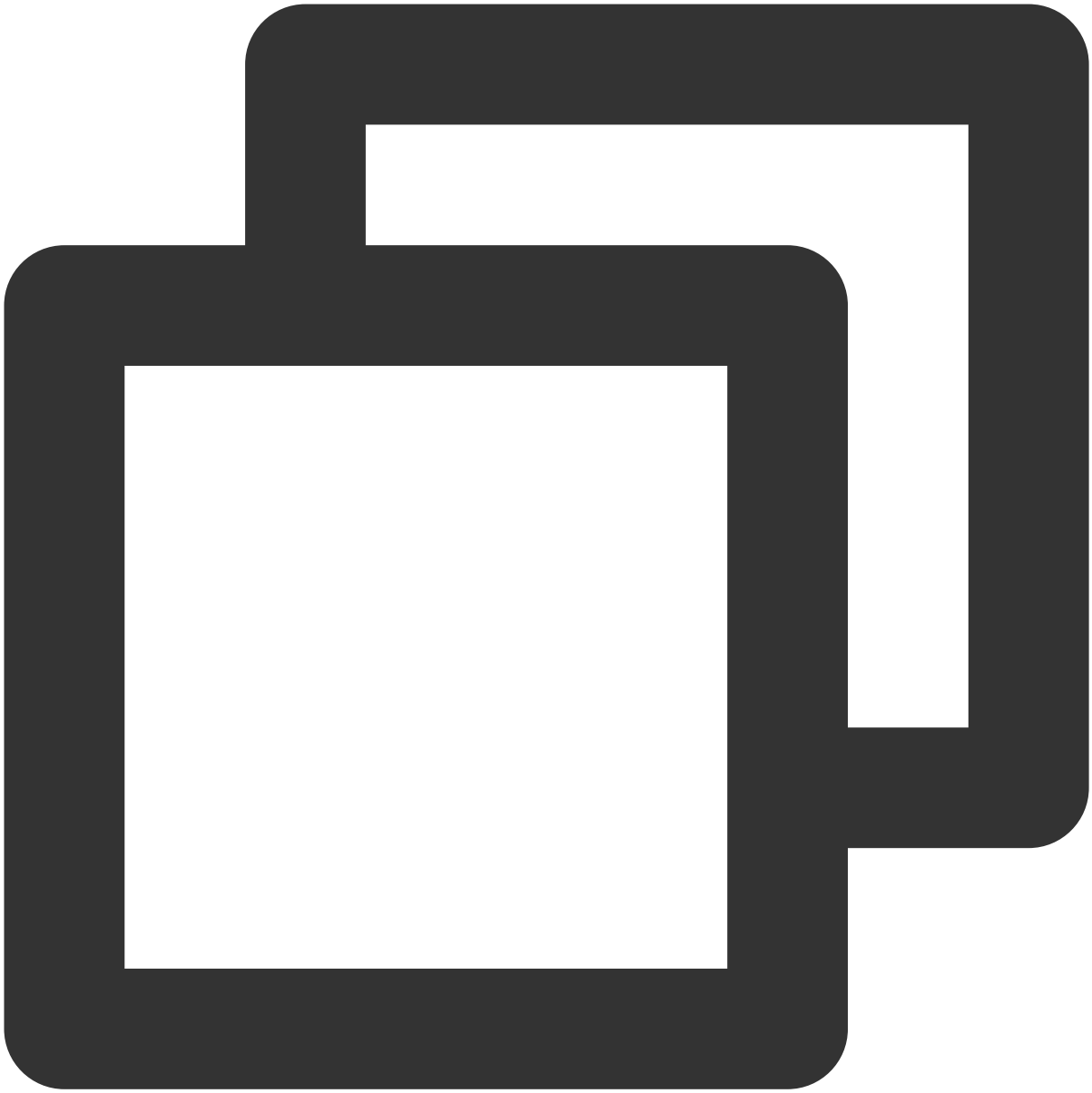
Object-C

C++

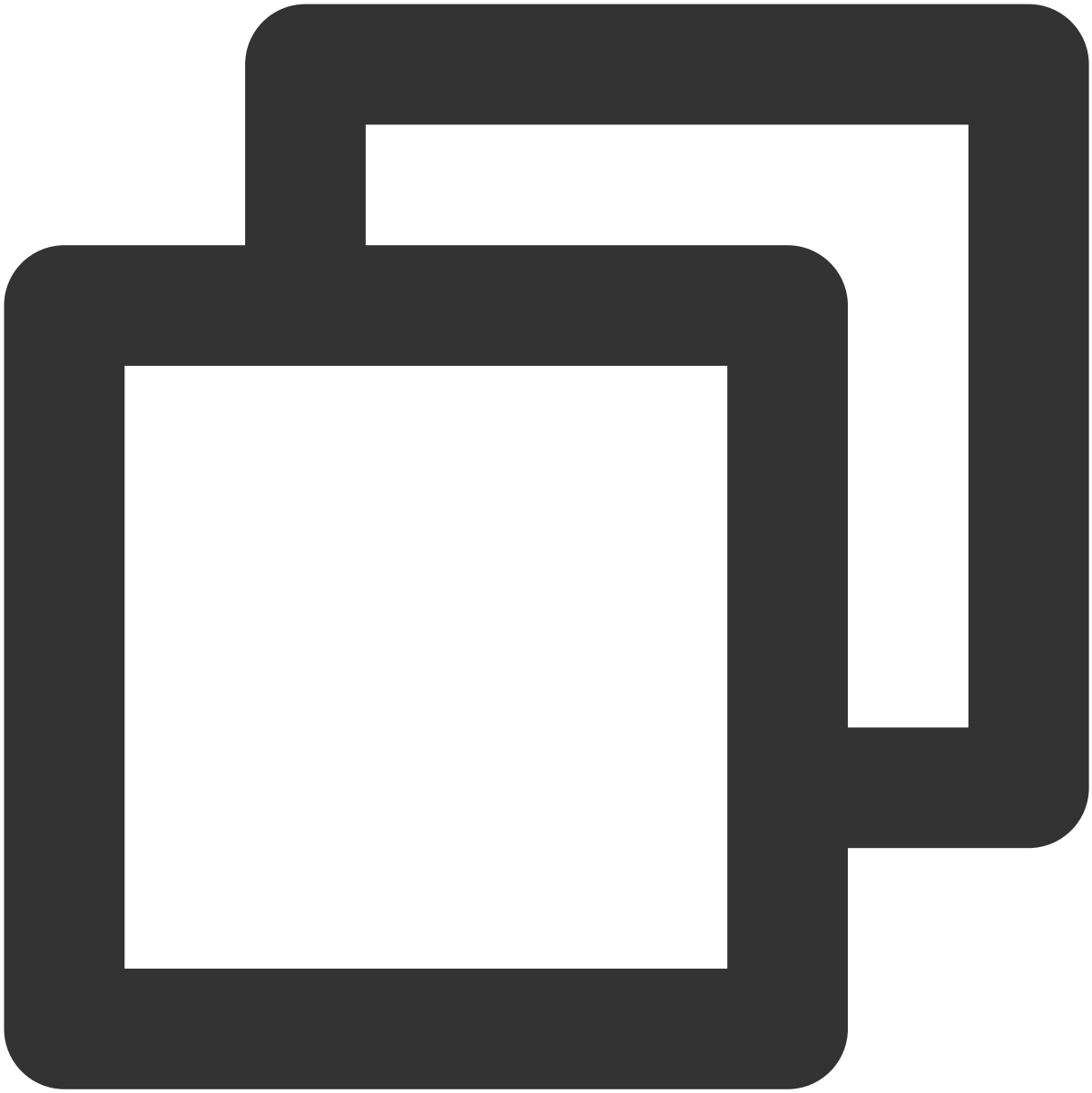


```
//MainActivity.java
[EnginePollHelper createEnginePollHelper];
```

```
//EnginePollHelper.java
private Handler mhandler = new Handler();
    private Runnable mRunnable = new Runnable() {
        @Override
        public void run() {
            if (s_pollEnabled) {
                if (ITMGContext.GetInstance(null) != null)
                    ITMGContext.GetInstance(null).Poll();
            }
            mhandler.postDelayed(mRunnable, 33);
        }
    };
//Pollの周期的な呼び出しについてはEnginePollHelper.javaの書き方をご参照ください
```



```
//TMGSampleViewController.m
[EnginePollHelper createEnginePollHelper];
//EnginePollHelper.mとEnginePollHelper.hを参考にしてください
```



```
void TMGTestScene::update(float delta)
{
  ITMGContextGetInstance()->Poll();
}
```

7. ローカル認証計算

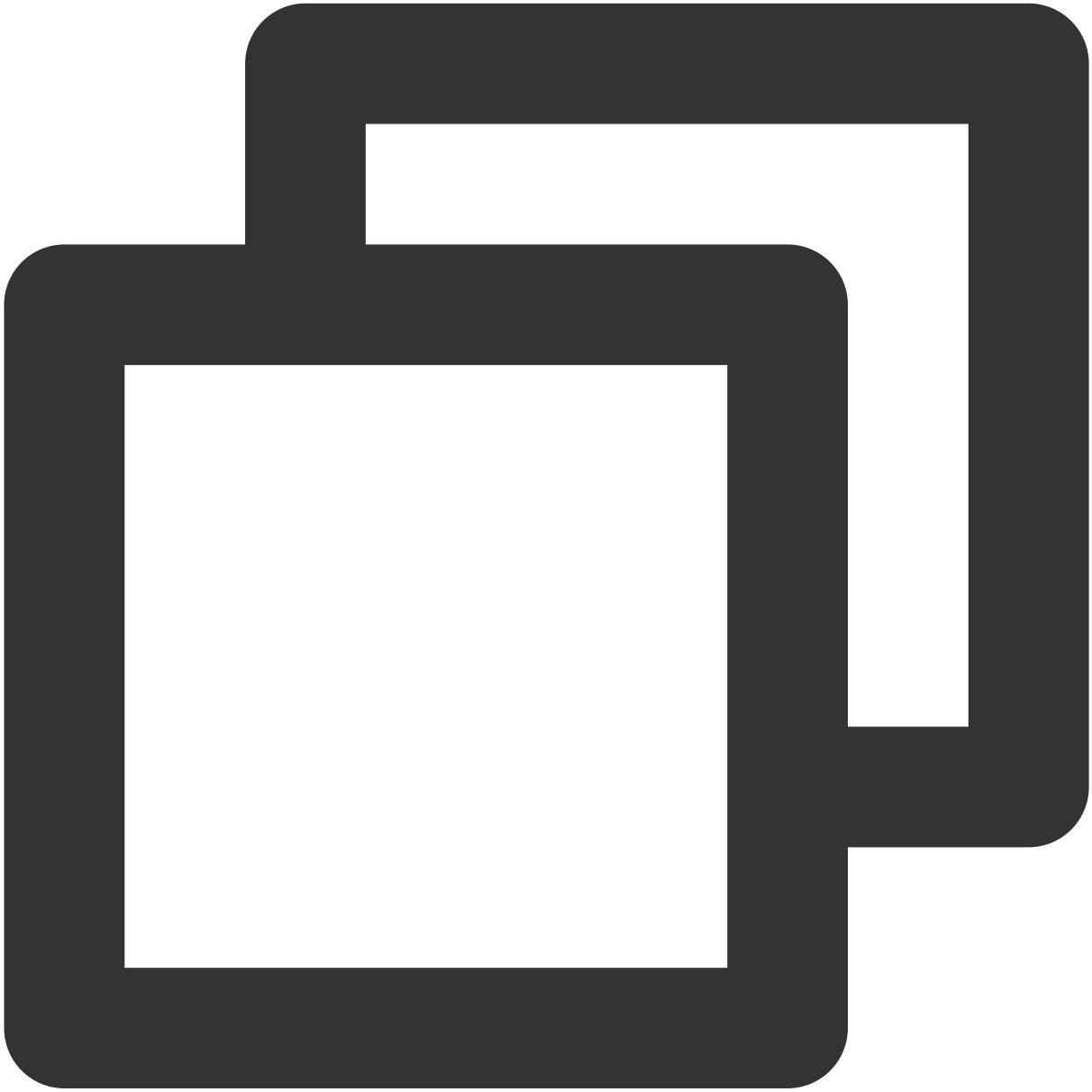
AuthBufferを生成し、関連機能の暗号化と認証に使用します。本格なリリースについてバックグラウンドのデプロイキーを使用してください。バックグラウンドのデプロイについては、認証キーをご参照ください。

インターフェースのプロトタイプ

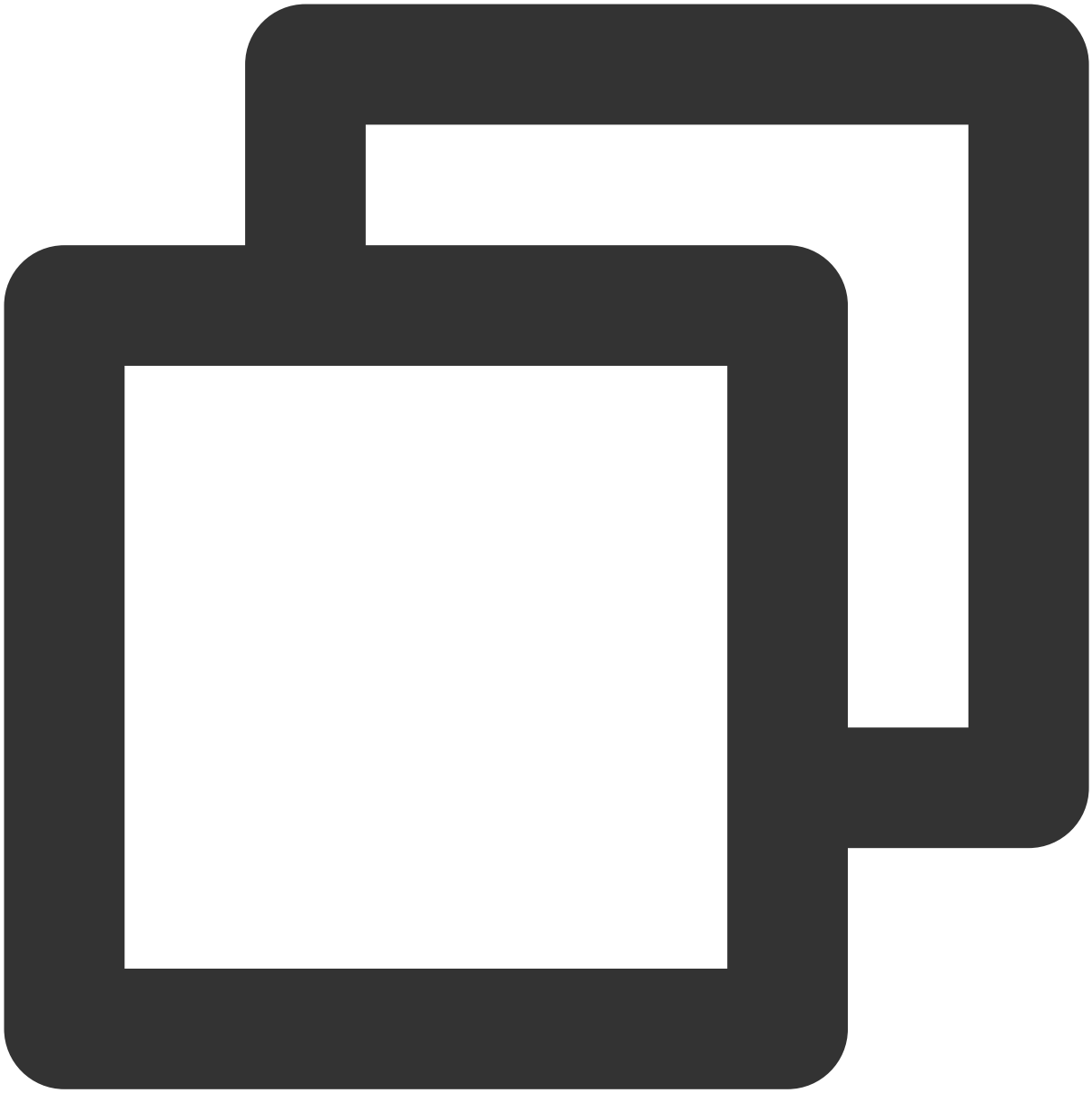
Java

Object-C

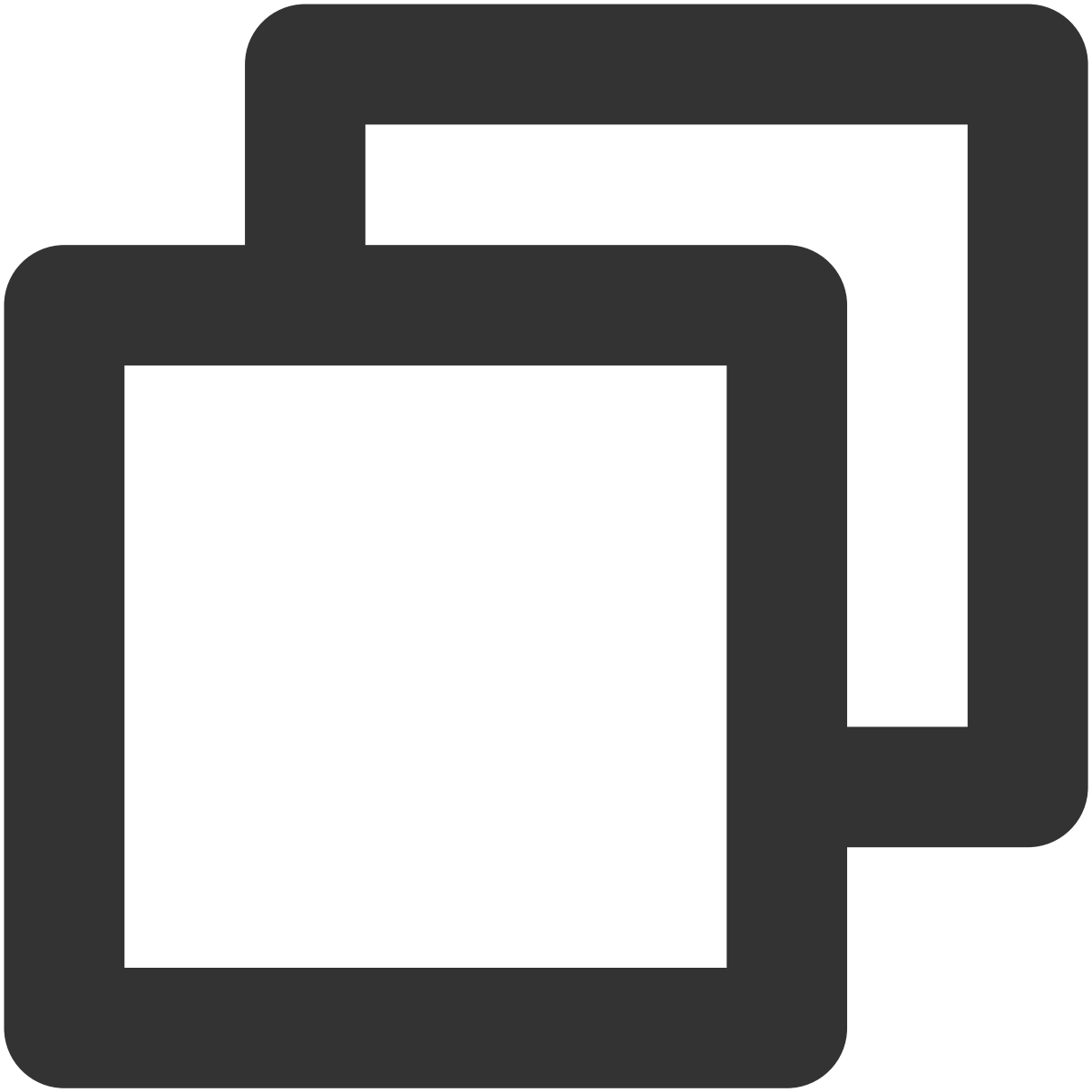
C++



```
AuthBuffer public native byte[] genAuthBuffer(int sdkAppId, String roomId, String o
```



```
//TMGSampleViewController.m
[EnginePollHelper createEnginePollHelper];
//EnginePollHelper.mとEnginePollHelper.hを参考にしてください
```



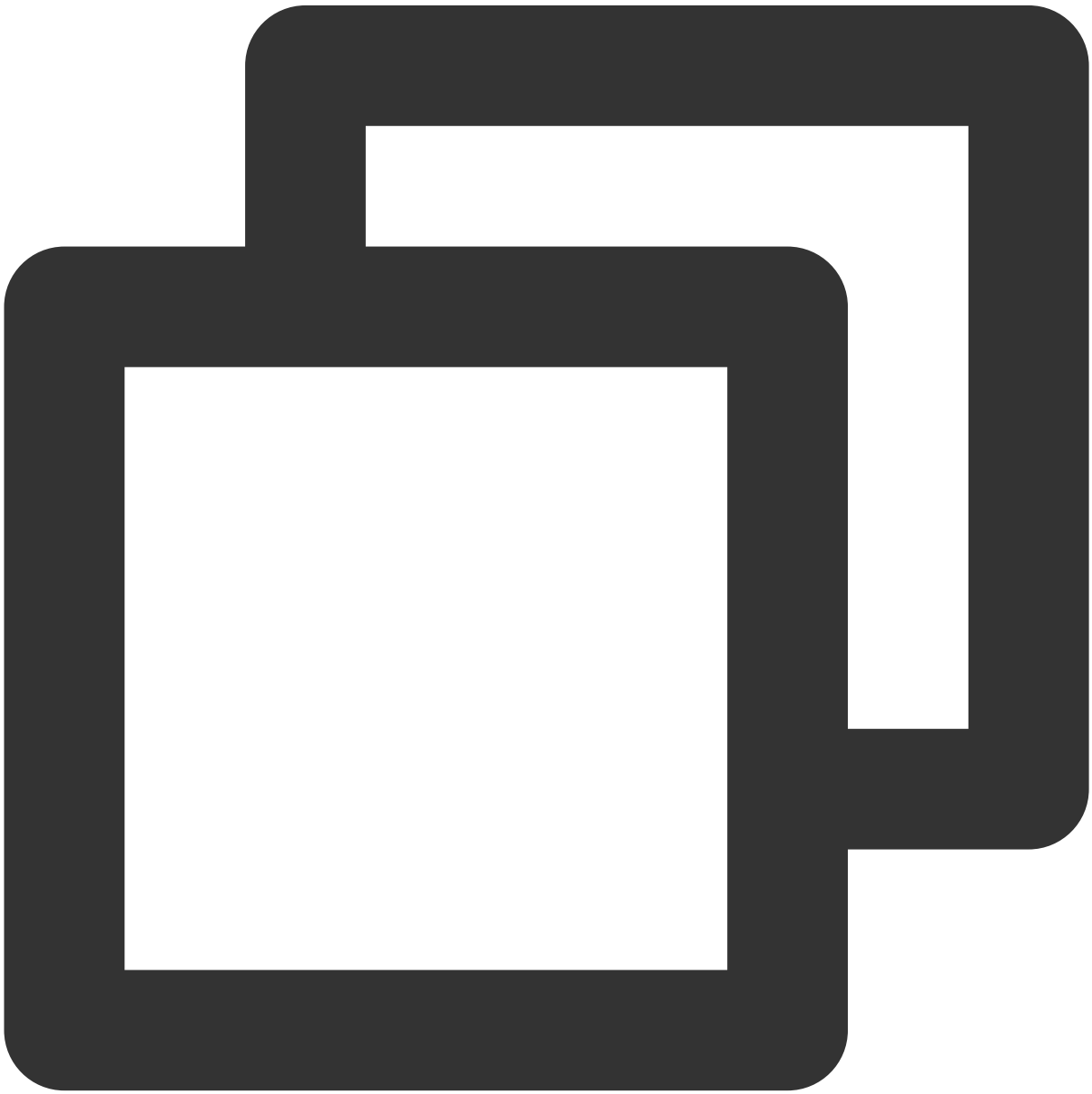
```
void TMGTestScene::update(float delta)
{
  ITMGContextGetInstance()->Poll();
}
```

パラメータ	タイプ	意味
appId	int	Tencent Cloudコンソールからの AppId番号。
roomId	string	ルーム番号であり、最大127文字まで対応しています（オフライン音声ルーム番

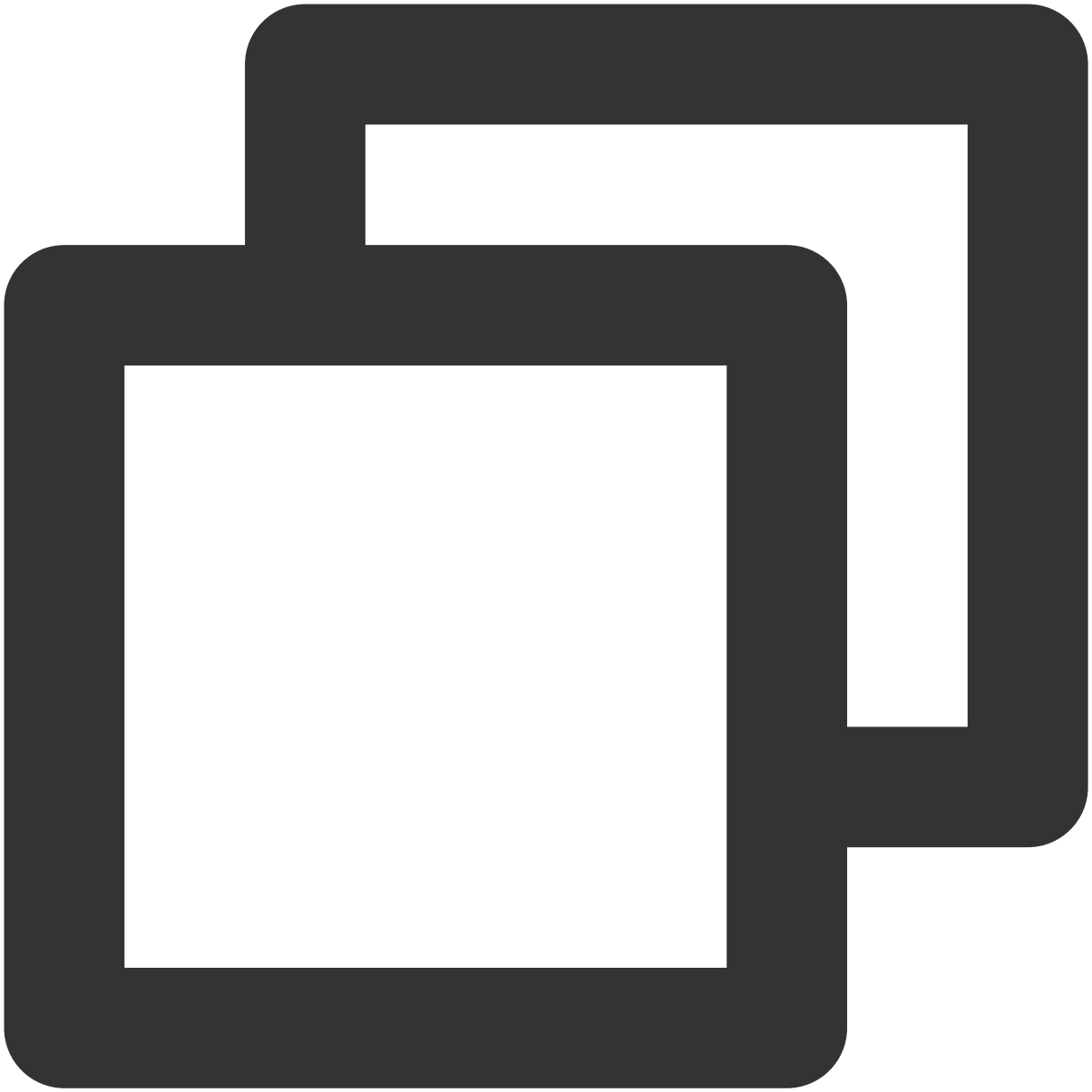
		号のパラメータをnullに設定しなければなりません）。
openId	string	ユーザーID。Initの場合のopenIdと同じです。
key	string	Tencent Cloudコンソールからの権限キー。

サンプルコード

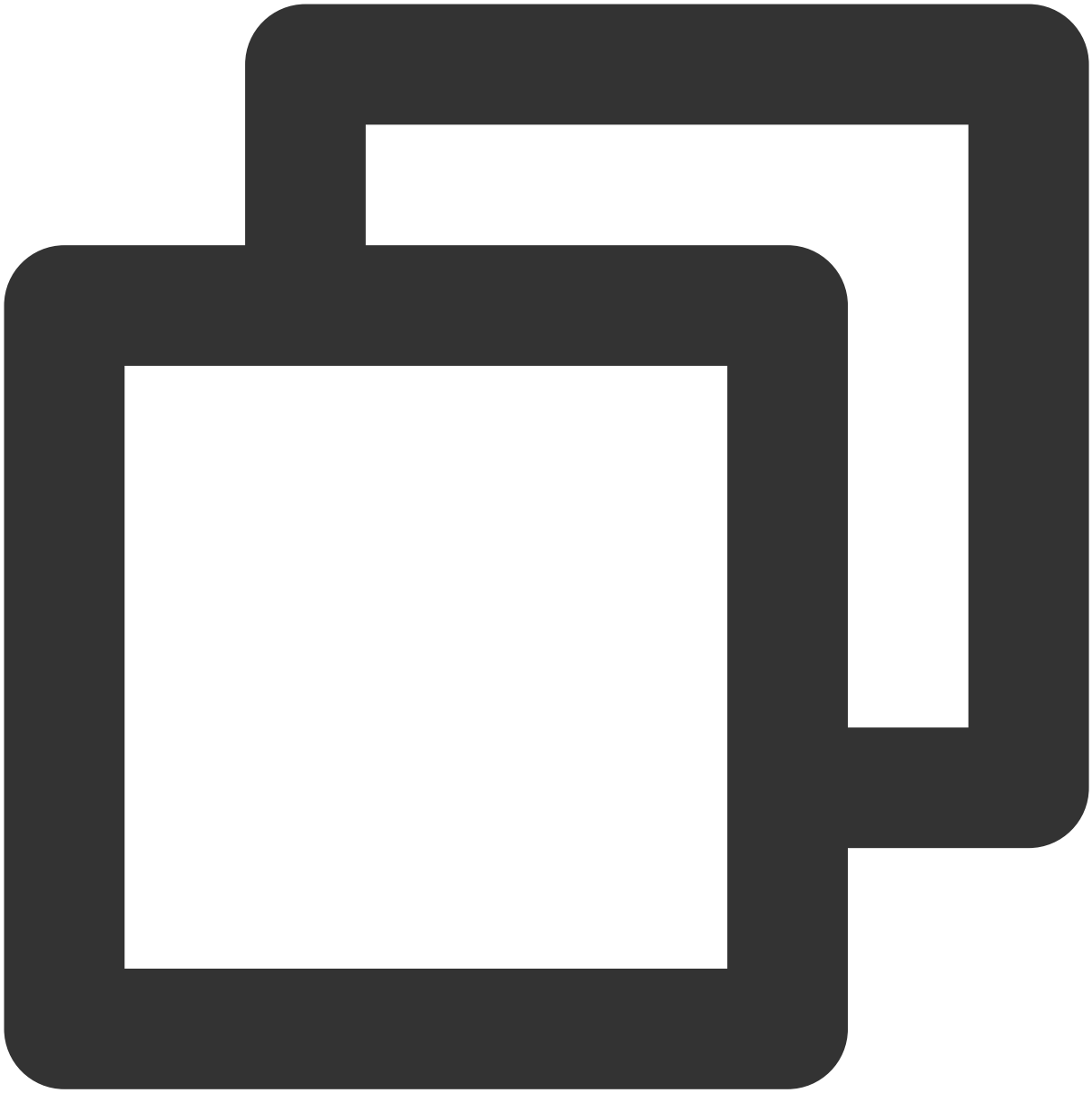
Java
Object-C
C++



```
//GMEAuthBufferHelper.java
import com.tencent.av.sig.AuthBuffer;//ヘッダーファイル
public byte[] createAuthBuffer(String roomId)
    {
        byte[] authBuffer;
        // Generate AuthBuffer for encryption and authentication of relevant featur
        // please use the backend deployment key as detailed in https://intl.cloud.
        if (TextUtils.isEmpty(roomId))
        {
            authBuffer =  AuthBuffer.getInstance().genAuthBuffer(Integer.parseInt(m
        }else
        {
            authBuffer =  AuthBuffer.getInstance().genAuthBuffer(Integer.parseInt(m
        }
        return authBuffer;
    }
```



```
//リアルタイム音声認証
NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:SDKAPPID3RD.intValue roomID:self.
//音声メッセージ認証
NSData* authBuffer =  [QAVAuthBuffer GenAuthBuffer:(unsigned int)SDKAPPID3RD.intege
```



```
unsigned int bufferLen = 512;
unsigned char retAuthBuff[512] = {0};
QAVSDK_AuthBuffer_GenAuthBuffer(atoi(SDKAPPID3RD), roomId, "10001", AUTHKEY,retAuth
```

リアルタイム音声アクセス

1. ルームに参加

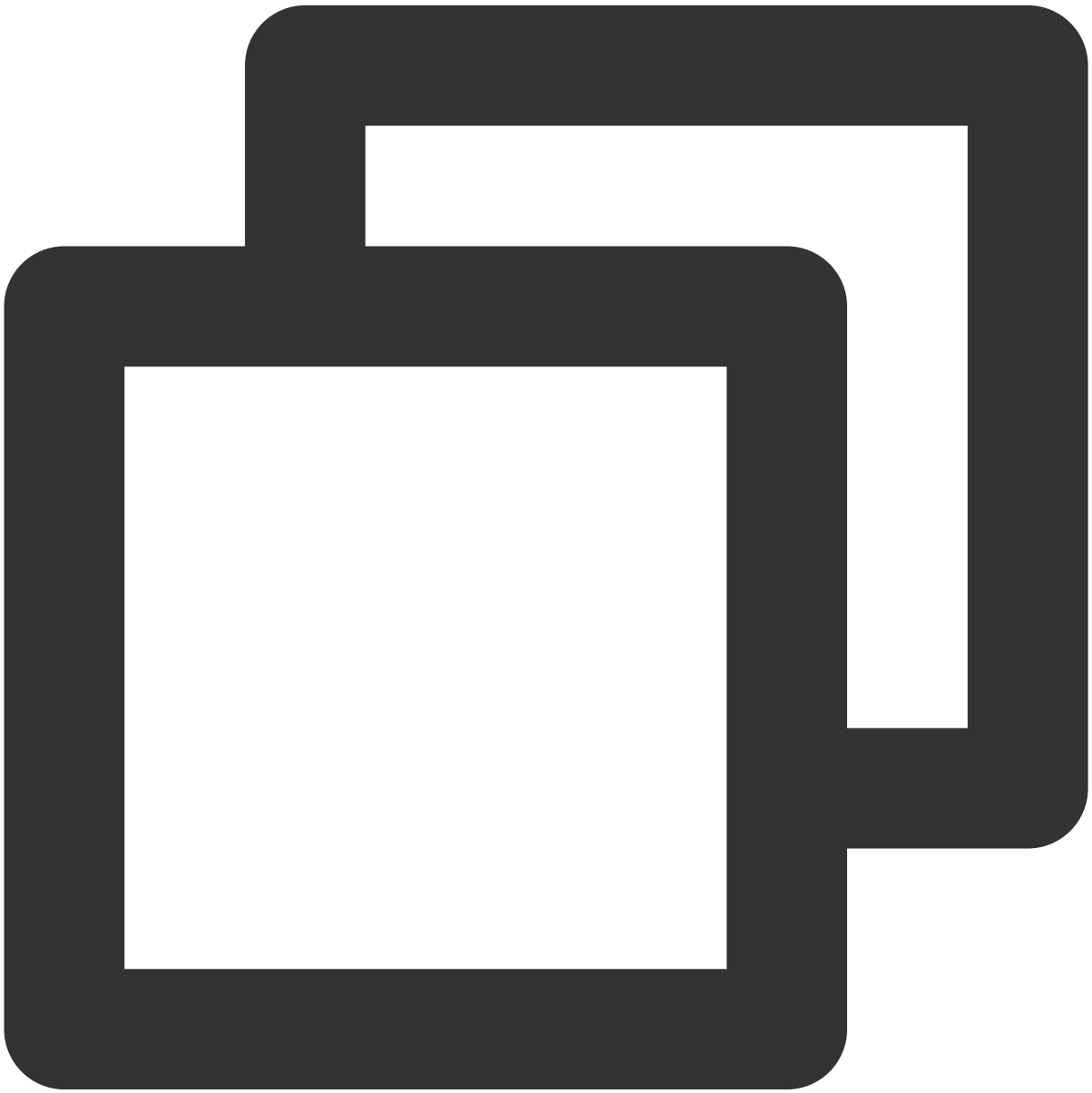
生成した認証情報を用いてルームに参加します。ルームに参加するとき、デフォルトでマイクとスピーカーはオフです。戻り値がAV_OKの場合はルーム参加が成功したことでなく、呼び出しが成功したことを意味します。

インターフェースのプロトタイプ

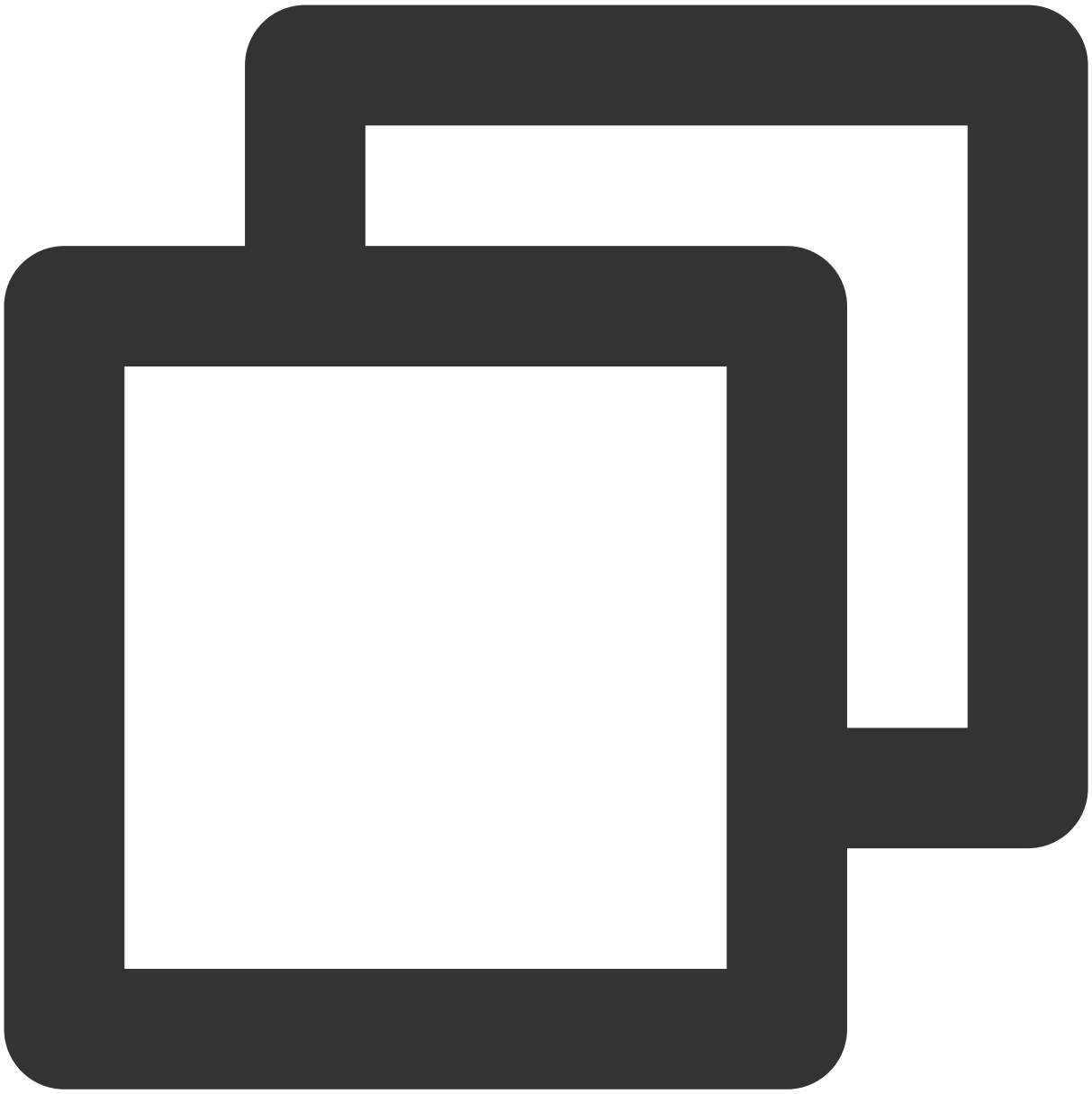
Java

Object-C

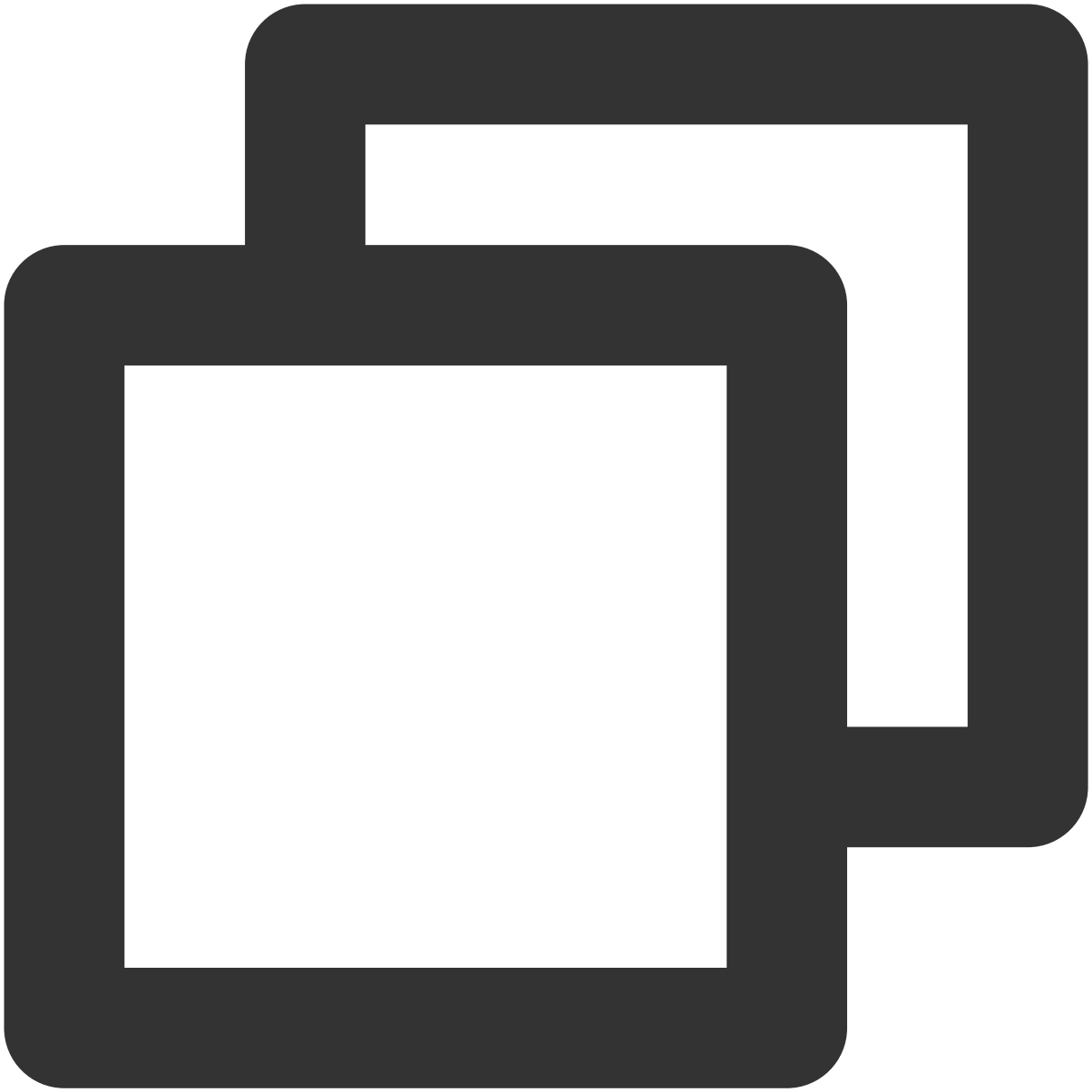
C++



```
public abstract int EnterRoom(String roomID, int roomType, byte[] authBuffer);
```



```
-(int)EnterRoom:(NSString*) roomId roomType:(int)roomType authBuffer:(NSData*)authB
```

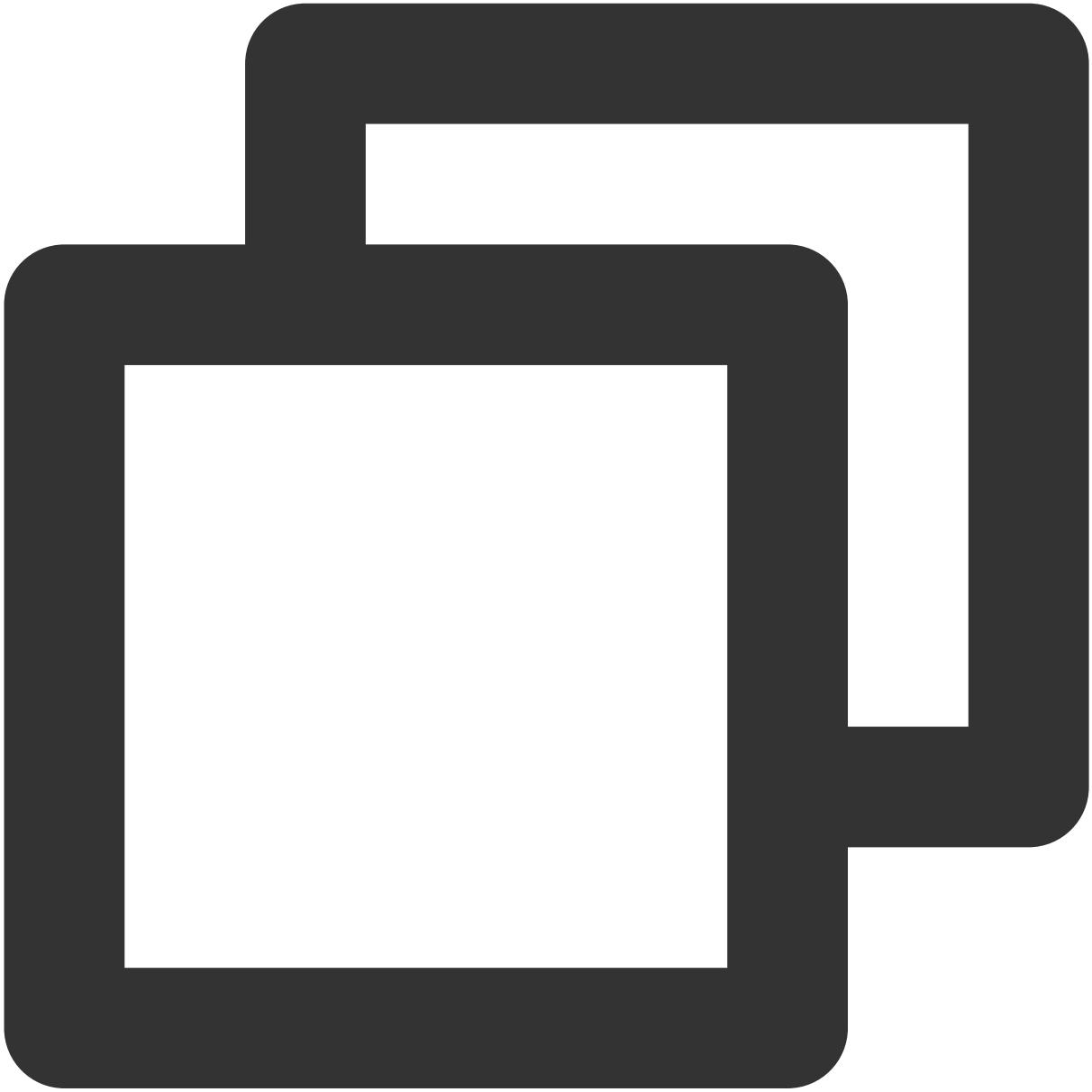


```
ITMGContext virtual int EnterRoom(const char*  roomID, ITMG_ROOM_TYPE roomType, con
```

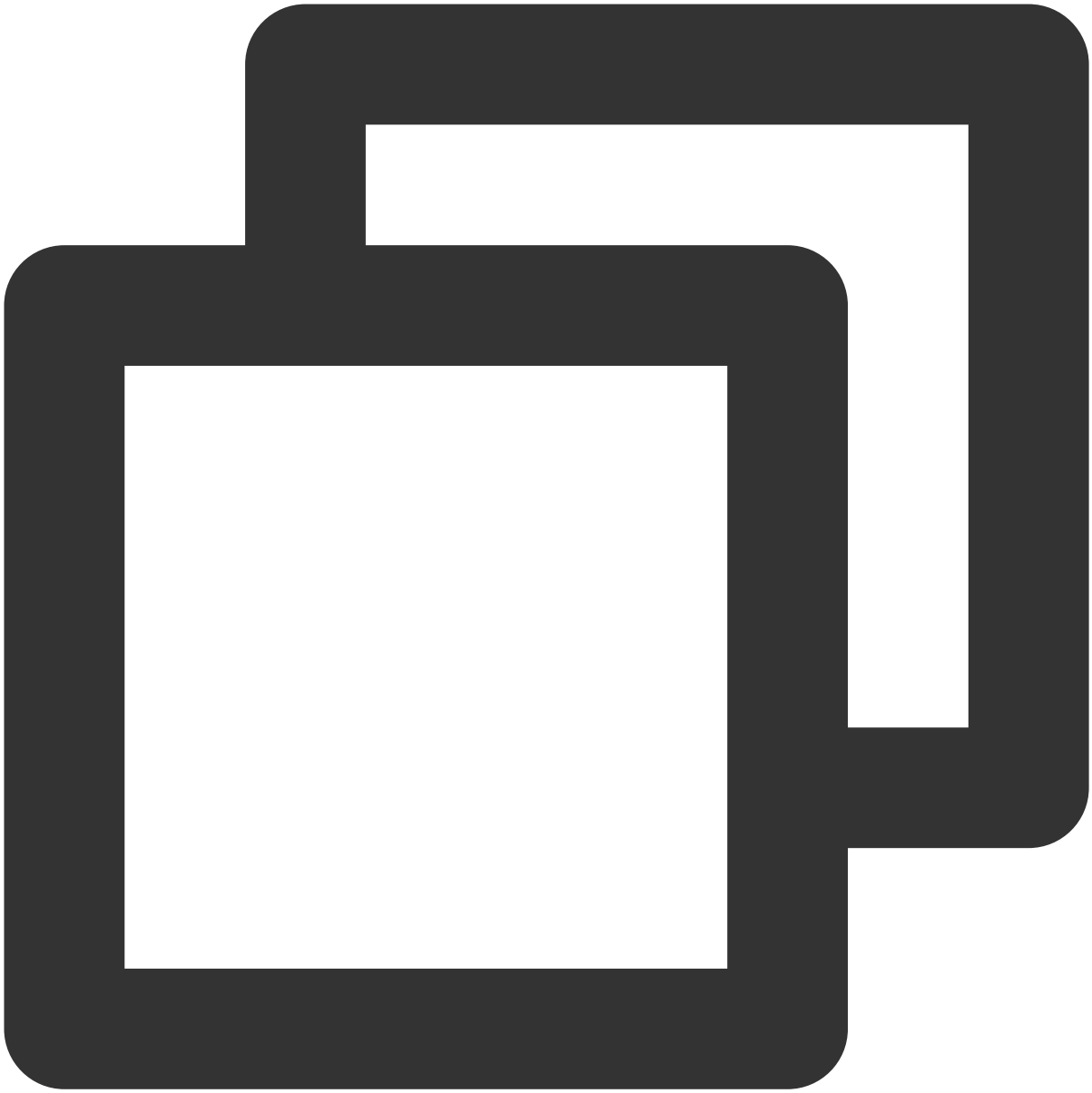
パラメータ	タイプ	意味
roomId	String	ルーム番号、127文字まで入力可能
roomType	int	FLUENCYタイプの音質を使用してルームに参加してください
authBuffer	byte[]	認証コード

サンプルコード

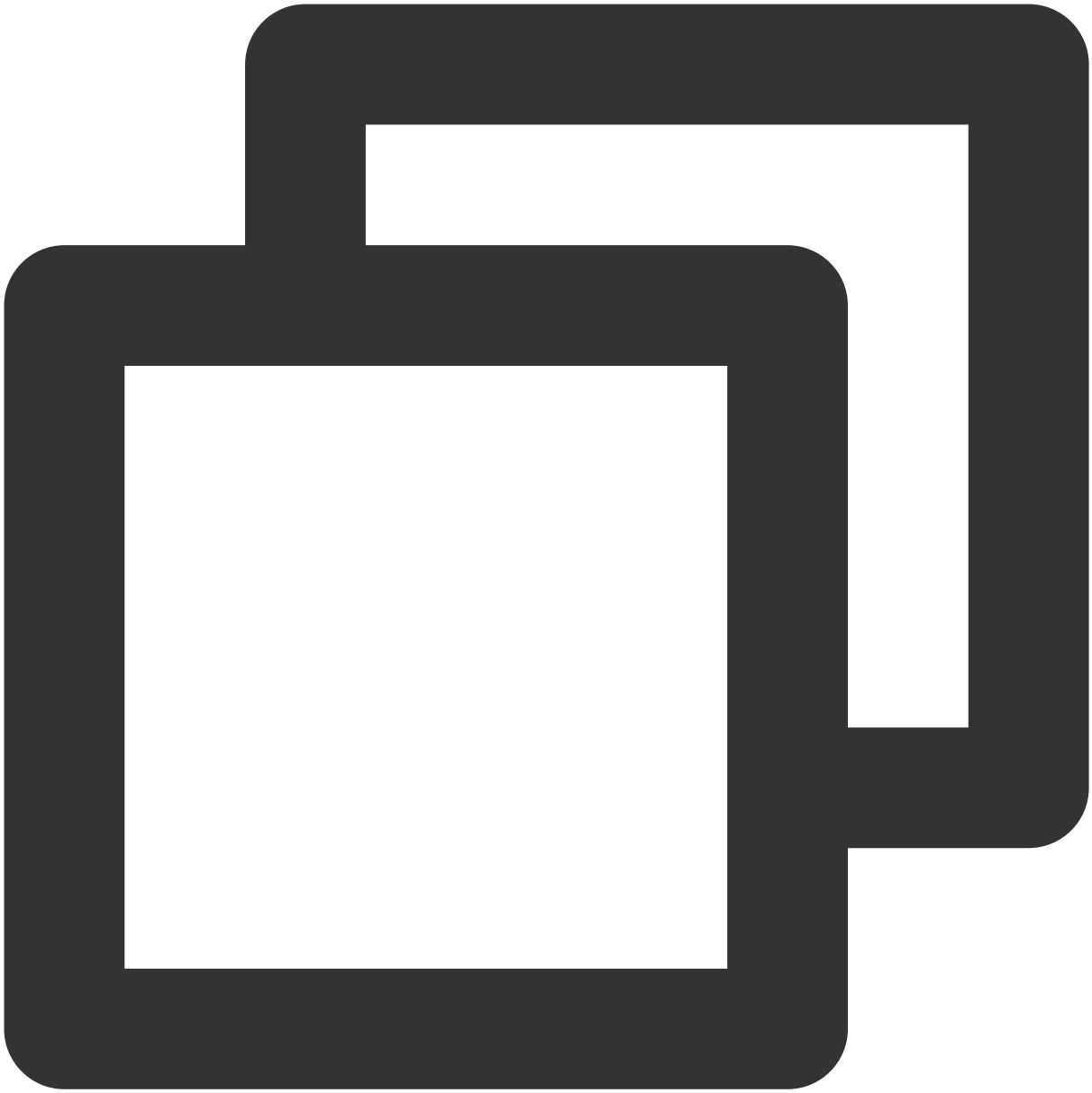
Java
Object-C
C++



```
//RealTimeVoiceActivity.java
byte[] authBuffer =  GMEAuthBufferHelper.getInstance().createAuthBuffer(roomId);
ITMGContext.GetInstance(this).EnterRoom(roomId, roomType, authBuffer);
```



```
//TMGRealTimeViewController.m
[[ITMGContext GetInstance] EnterRoom:self.roomIdTF.text roomType:(int)self.roomType
```



```
ITMGContext* context = ITMGContextGetInstance();
context->EnterRoom(roomID, ITMG_ROOM_TYPE_FLUENCY, (char*)retAuthBuff,bufferLen);
```

入室イベントのコールバック

入室後とメッセージITMG_MAIN_EVENT_TYPE_ENTER_ROOMを送信し、OnEvent関数でコールバックを判定した後に処理します。コールバックが成功した場合、その時点で正常に入室したと判定し、**課金**が開始されます。

计费问题参考

購入ガイド。

課金に関するよくあるご質問。

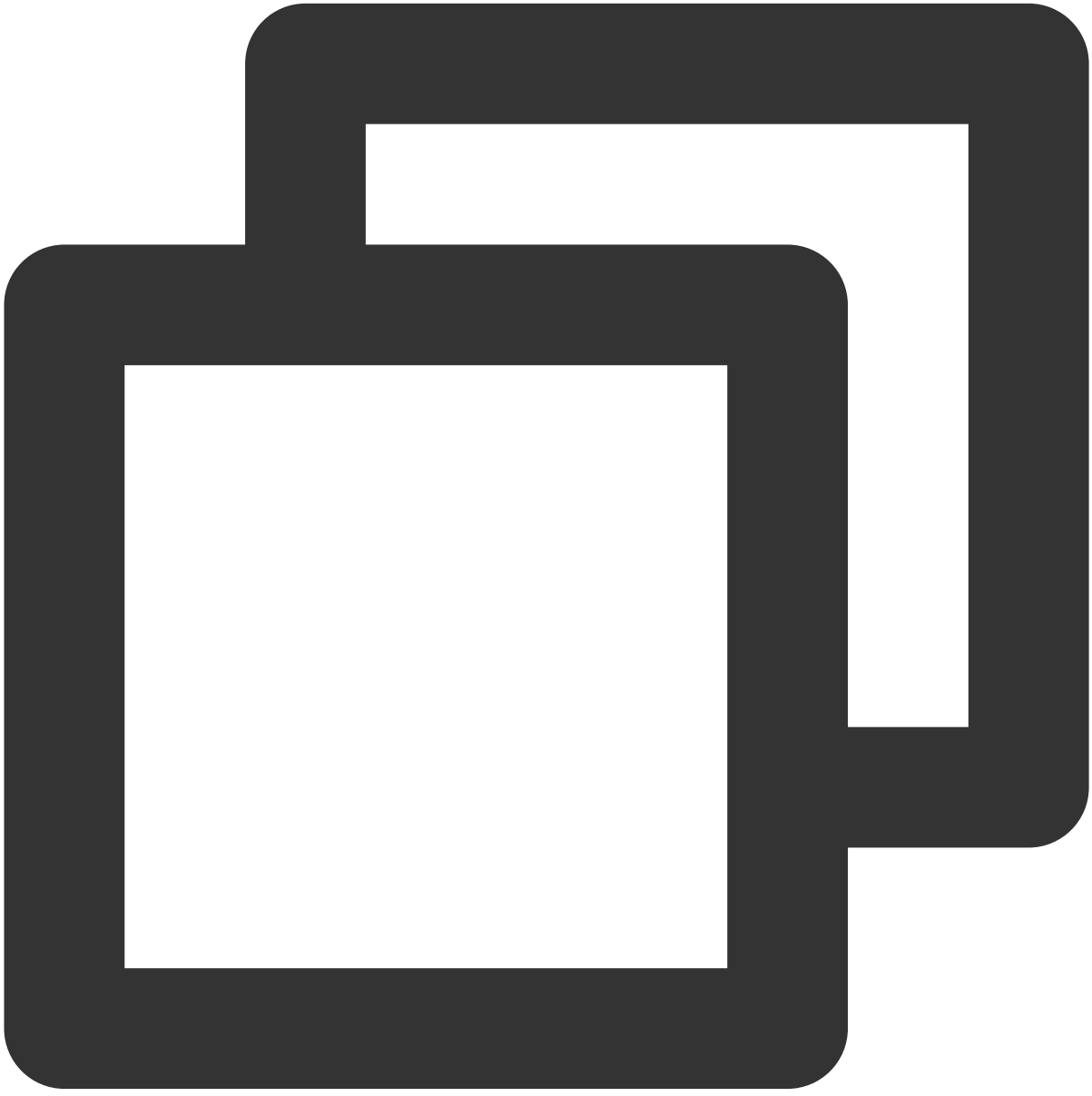
リアルタイム音声を使用した後、クライアントの接続が切れた場合課金は継続されますか。

サンプルコードルーム参加イベントや接続切断イベントなど、コールバック処理の関連コードです。

Java

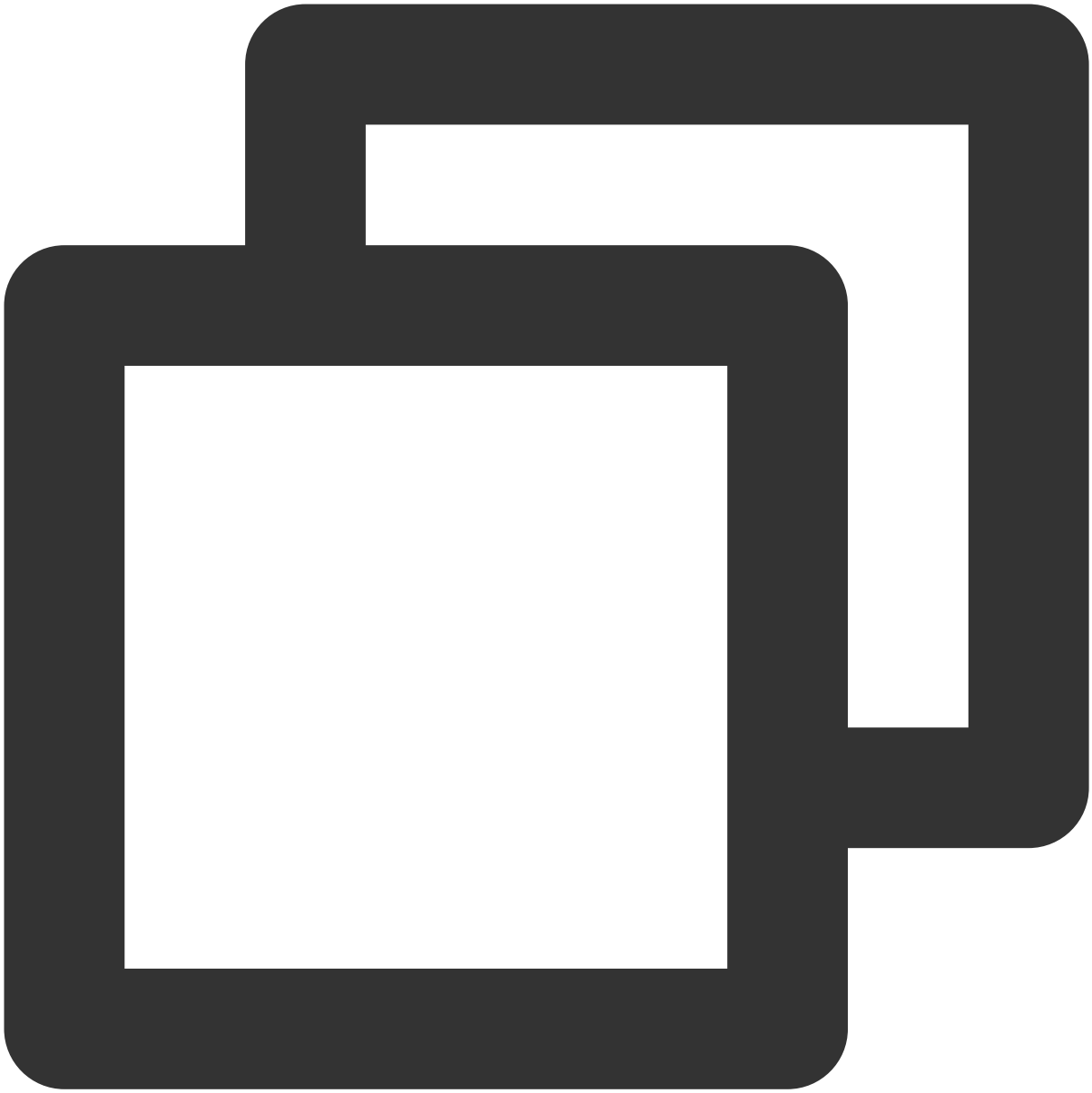
Object-C

C++



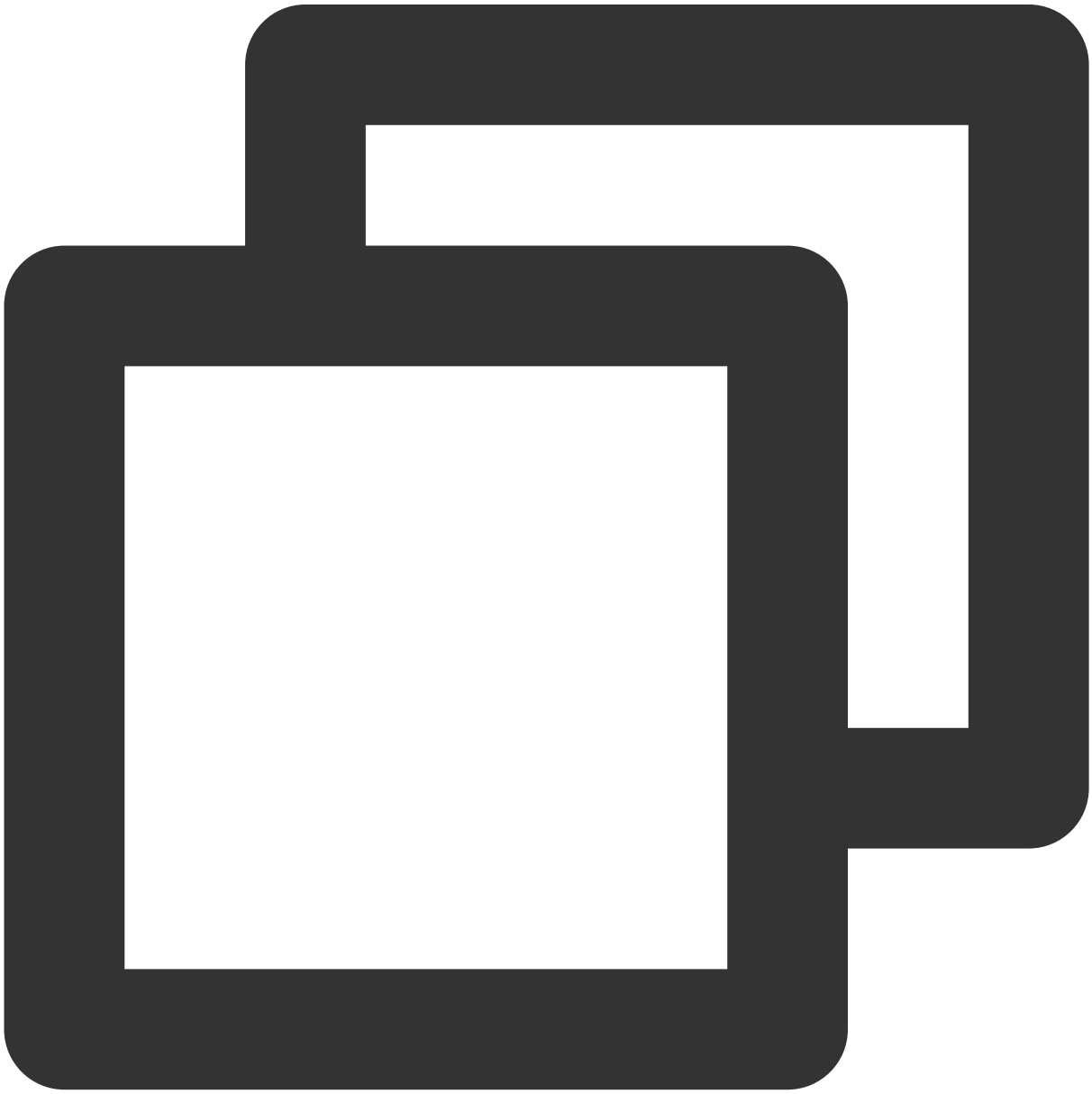
```
//RealTimeVoiceActivity.java
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if (type == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM)
```

```
    {
        // Step 6/11 : Perform the enter room event
        int nErrCode = TMGCallbackHelper.ParseIntentParams2(data).nErrCode;
        String strMsg = TMGCallbackHelper.ParseIntentParams2(data).strErrMsg;
        if (nErrCode == AV_OK)
        {
            appendLog2MonitorView("EnterRomm success");
        }else
        {
            appendLog2MonitorView(String.format(Locale.getDefault(), "EnterRomm err
        }
    }
}
```



```
//TMGRealTimeViewController.m

- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data {
  NSString *log = [NSString stringWithFormat:@"OnEvent:%d,data:%@", (int)eventType,
  [self showLog:log];
  NSLog(@"====%@====", log);
  switch (eventType) {
      // Step 6/11 : Perform the enter room event
      case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM: {
          int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
          NSString* error_info = [data objectForKey:@"error_info"];
```

```
            [self showLog:[NSString stringWithFormat:@"OnEnterRoomComplete:%d msg:(%@

            if (result == 0) {
                [self updateStatusEnterRoom:YES];
            }
        }
        break;

    }
```



```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
switch (eventType) {
    case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
    {
        ListMicDevices();
        ListSpeakerDevices();
            std::string strText = "EnterRoom complete: ret=";
        strText += data;
        m_EditMonitor.SetWindowText(MByteToWChar(strText).c_str());
        }
}
}
```

エラーコード

エラーコードの値	原因と解決策
7006	次の理由で認証に失敗しました： AppIDが存在しないか、エラーです authbuff認証エラーです 認証期限切れです openIdが仕様に準拠していません
7007	他のルームにいます
1001	ルーム参加中でこの操作を繰り返しています。コールバックが戻るまで、ルーム参加インターフェースを呼び出さないことをお勧めします
1003	ルームに参加してルームにいますが、もう1回ルーム参加インターフェースを呼び出しました
1101	SDKが初期化されていること、openIdが規則に準拠していること、またはインターフェースが同じスレッドで呼び出されていること、およびPollインタフェースが正常に呼び出されていることを確認してください

2. マイクのオン/オフ

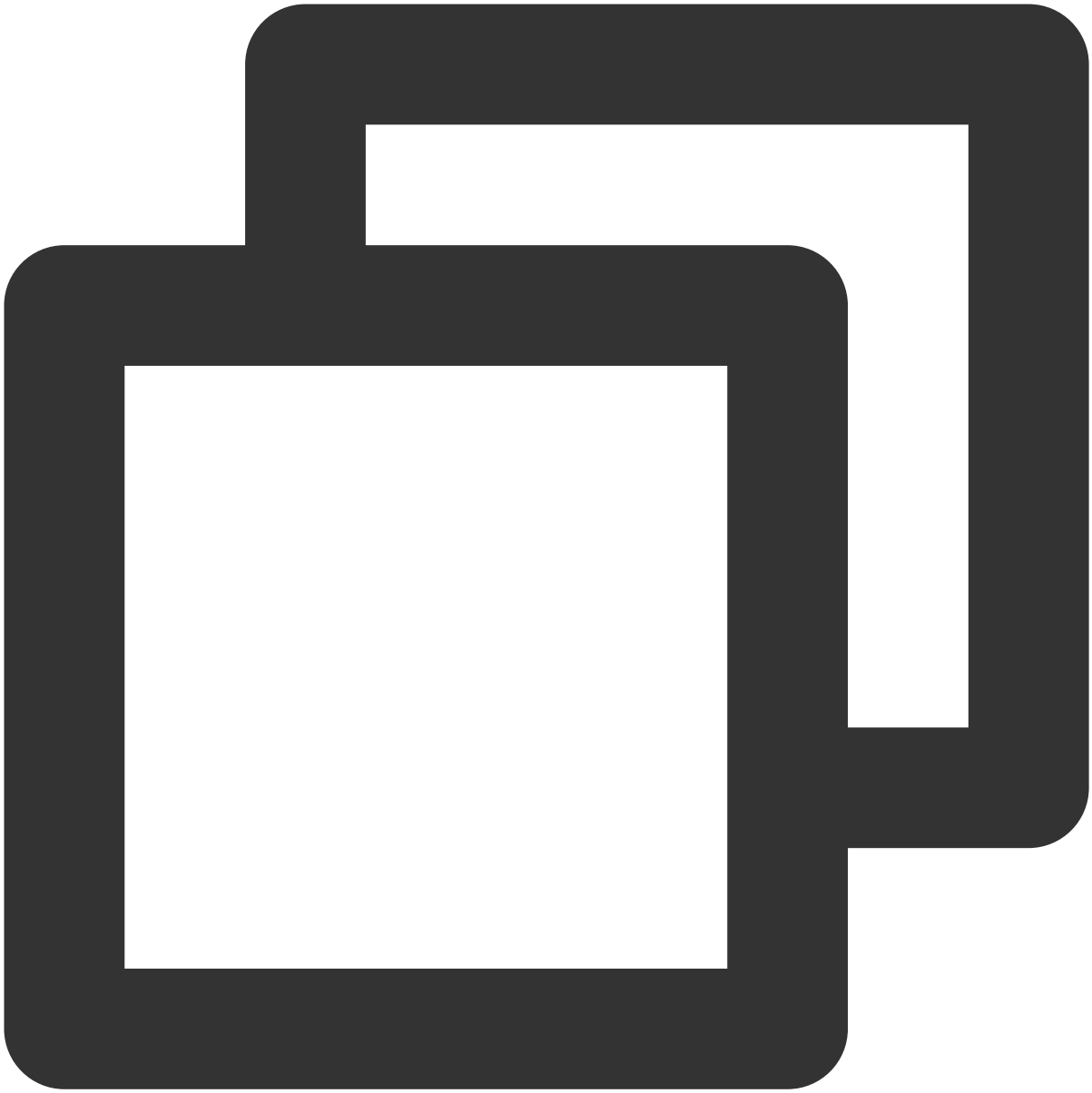
このインターフェースは、マイクのオン/オフに使用されます。入室する際、マイクとスピーカーはデフォルトでオフになっています。

サンプルコード

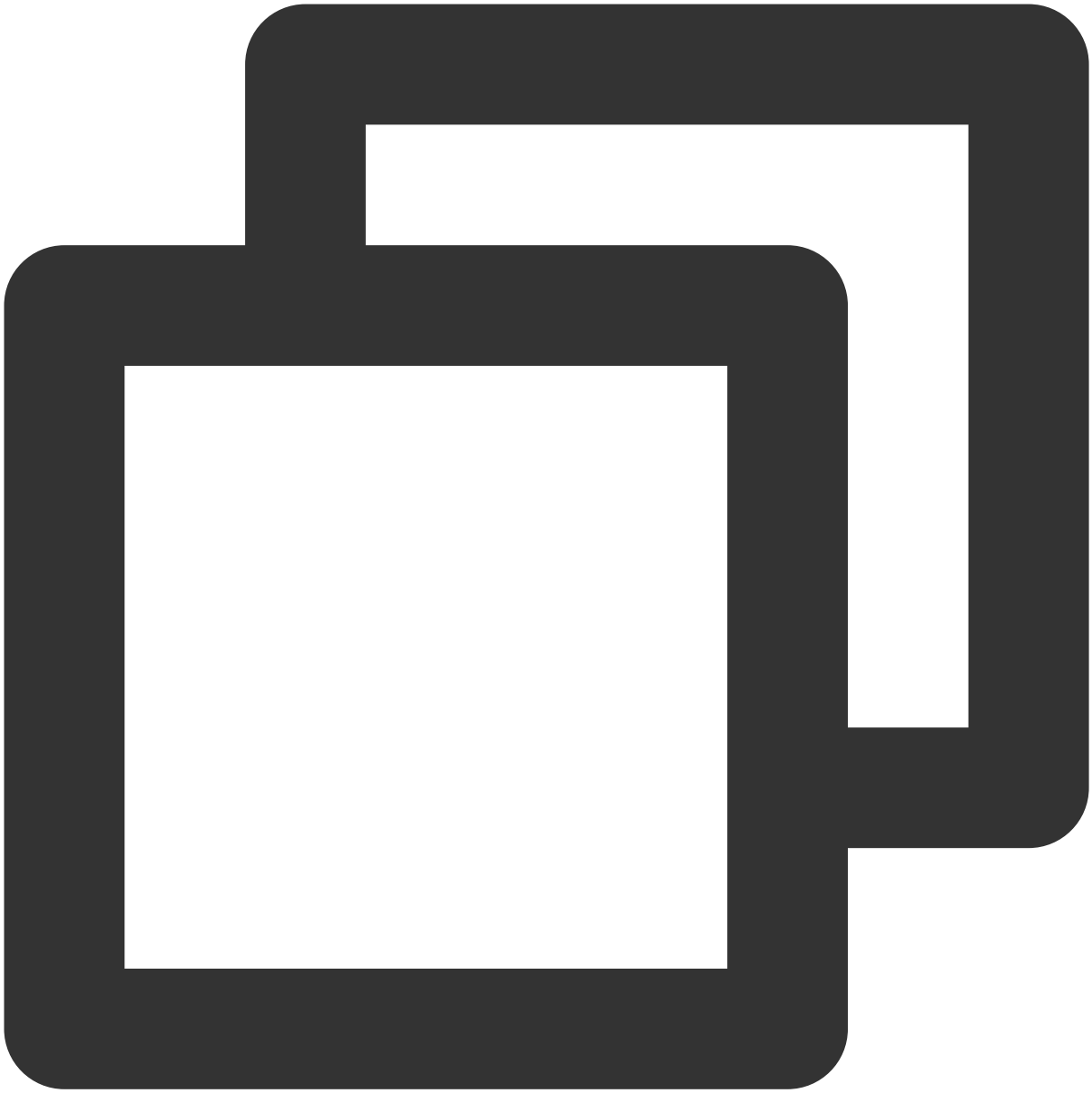
Java

Object-C

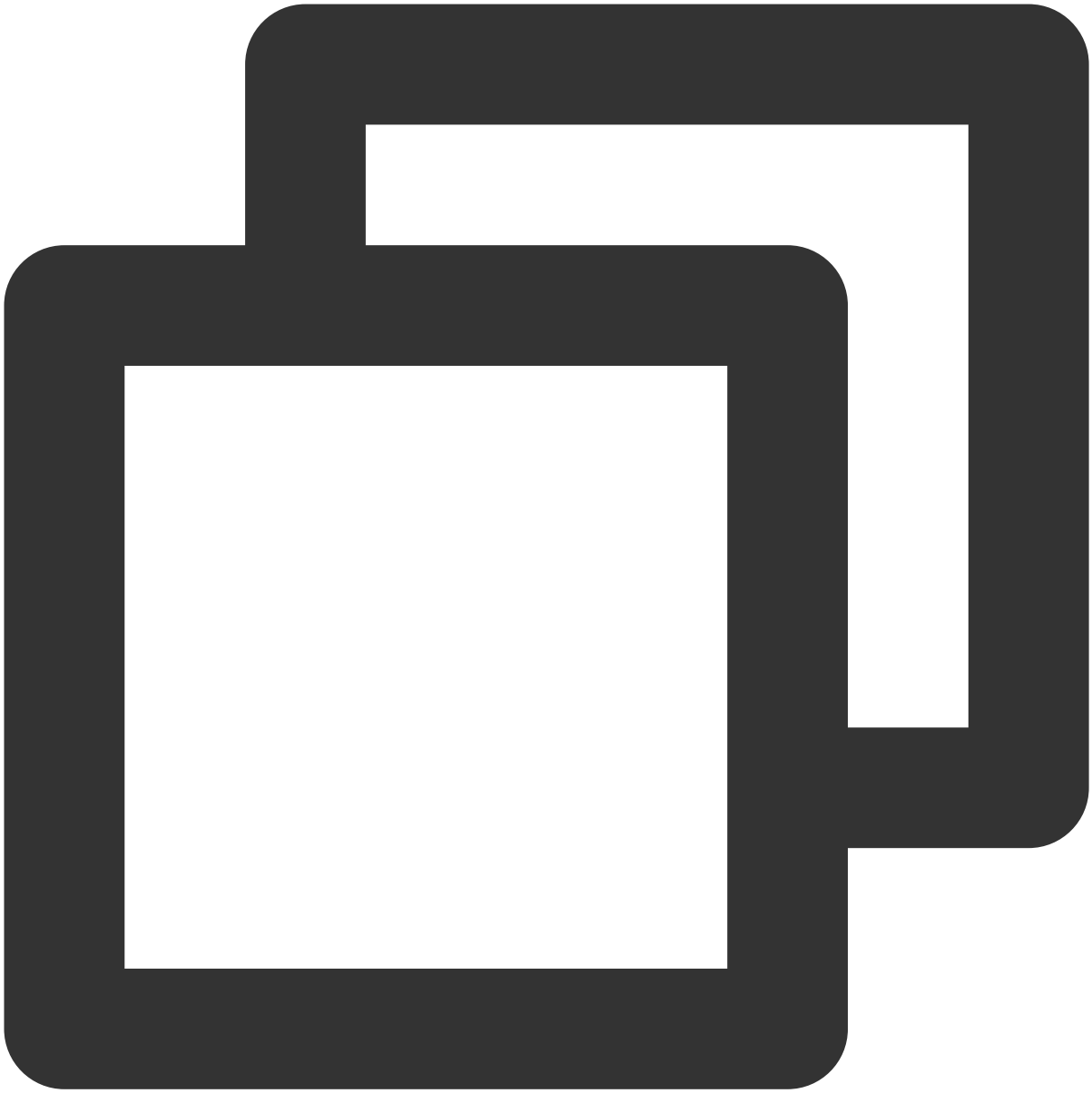
C++



```
//RealTimeVoiceActivity.java
ITMGContext.GetInstance(this).GetAudioCtrl().EnableMic(true);
```



```
//TMGRealTimeViewController.m
[[[ITMGContext GetInstance] GetAudioCtrl] EnableMic:YES];
```



```
ITMGContextGetInstance()->GetAudioCtrl()->EnableMic(true);
```

3. スピーカーのオン/オフ

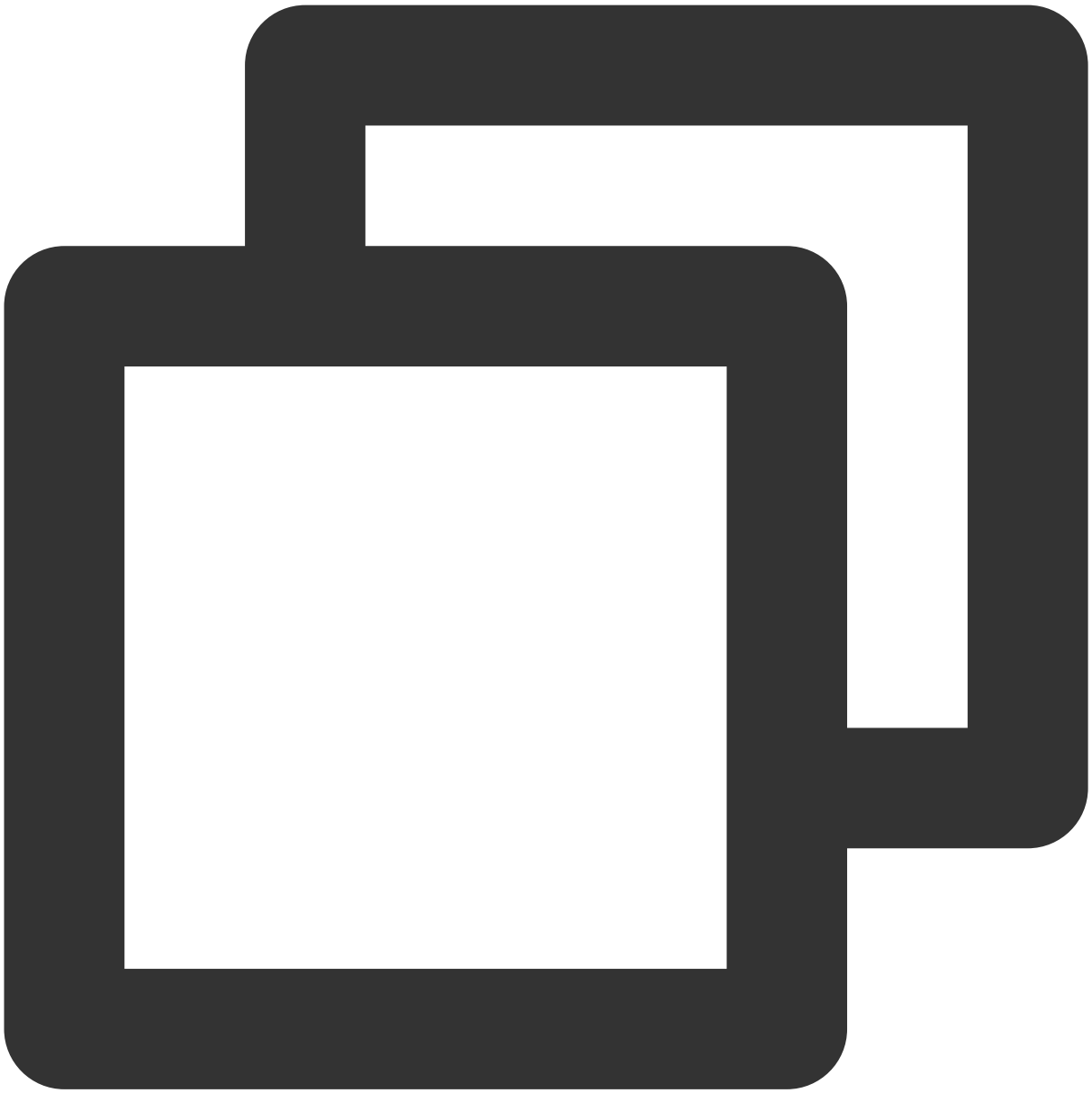
このインターフェースは、スピーカーのオン/オフに使用されます。

サンプルコード

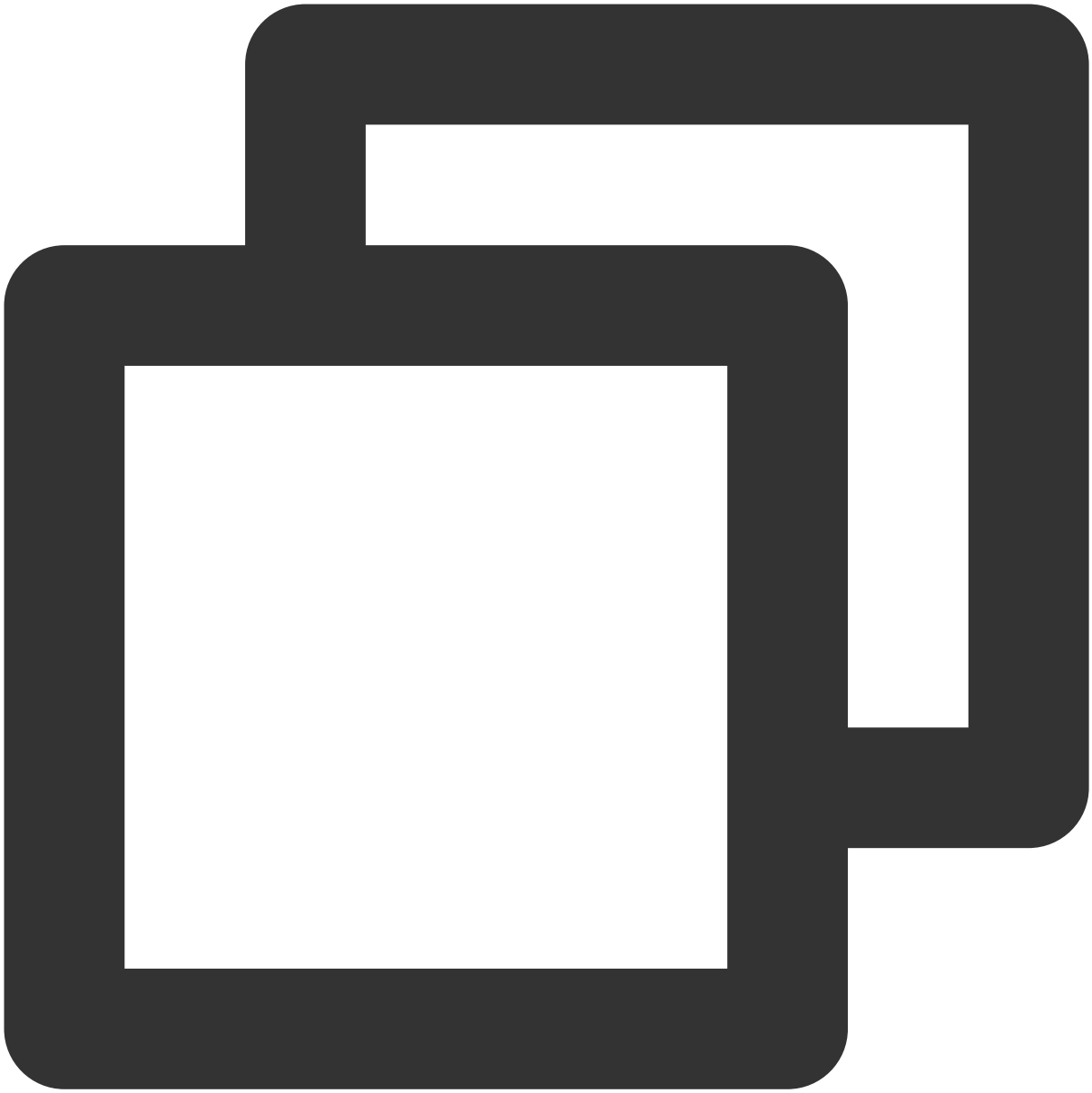
Java

Object-C

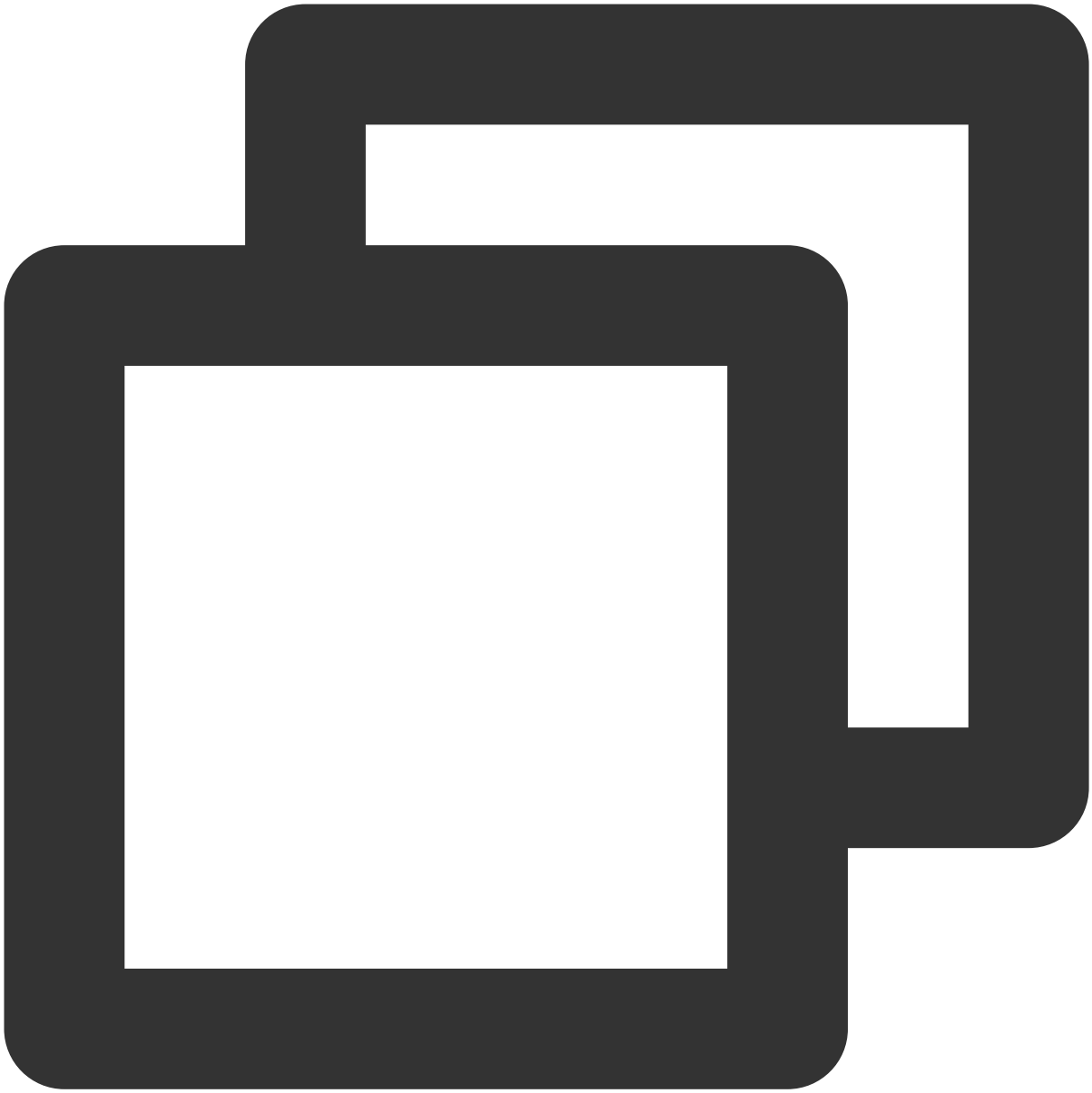
C++



```
//RealTimeVoiceActivity.java
ITMGContext.GetInstance(this).GetAudioCtrl().EnableSpeaker(true);
```



```
//TMGRealTimeViewController.m
[[[ITMGContext GetInstance] GetAudioCtrl] EnableSpeaker:YES];
```



```
ITMGContextGetInstance()->GetAudioCtrl()->EnableSpeaker(true);
```

4. 退室

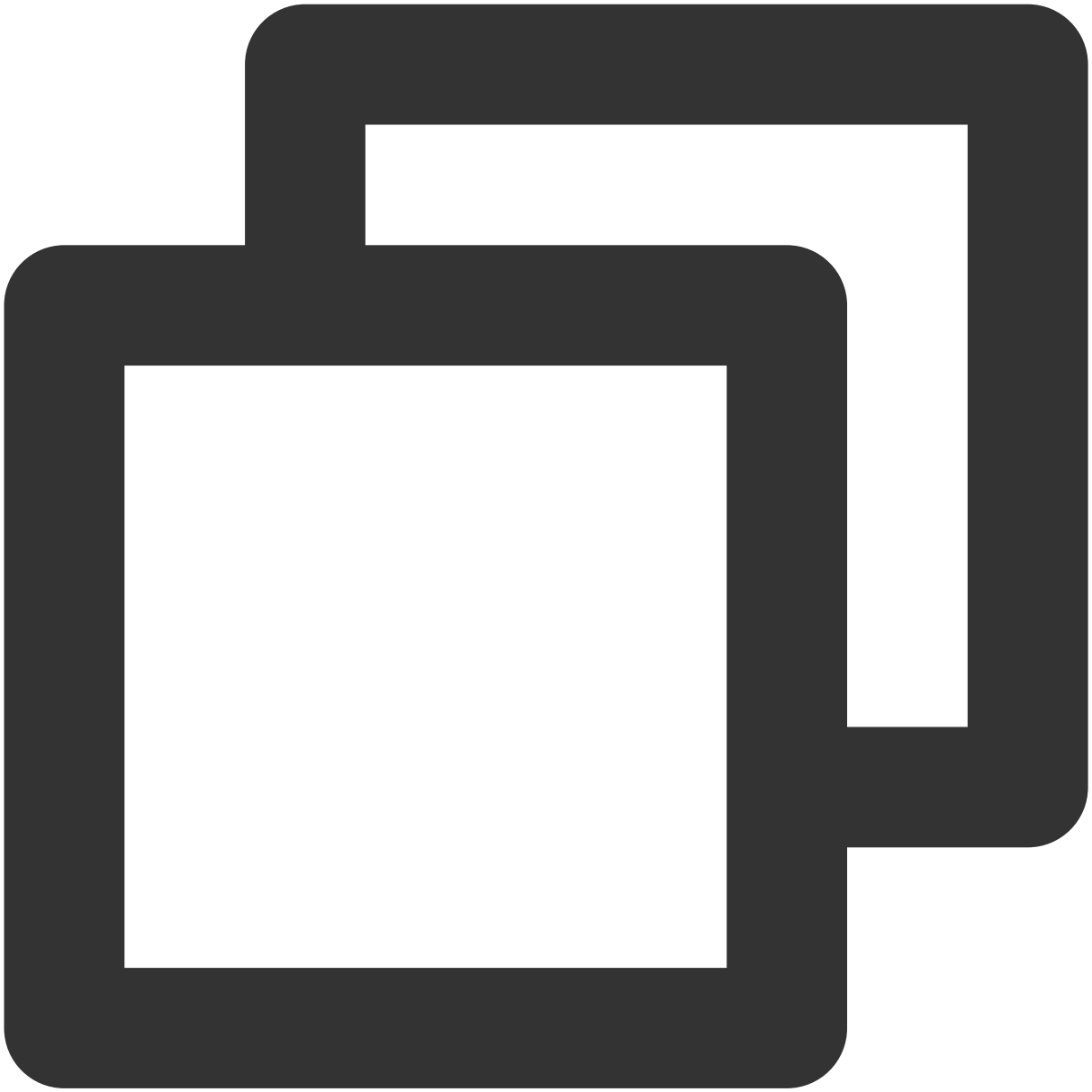
このインターフェースを呼び出すと、退室することができます。処理の実行は退室のコールバックを待つ必要があります。

サンプルコード

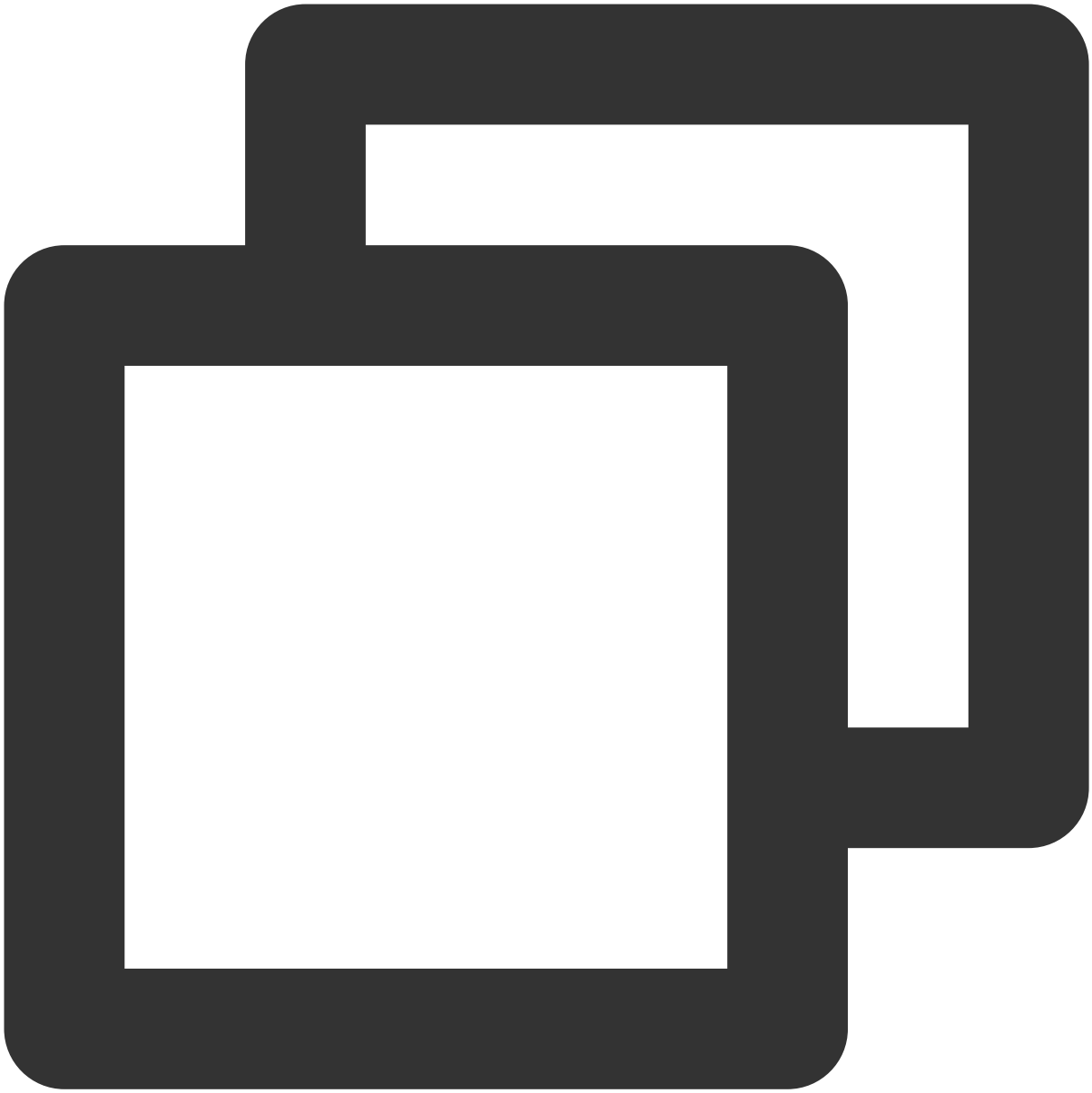
Java

Object-C

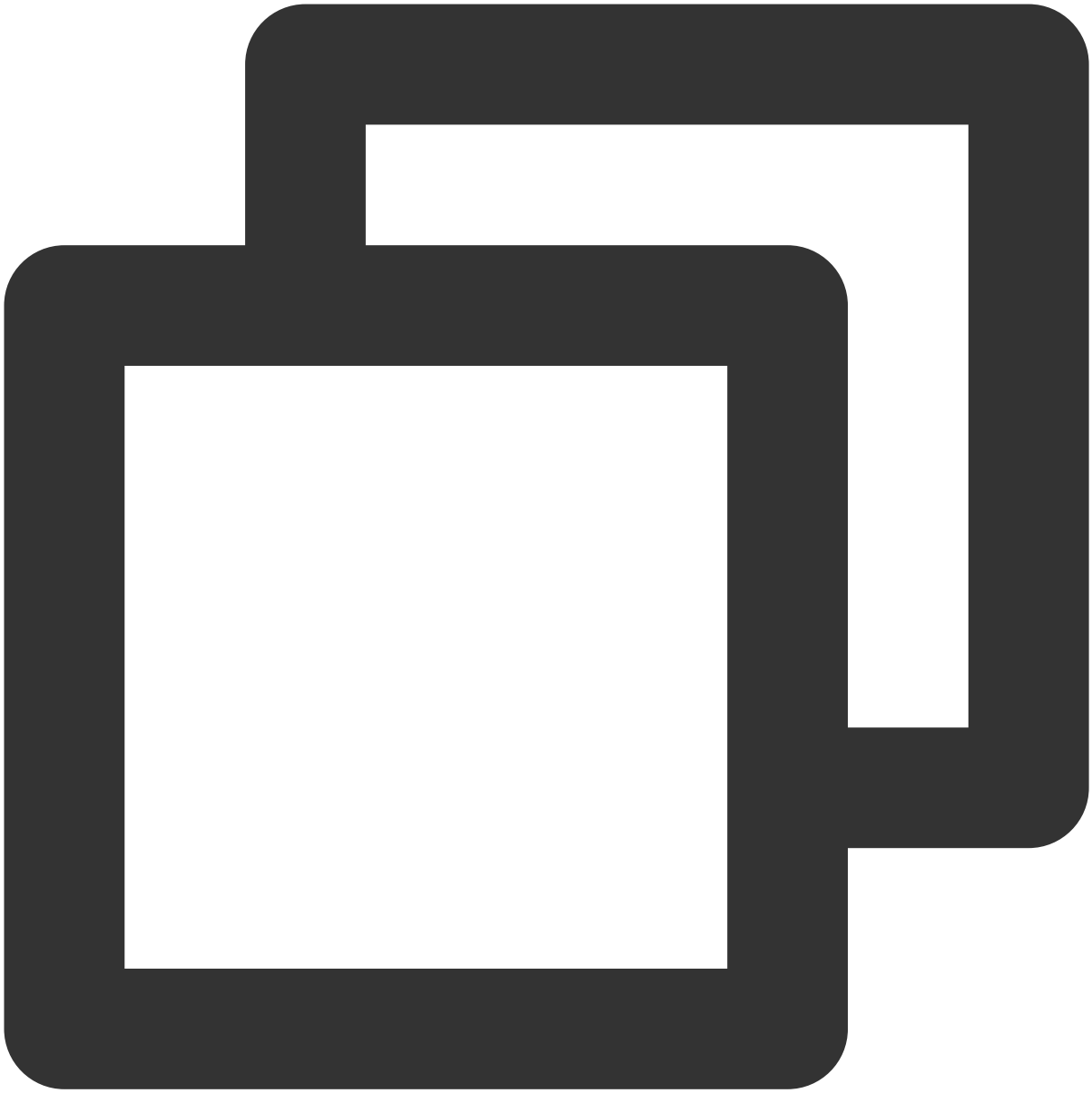
C++



```
//RealTimeVoiceActivity.java
ITMGContext.GetInstance().ExitRoom();
```



```
//TMGRealTimeViewController.m
[[ITMGContext GetInstance] ExitRoom];
```



```
ITMGContext* context = ITMGContextGetInstance();
context->ExitRoom();
```

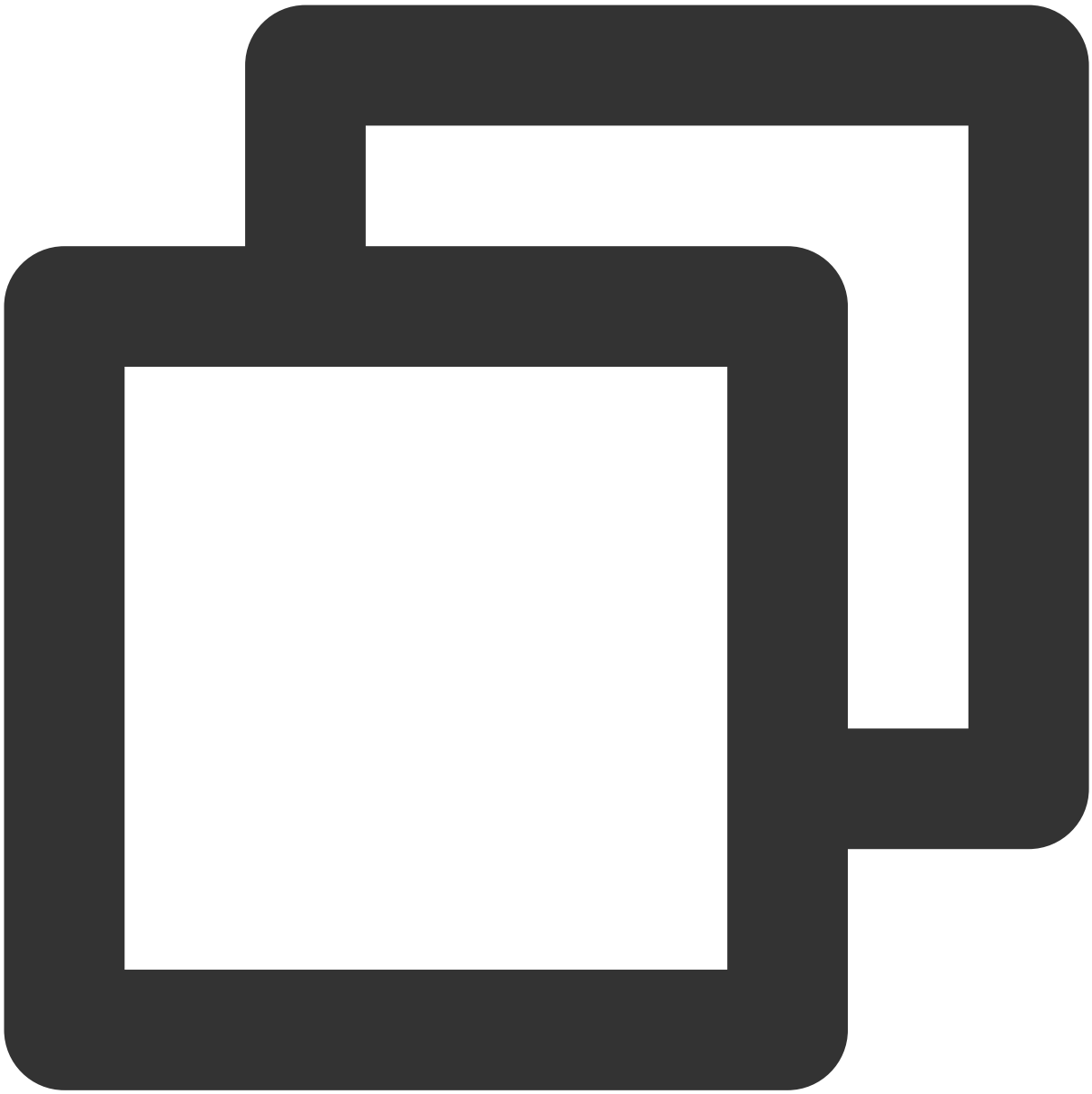
退室コールバック

退室してからはコールバックが発生します、メッセージはITMG_MAIN_EVENT_TYPE_EXIT_ROOMです。サンプルコードは次の通りです：

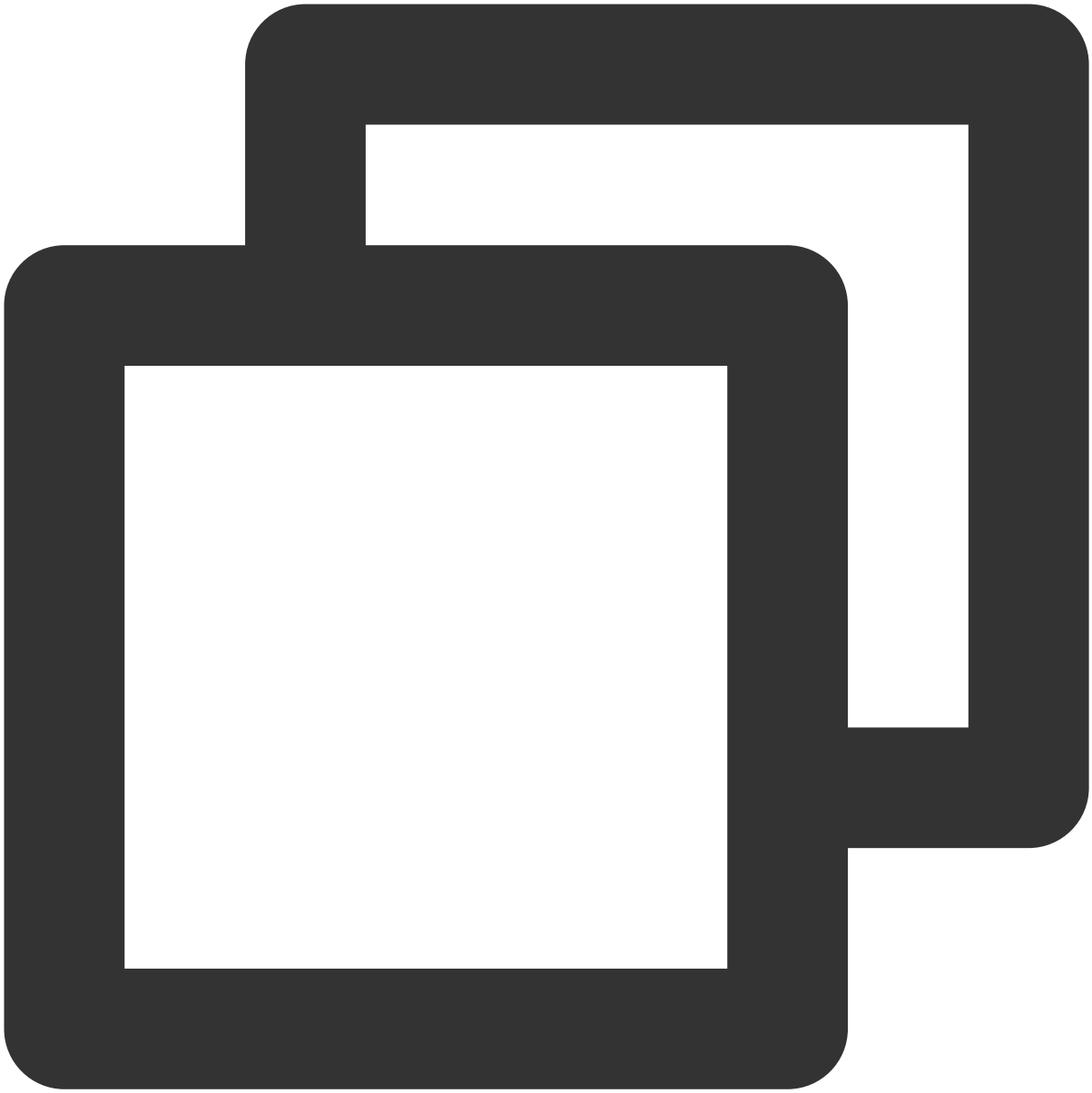
Java

Object-C

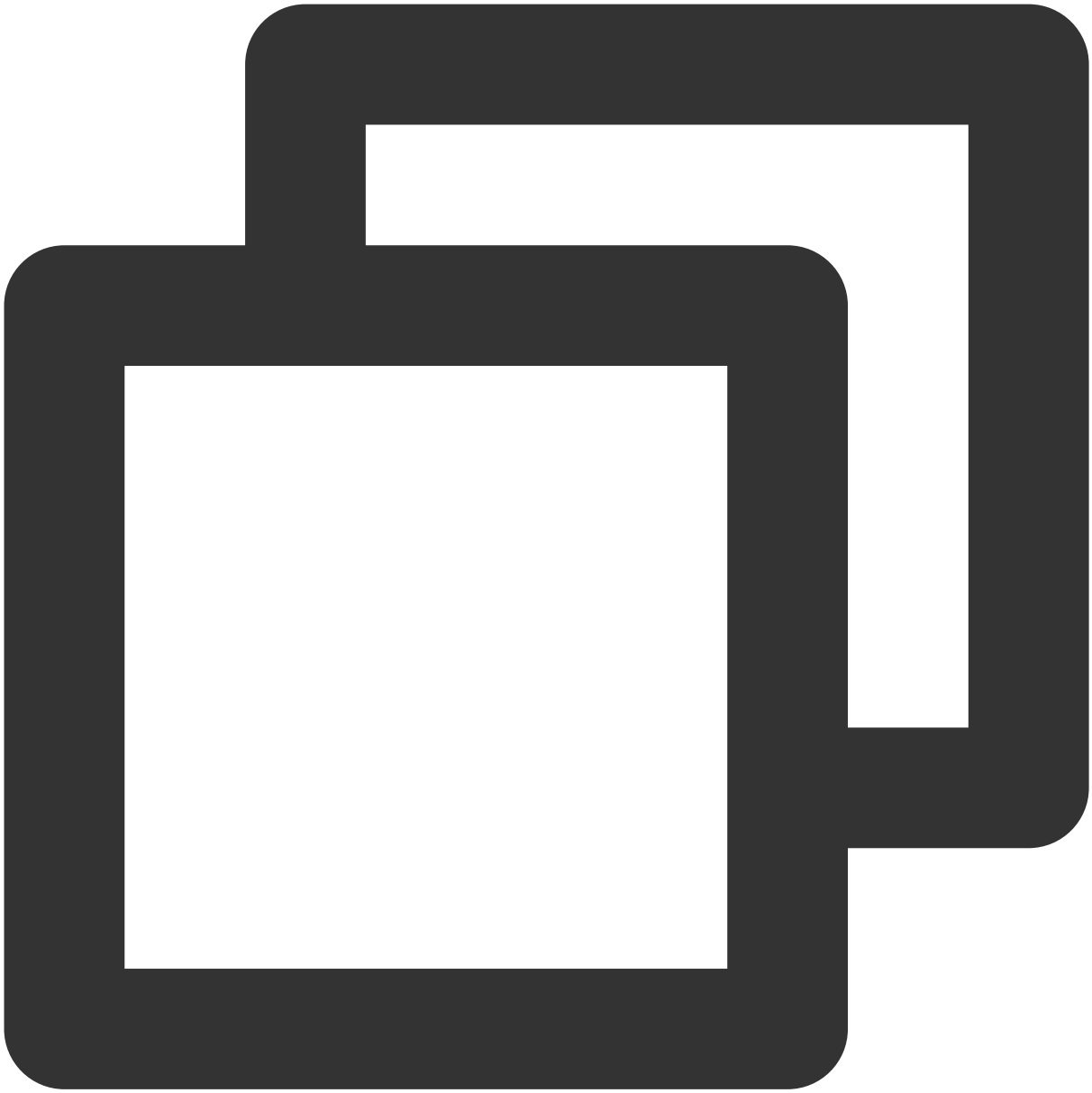
C++



```
//RealTimeVoiceActivity.java
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_EXIT_ROOM == type)
        {
            //退室成功イベントの受信
        }
}
```



```
//TMGRealTimeViewController.m
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
switch (eventType) {
    case ITMG_MAIN_EVENT_TYPE_EXIT_ROOM：
    {
        //退室成功イベントの受信
    }
        break;
    }
}
```



```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
switch (eventType) {
    case ITMG_MAIN_EVENT_TYPE_EXIT_ROOM:
    {
        //処理します
        break;
        }
    }
}
```

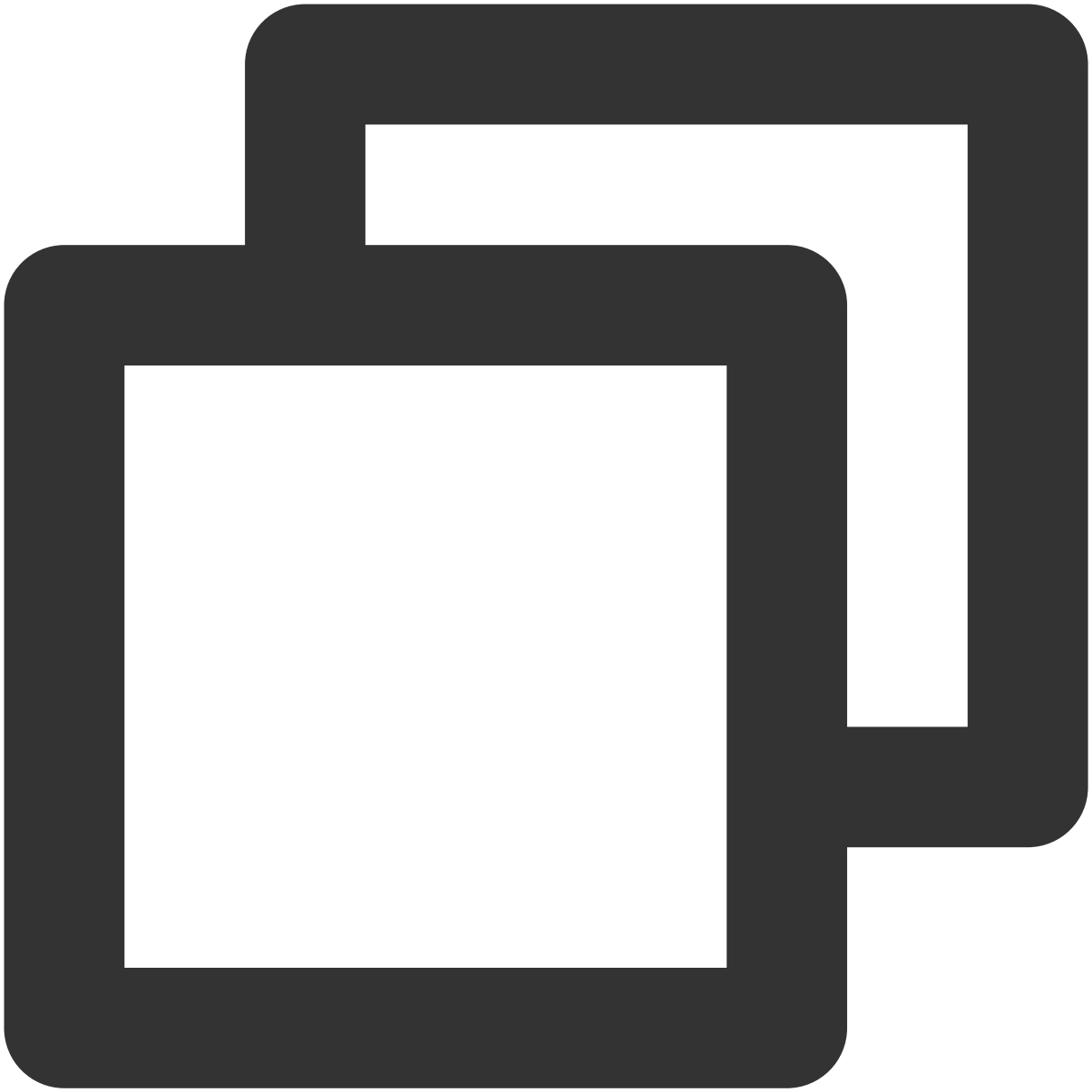
音声メッセージの導入

1. 認証初期化

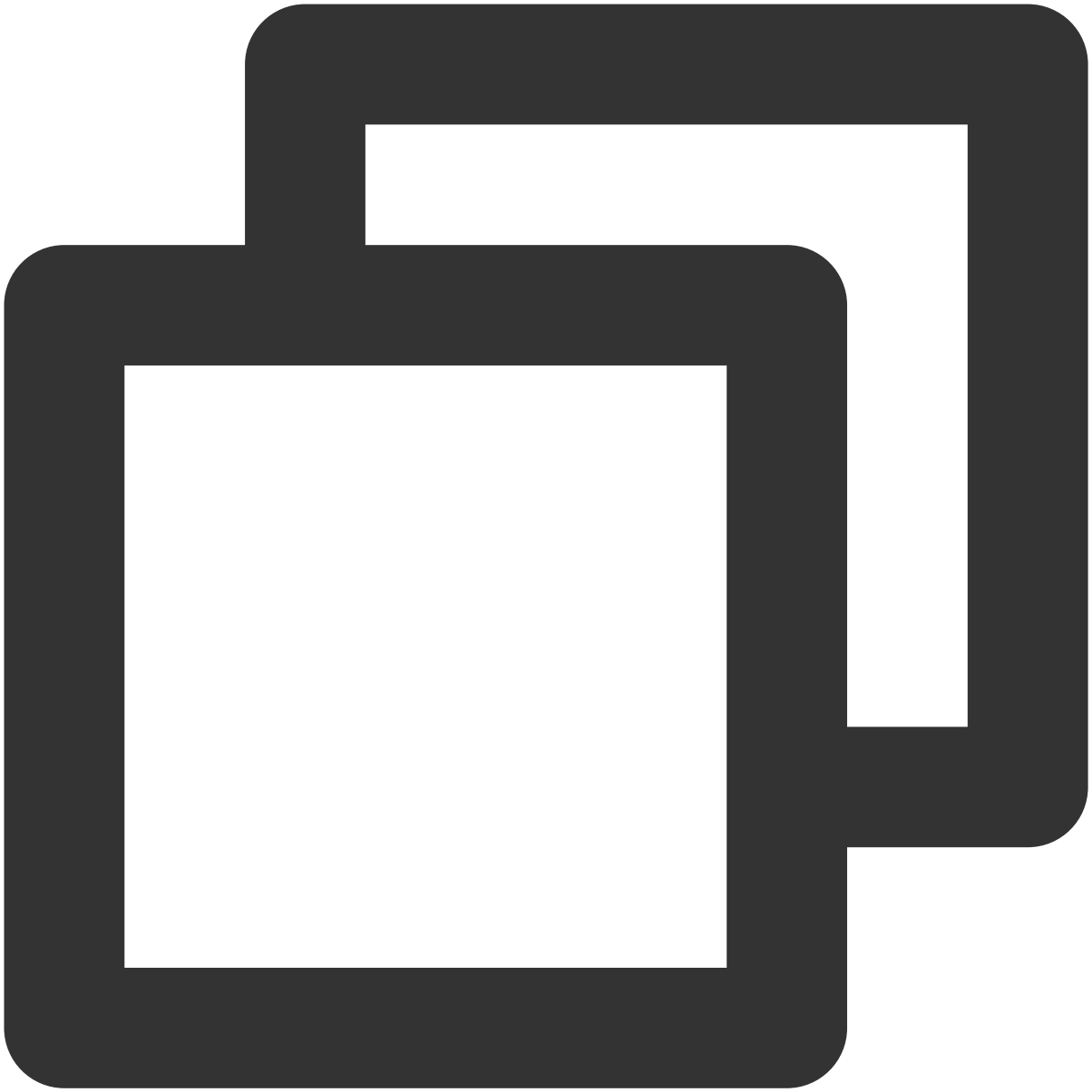
SDKを初期化してから認証の初期化を呼び出します。authBufferの取得については、前記のリアルタイム音声の認証情報インターフェースgenAuthBufferをご参照ください。

インターフェースのプロトタイプ

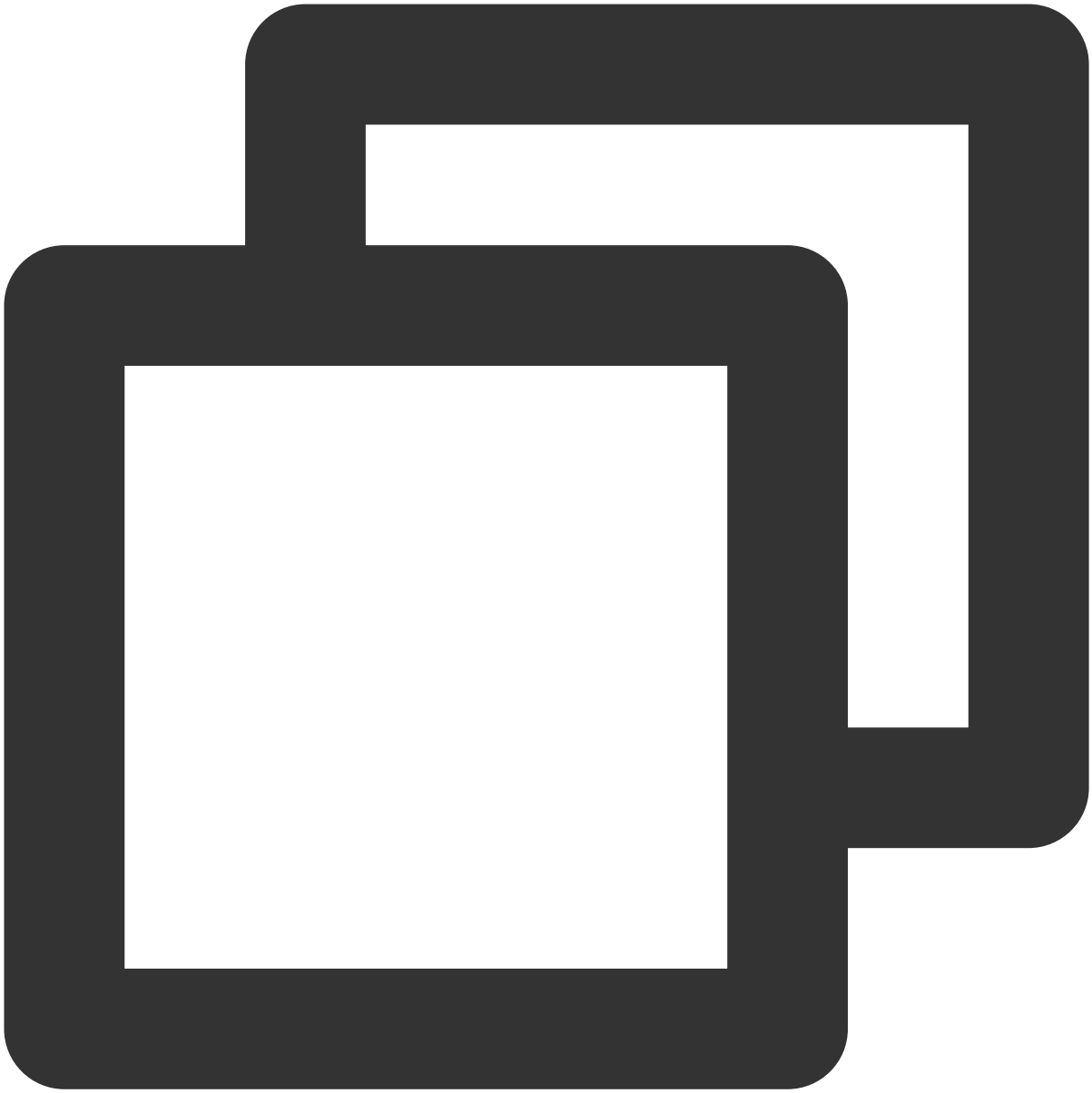
Java
Object-C
C++



```
public abstract int ApplyPTTAuthbuffer(byte[] authBuffer);
```



```
-(QAVResult)ApplyPTTAuthbuffer:(NSData *)authBuffer;
```



```
ITMGPTT virtual int ApplyPTTAuthbuffer(const char* authBuffer, int authBufferLen)
```

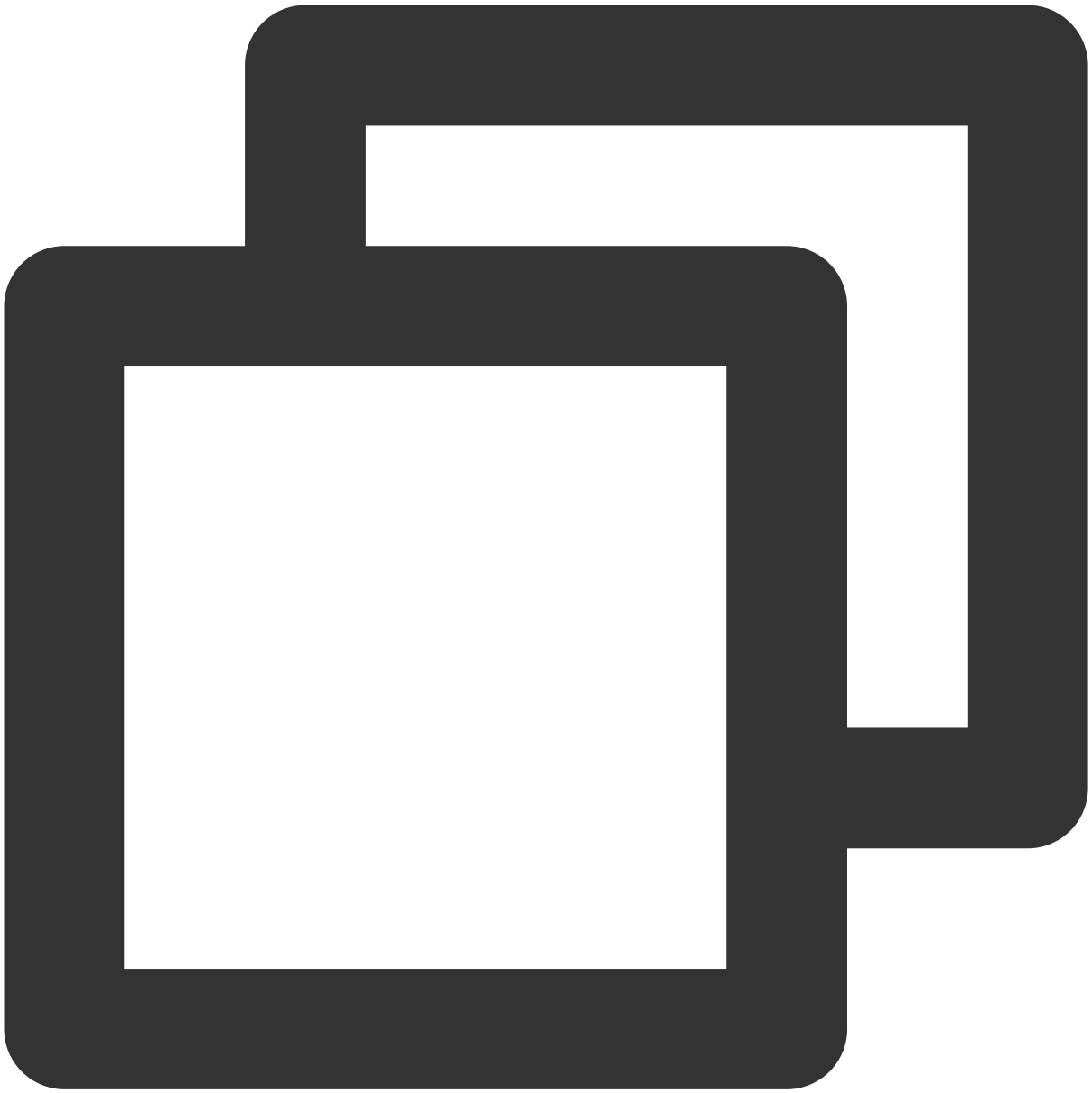
パラメータ	タイプ	意味
authBuffer	String	認証

サンプルコード

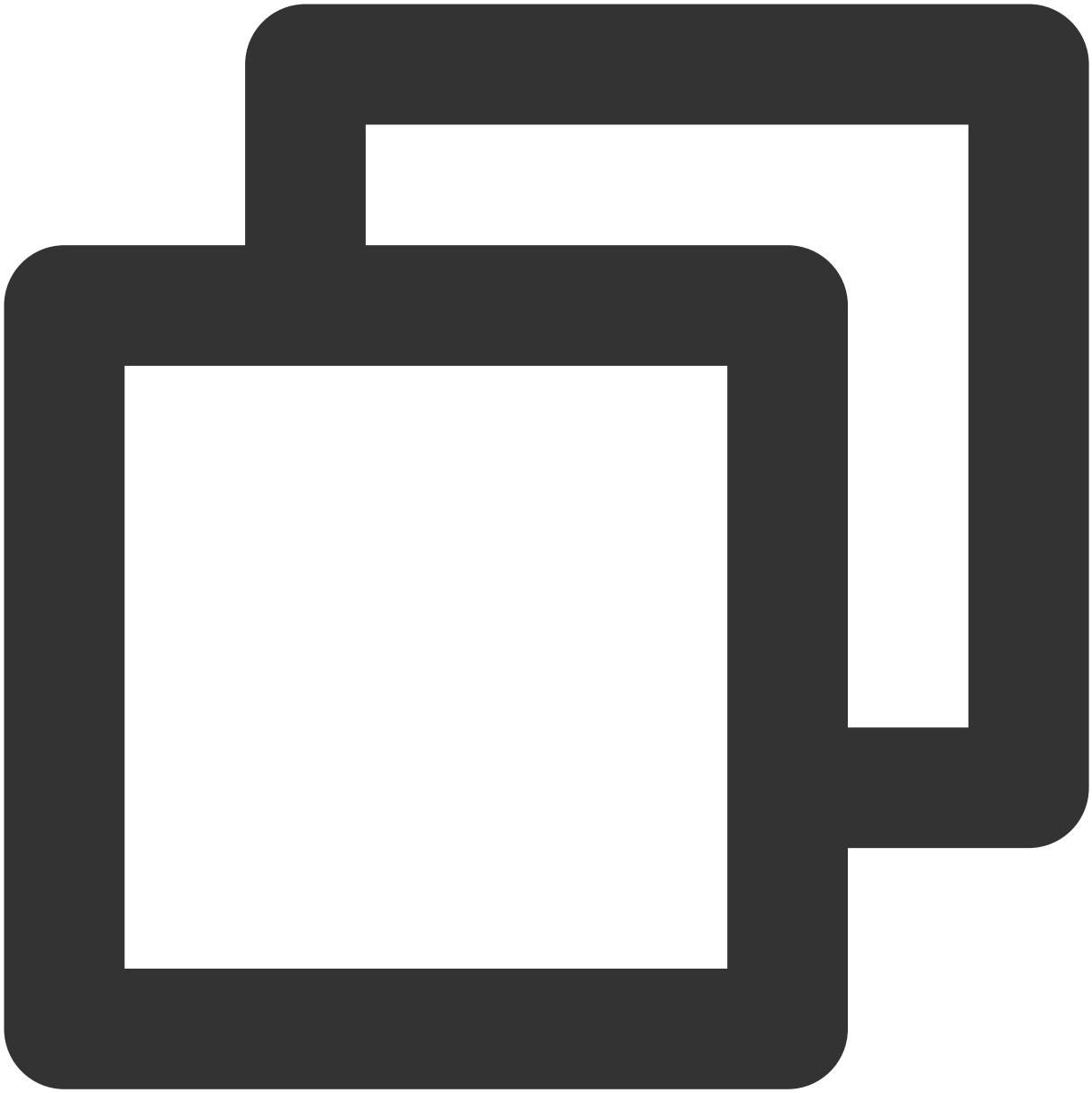
Java

Object-C

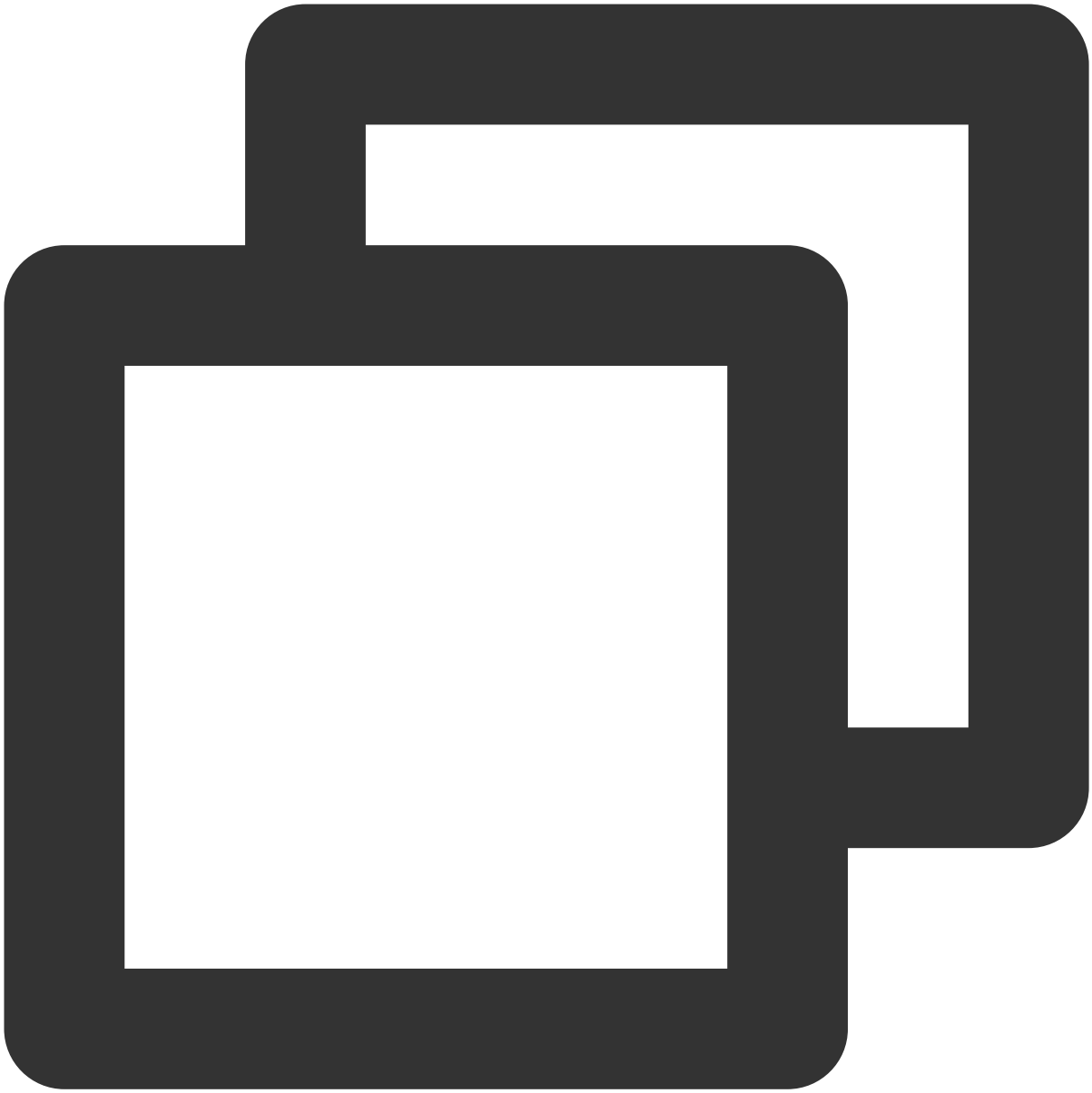
C++



```
//VoiceMessageRecognitionActivity.java
byte[] authBuffer = GMEAuthBufferHelper.getInstance().createAuthBuffer("");
ITMGContext.GetInstance(this).GetPTT().ApplyPTTAuthbuffer(authBuffer);
```



```
//TMGPTTViewController.m
NSData* authBuffer =  [QAVAuthBuffer GenAuthBuffer:(unsigned int)SDKAPPID3RD.intege
[[[ITMGContext GetInstance] GetPTT] ApplyPTTAuthbuffer:authBuffer];
```



```
ITMGContextGetInstance()->GetPTT()->ApplyPTTAuthbuffer(authBuffer,authBufferLen);
```

2. ストリーミング音声認識を起動

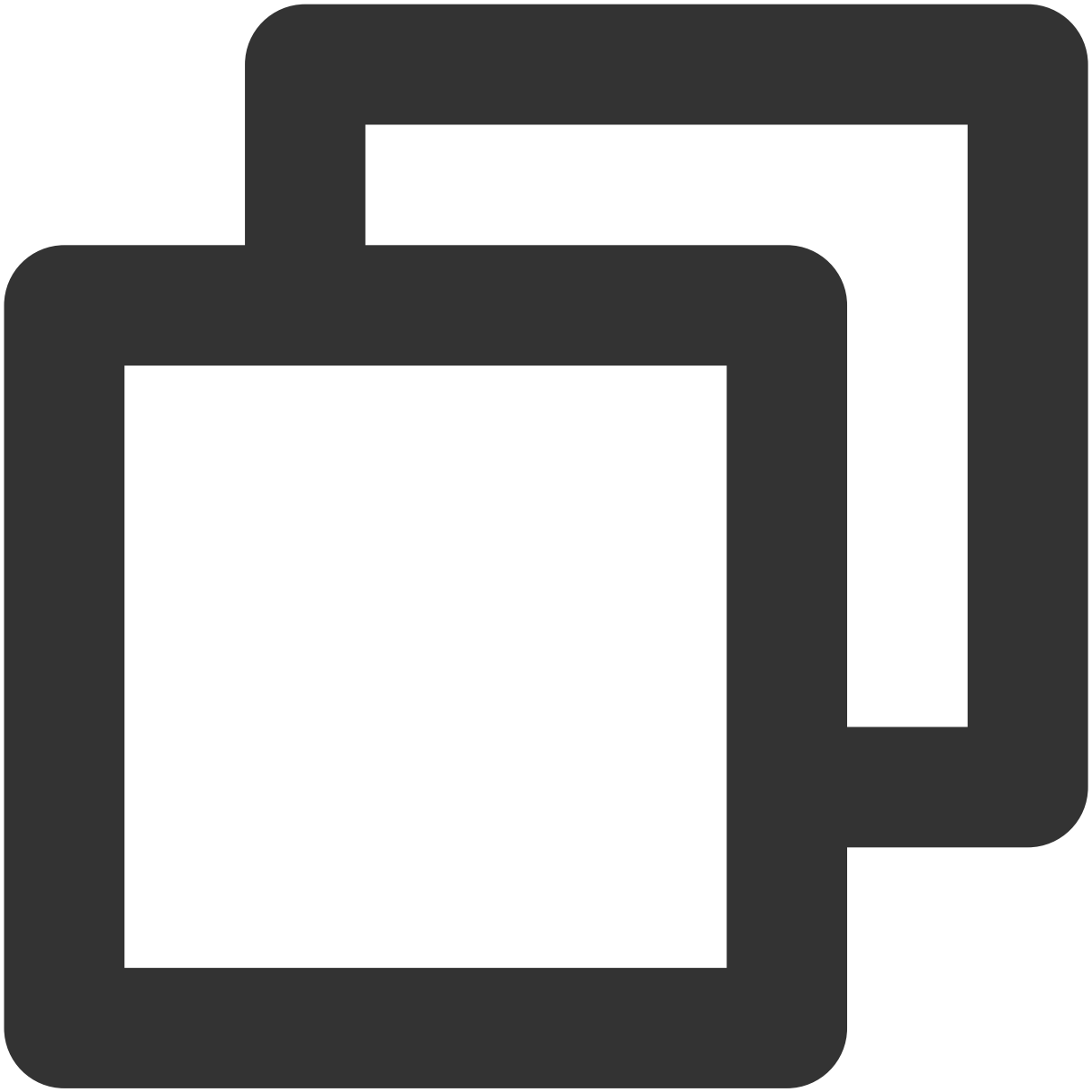
このインターフェースは、ストリーミング音声識別の開始に使われています。コールバックにおいて、音声はリアルタイムでテキストに変換されて返されます。**録音の停止にはStopRecordingを呼び出します**。停止後にコールバックが発生します。

インターフェースのプロトタイプ

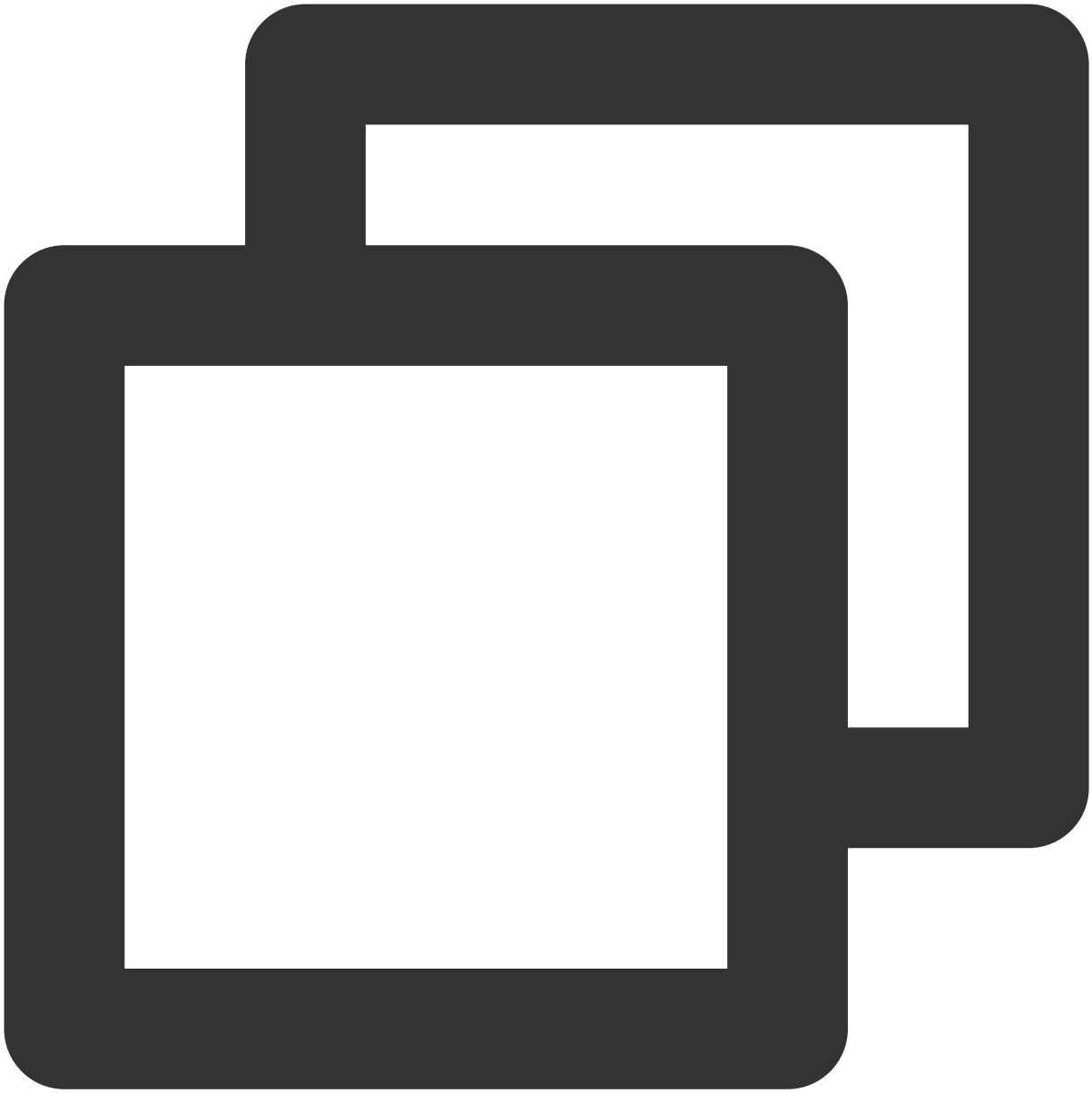
Java

Object-C

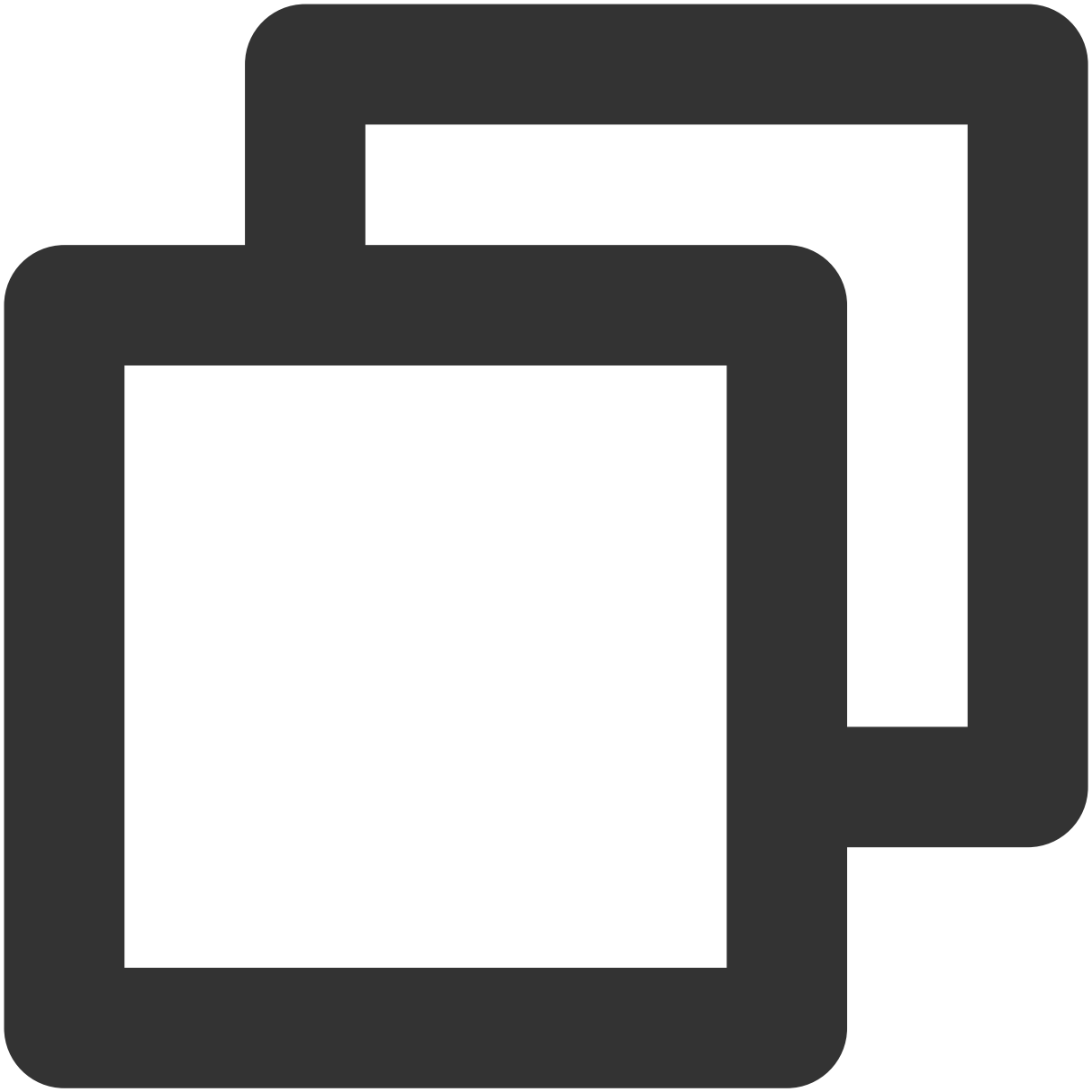
C++



```
public abstract int StartRecordingWithStreamingRecognition (String filePath);

public abstract int StopRecording();
```



```
-(int)StartRecordingWithStreamingRecognition:(NSString *)filePath;

-(QAVResult)StopRecording;
```

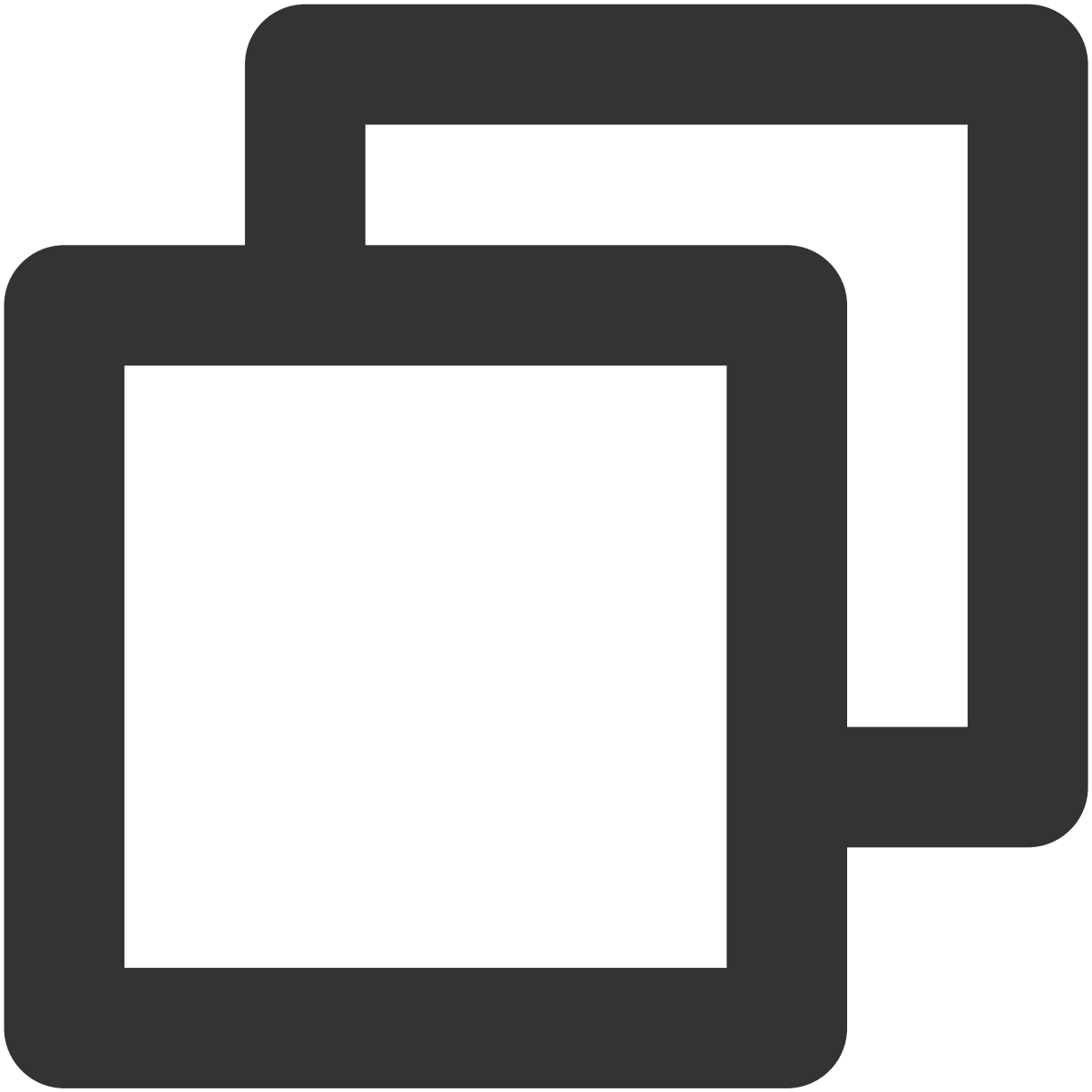


```
ITMGPTT virtual int StartRecordingWithStreamingRecognition(const char* filePath)

ITMGPTT virtual int StopRecording()
```

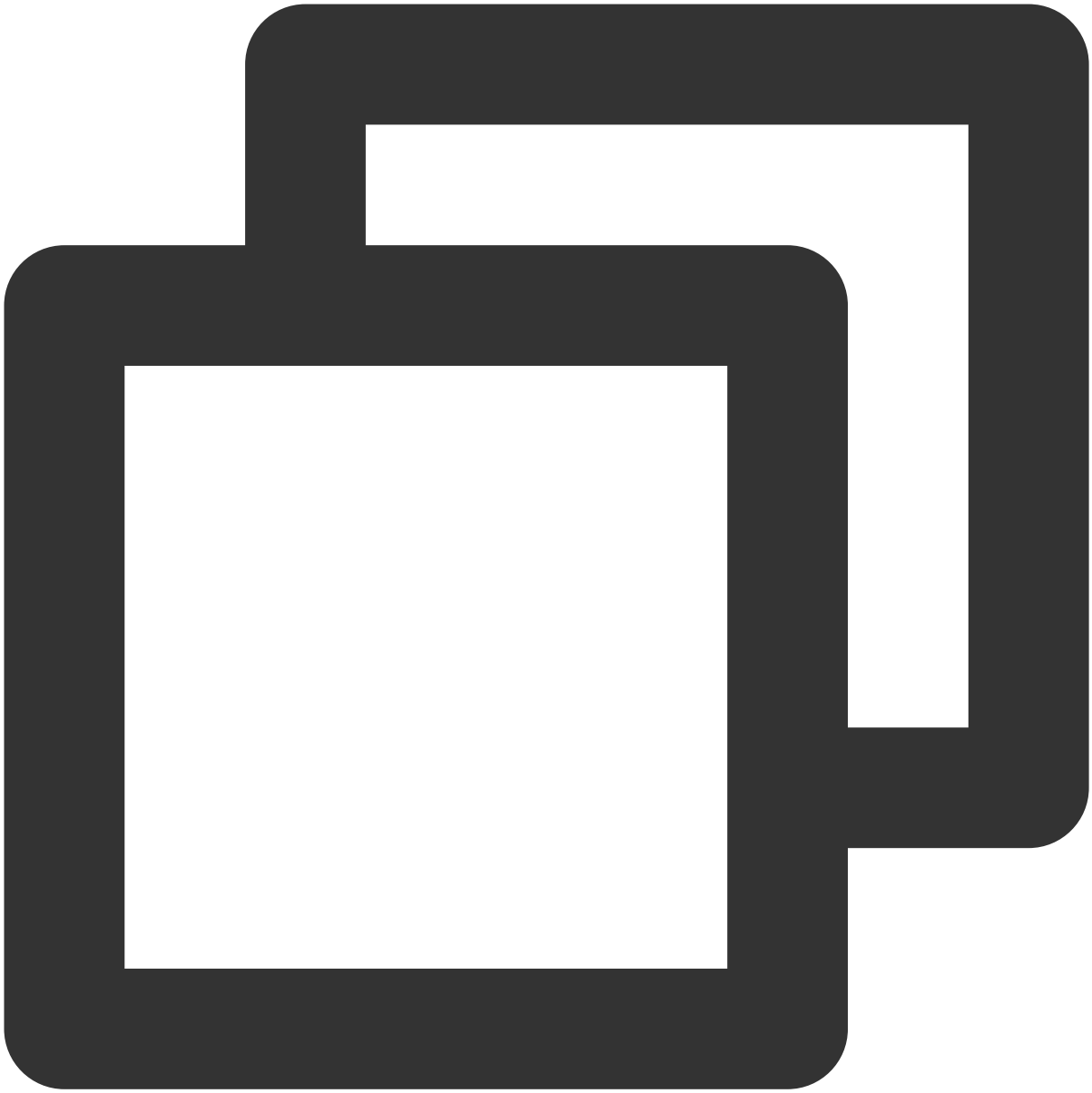
パラメータ	タイプ	意味
filePath	String	ボイスの保存パス

サンプルコード

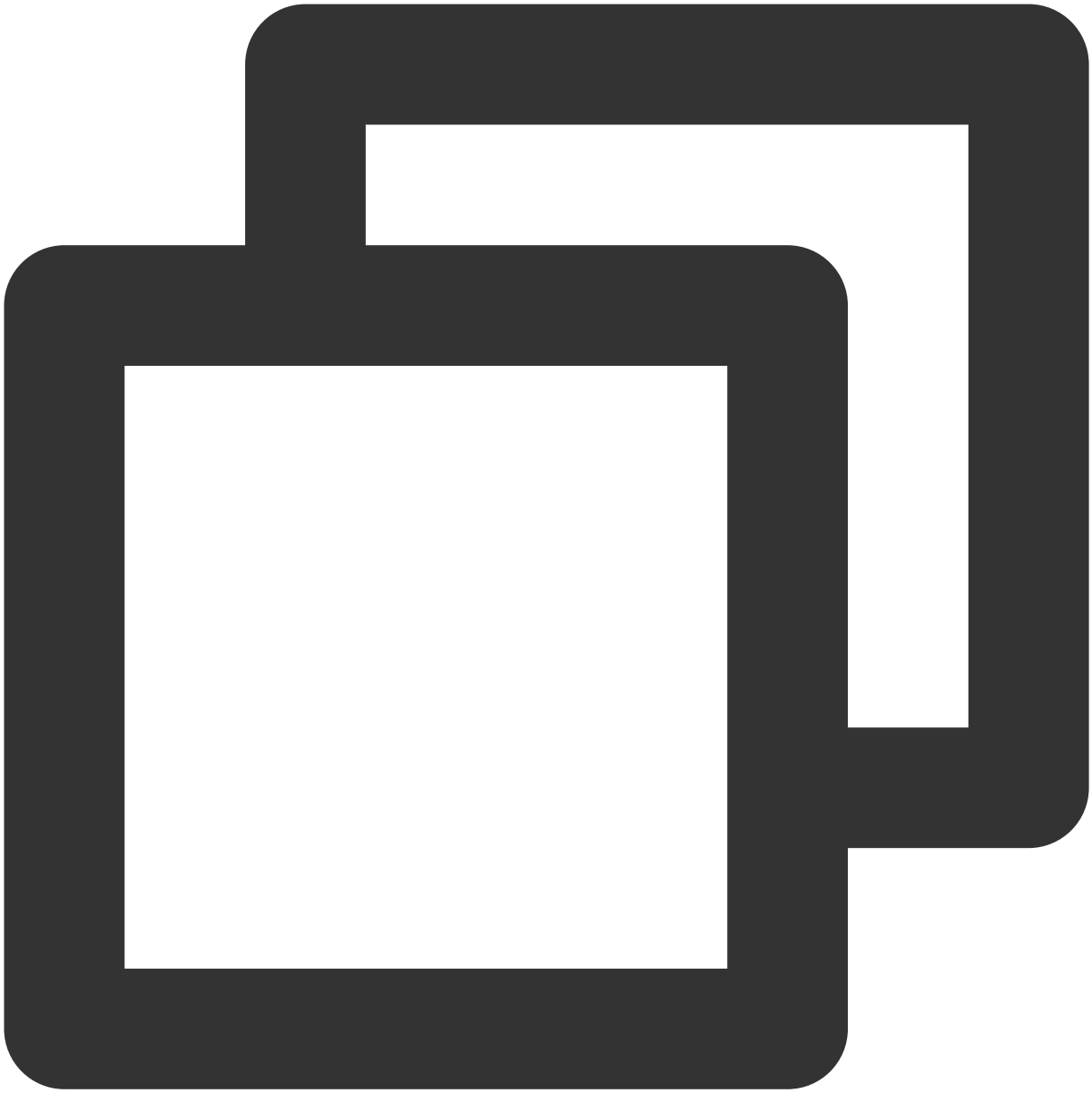
Java
Object-C
C++



```
//VoiceMessageRecognitionActivity.java
ITMGContext.GetInstance(this).GetPTT().StartRecordingWithStreamingRecognition(recor
```



```
//TMGPTTViewController.m
QAVResult ret = [[[ITMGContext GetInstance] GetPTT] StartRecordingWithStreamingReco
if (ret == 0) {
    self.currentStatus = @"ストリーミング録音を開始します";
}else{
    self.currentStatus = @"ストリーミング録音の開始に失敗しました";
}
```



```
ITMGContextGetInstance()->GetPTT()->StartRecordingWithStreamingRecognition(filePath
```

ストリーミング音声識別コールバック

ストリーミング音声認識を開始した後、コールバック関数OnEventでコールバックメッセージを受信する必要があります。イベントメッセージは `ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` があり、録音を停止して認識を完了した後にテキストを返します。これは、話が終わってから認識されたテキストを返すことに相当します。

OnEvent関数で、必要に応じて適切なイベントメッセージを判断します。渡されるパラメータには次の4つの情報が含まれます。

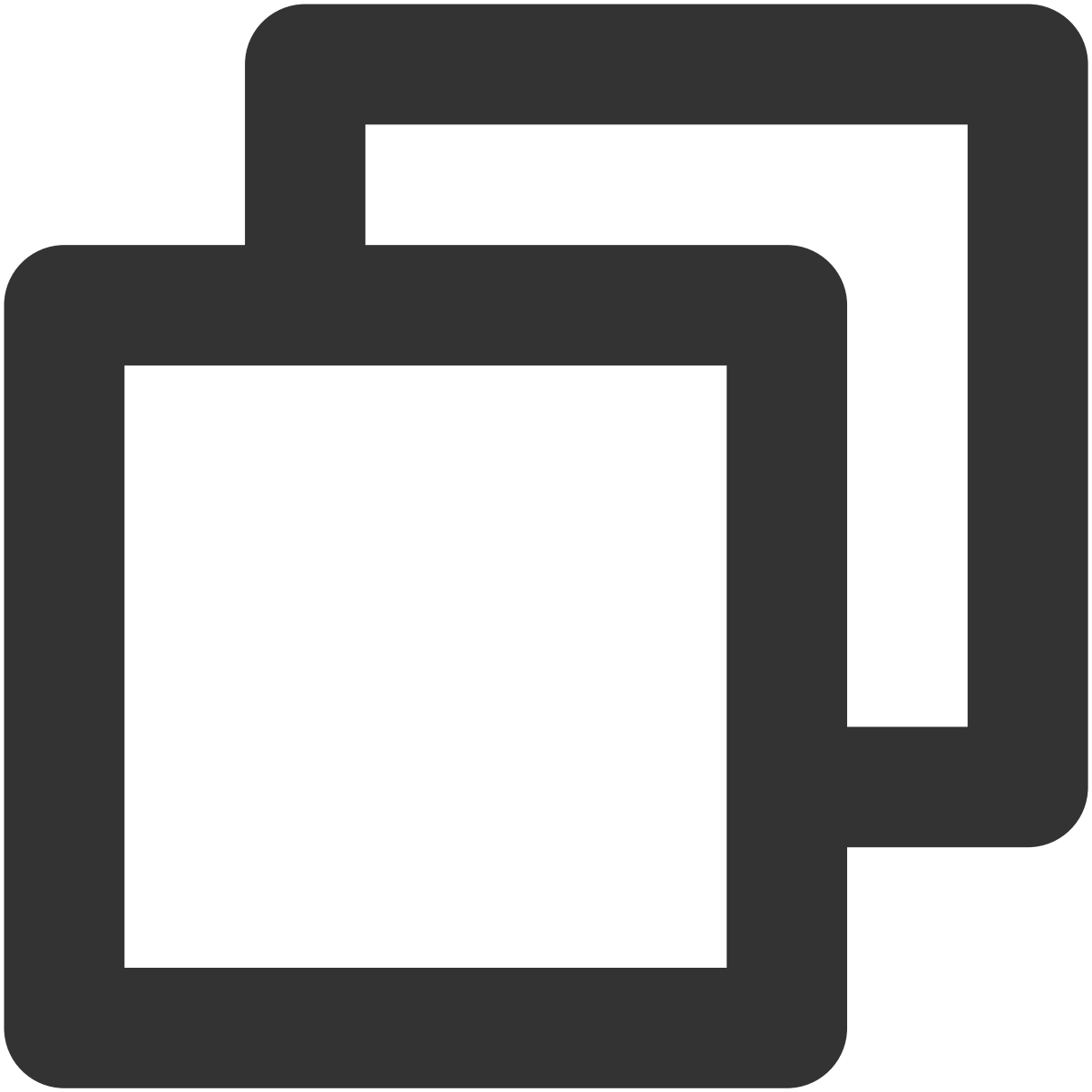
メッセージ名称	意味
result	ストリーミングボイス認識が完了したかどうかを判断するための戻りコード
text	ボイステキスト変換で認識されたテキスト
file_path	録音を保存するローカルアドレス
file_id	録音はバックグラウンドのURLアドレスにあり、録音はサーバーで90日間保存されます

サンプルコード

Java

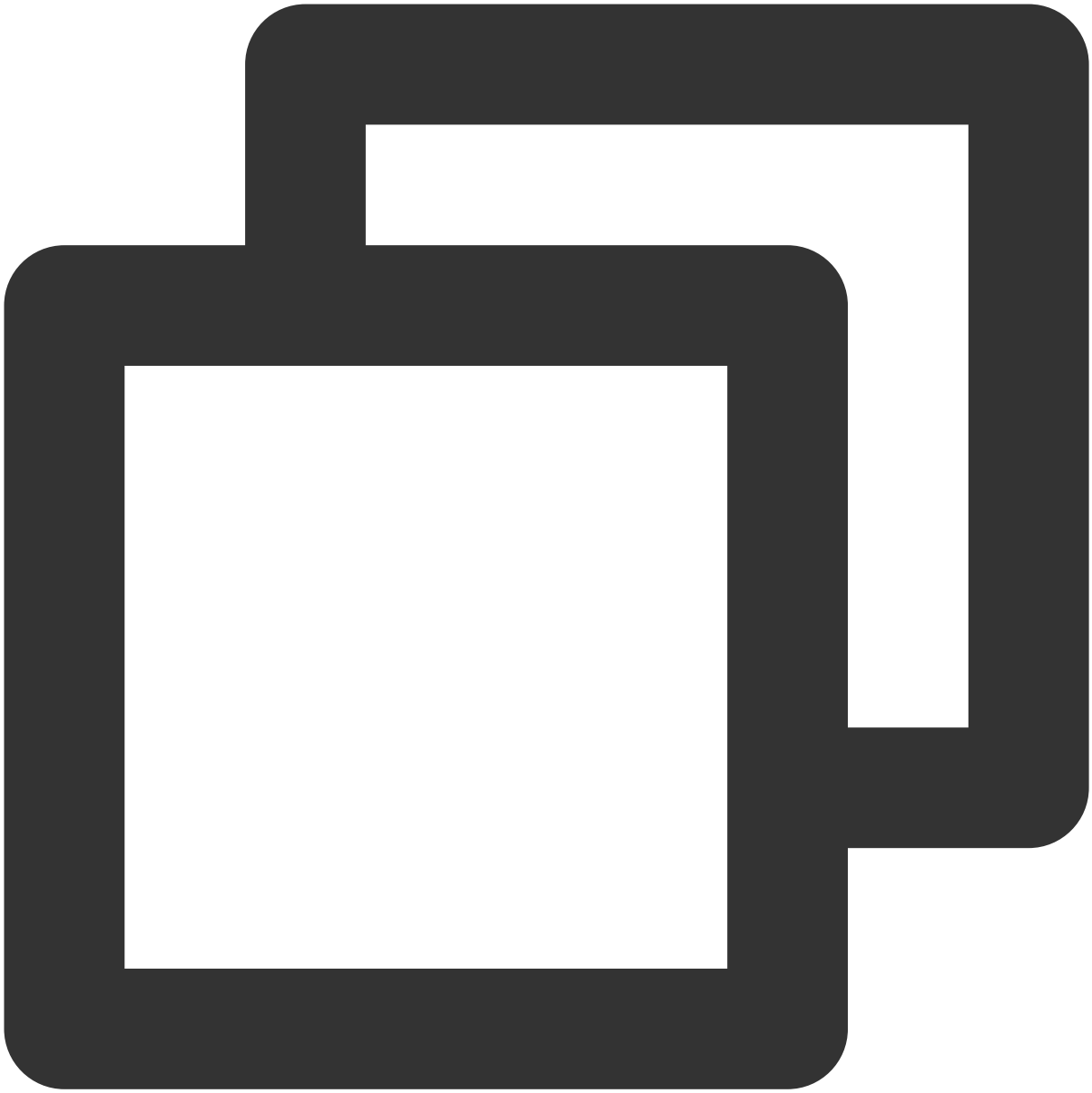
Object-C

C++



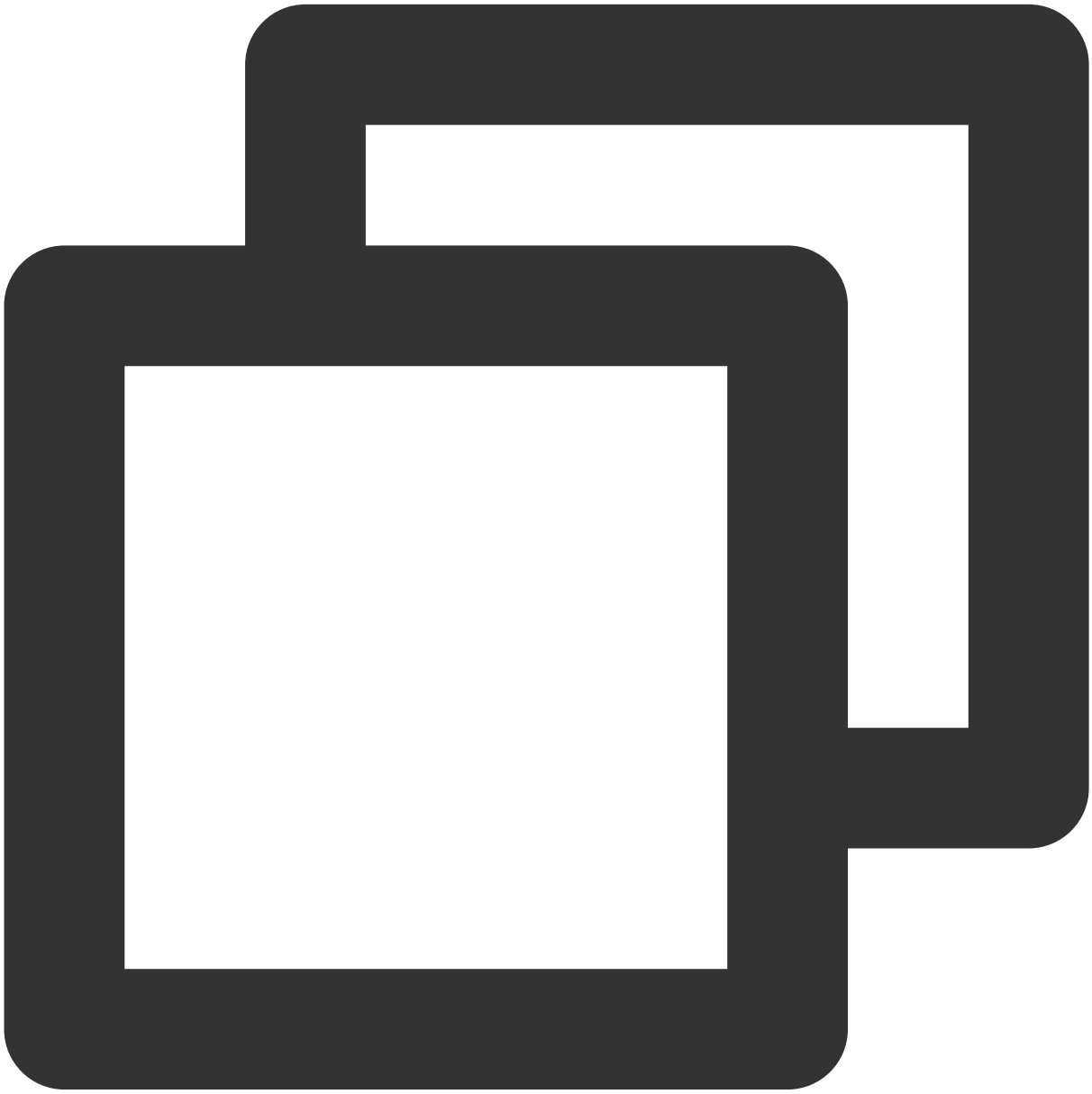
```
//VoiceMessageRecognitionActivity.java
import static com.tencent.TMG.ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if (type == ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE)
        {
            // Step 1.3/3 handle ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMP
            mIsRecording = false;
            if (nErrCode ==0)
            {
                String recordfilePath = data.getStringExtra("file_path");
                mRecFilePathView.setText(recordfilePath);
```

```
                String recordFileUrl = data.getStringExtra("file_id");
                mRecFileUrlView.setText(recordFileUrl);
            }
            else
            {
                appendLog2MonitorView("Record and recognition fail errCode:" + nErr
            }
        }

}
```



```
//TMGPTTViewController.m

- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary*)data
  {
  NSNumber *number = [data objectForKey:@"result"];
  switch(eventType)
  {
      case ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE:
      {
            if (data != NULL &&[[data objectForKey:@"result"] intValue]== 0)
            {
```

```
                self.translateTF.text = [data objectForKey:@"text"] ;
                 self.currentStatus = @"ストリーム変換が完了しました";
            }
        }
    break;
}
```



```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
  switch (eventType) {
      case ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE:
```

```
        {
            HandleSTREAM2TEXTComplete(data,true);
            break;
        }
        ...
        case ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING:
        {
            HandleSTREAM2TEXTComplete(data, false);
            break;
        }
    }
}
void CTMGSDK_For_AudioDlg::HandleSTREAM2TEXTComplete(const char* data, bool isCom
{
    std::string strText = "STREAM2TEXT: ret=";
    strText += data;
    m_EditMonitor.SetWindowText(MByteToWChar(strText).c_str());
    Json::Reader reader;
    Json::Value root;
    bool parseRet = reader.parse(data, root);
    if (!parseRet) {
        ::SetWindowText(m_EditInfo.GetSafeHwnd(),MByteToWChar(std::string("parse re
    }
        else
        {
            if (isComplete) {
                                ::SetWindowText(m_EditUpload.GetSafeHwnd(), MByteTo
                            }
                            else{
                                    std::string isruning = "STREAMINGRECOGNITION_IS
                                    ::SetWindowText(m_EditUpload.GetSafeHwnd(), MBy
                                 }
    }
}
```

エラーコード

エラーコード	意味	処理方式
32775	ストリーミング音声をテキストに変更できませんが、録音は成功しました	UploadRecordedFileインターフェースを呼び出して録音をアップロードし、SpeechToTextインターフェースを呼び出して音声を文字に変換します
32777	ストリーミング音声をテキストに変更できませんが、	返されたメッセージには正常にアップロードしたバックグラウンドURLがあり、SpeechToTextインターフェースを呼び

	録音とアップロードは成功しました	出して音声から文字への変換操作を行います
32786	ストリーミング音声をテキストに変更できませんでした	ストリーミングレコーディングステータスでは、ストリーミングレコーディングインターフェースの実行結果が返されるまで待ってください

3. 録音を停止

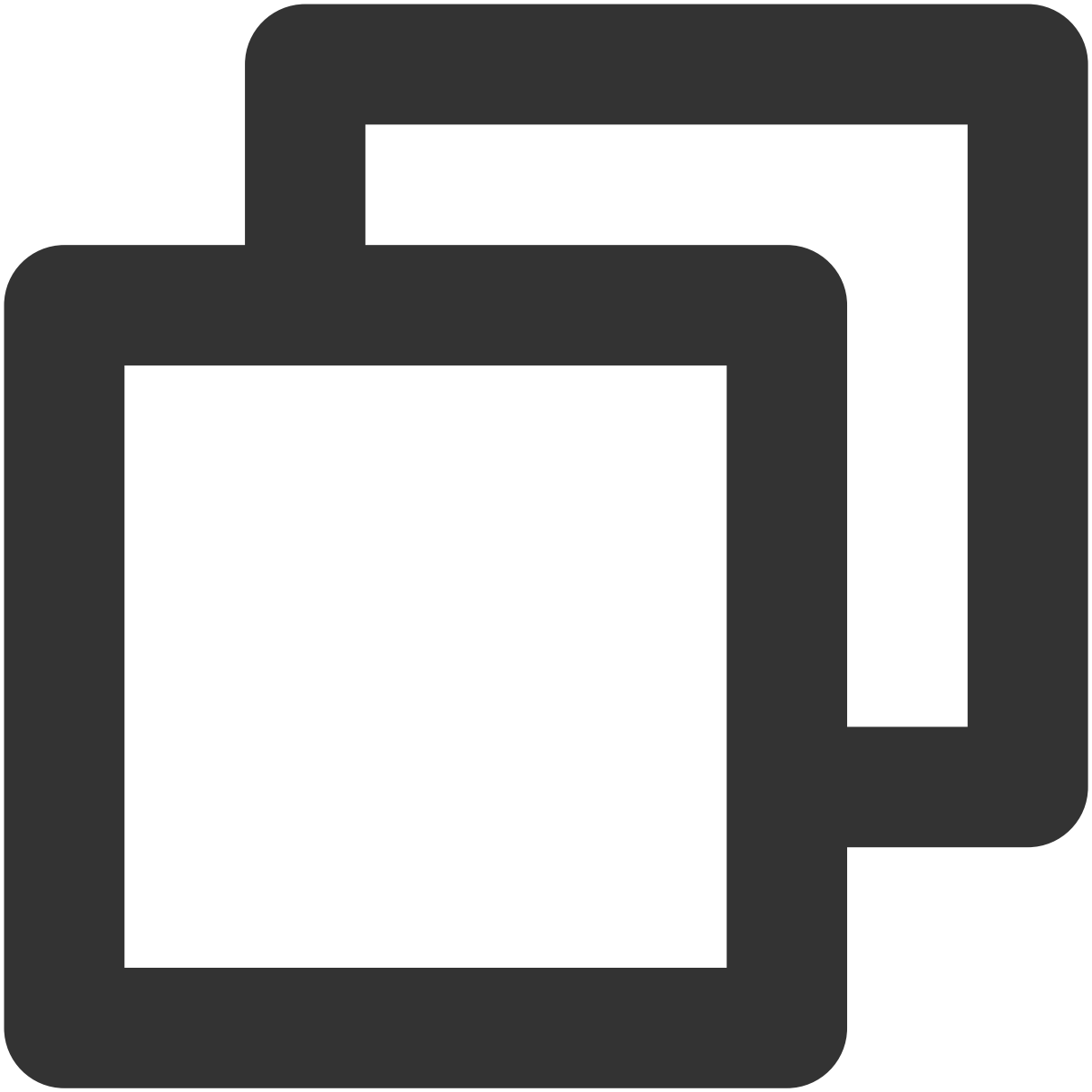
このインターフェースは、録音の停止に使われています。このインターフェースが非同期インターフェースであるため、録音を停止した後には録音完了のコールバックがあります。コールバックが成功してから、録音ファイルが利用できるようになります。

インターフェースのプロトタイプ

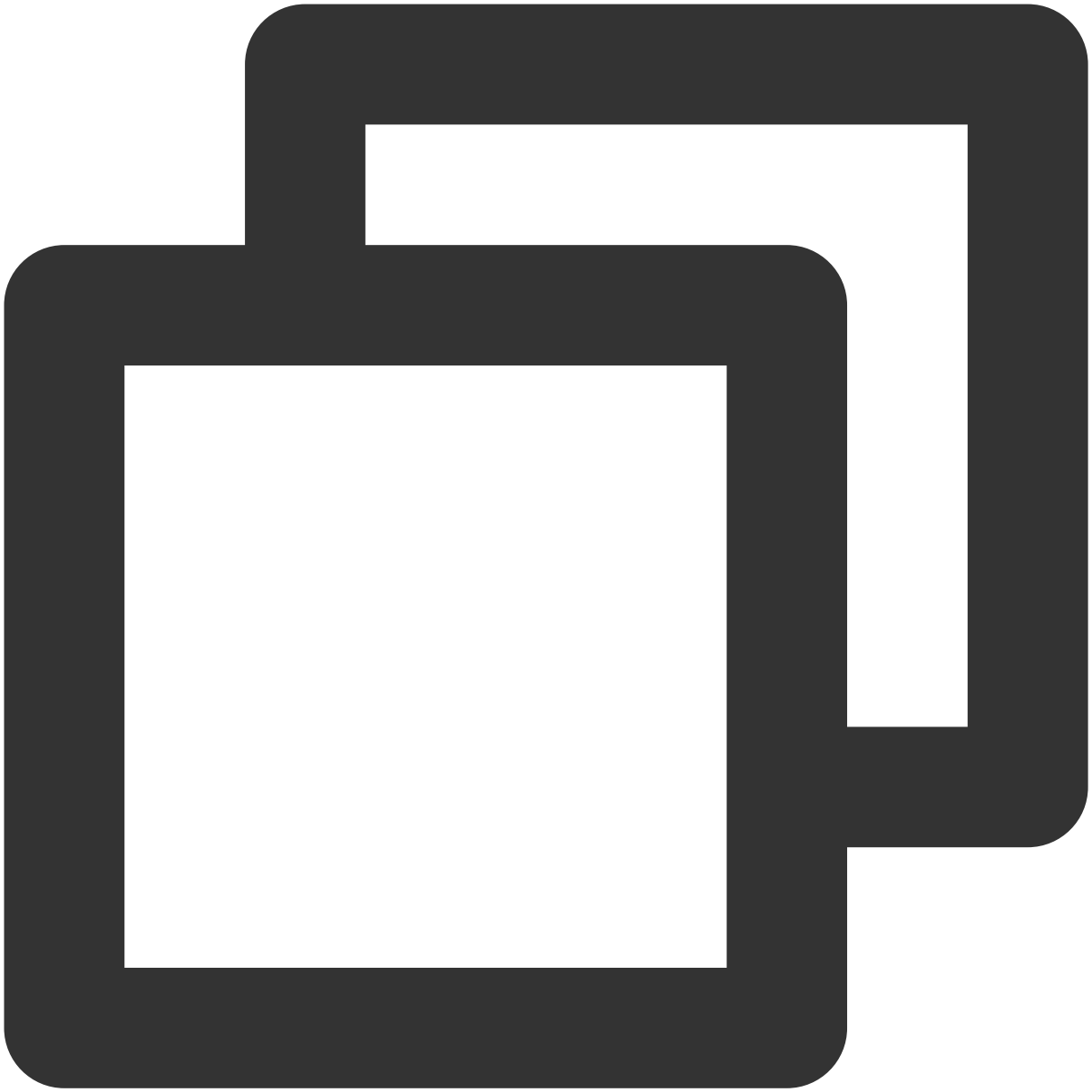
Java

Object-C

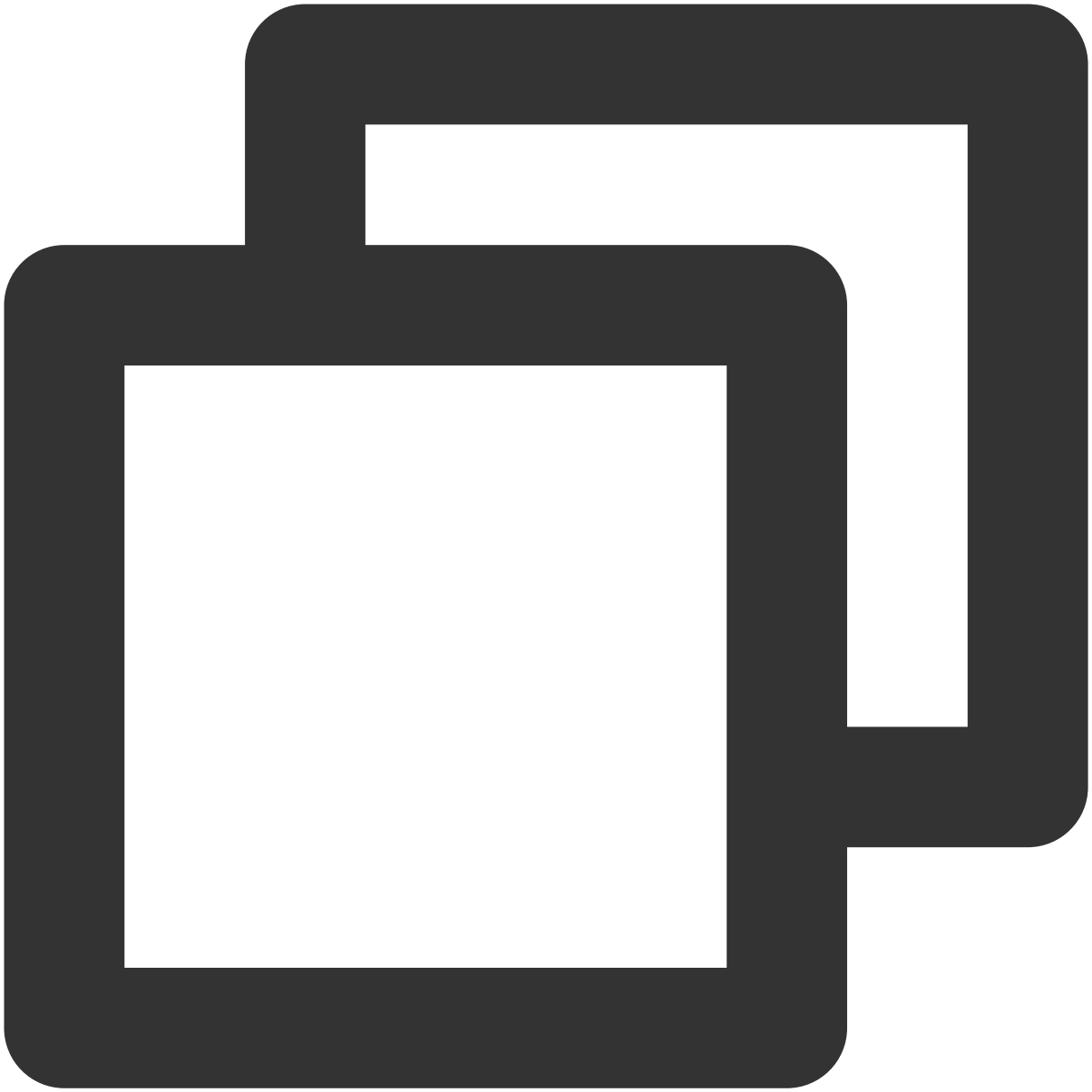
C++



```
public abstract int StopRecording();
```



```
-(QAVResult)StopRecording;
```



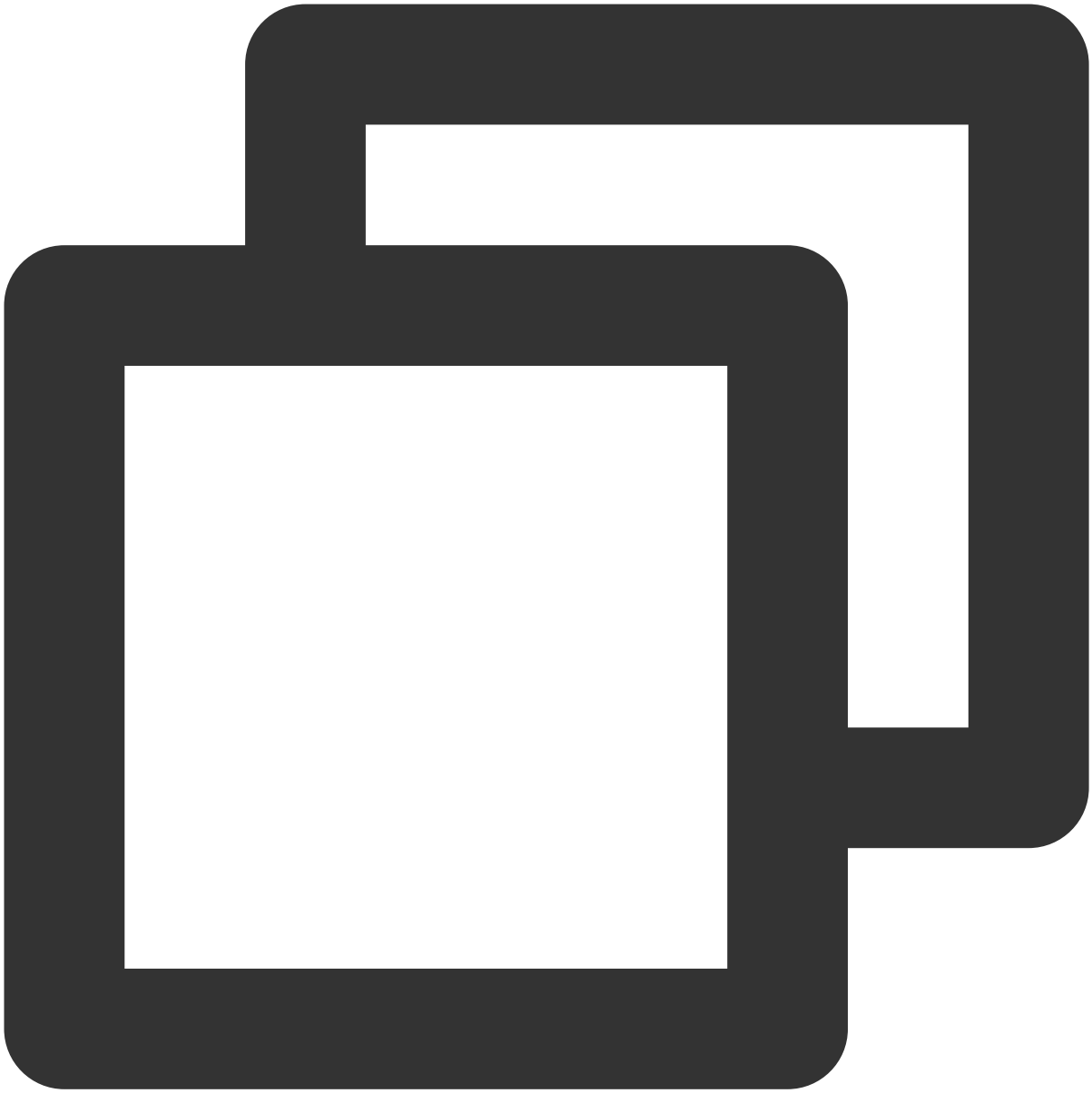
```
ITMGPTT virtual int StopRecording();
```

サンプルコード

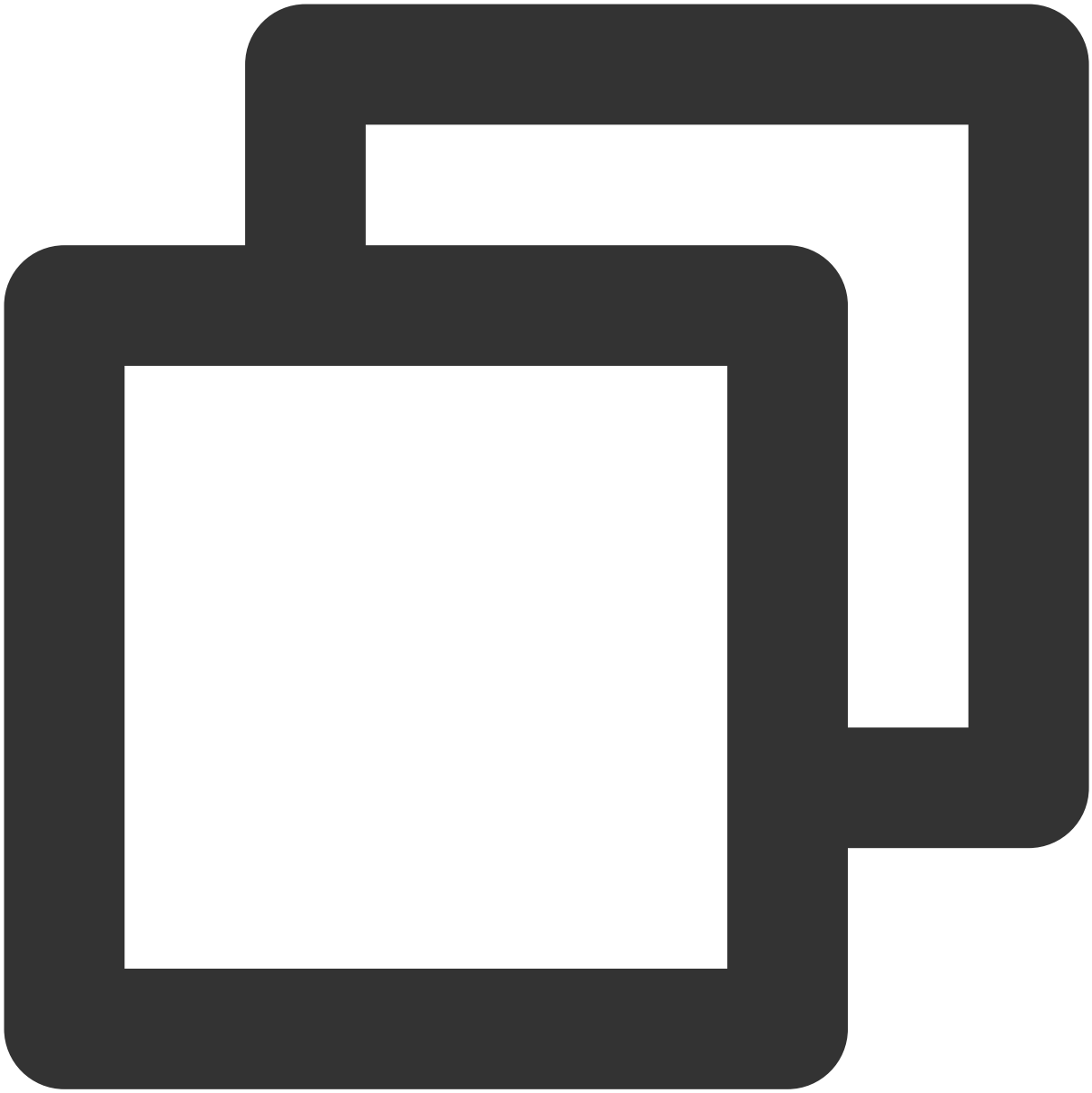
Java

Object-C

C++

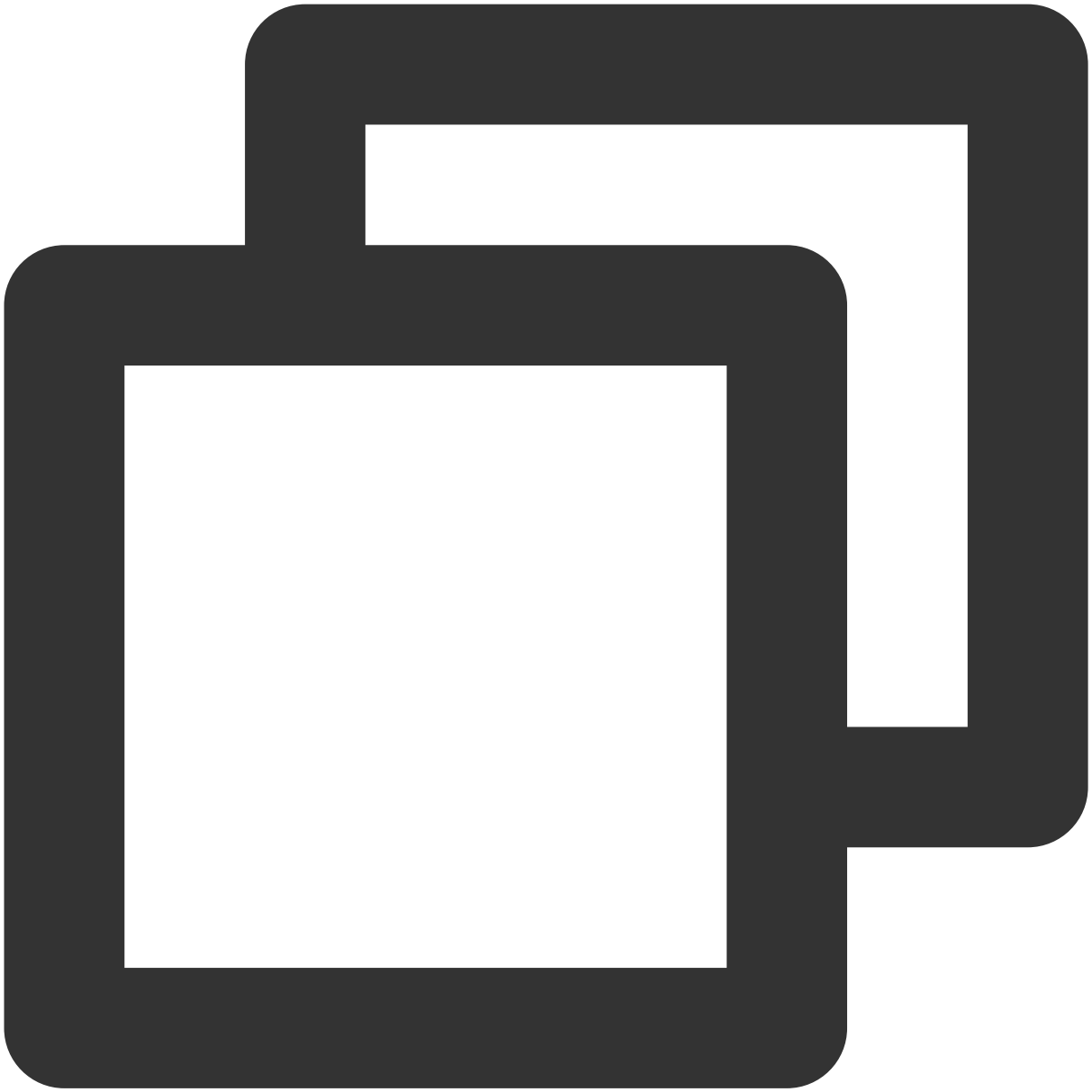


```
//VoiceMessageRecognitionActivity.java
ITMGContext.GetInstance(this).GetPTT().StopRecording();
```



```
//TMGPTTViewController.m

- (void)stopRecClick {
  // Step 3/12 stop recording,  need handle ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLET
  //https://intl.cloud.tencent.com/document/product/607/15221
  QAVResult ret =  [[[ITMGContext GetInstance] GetPTT] StopRecording];
   if (ret == 0) {
       self.currentStatus = @"録音を停止します";
   }else{
        self.currentStatus = @"録音の停止に失敗しました";
   }
```

```
    }
```



```
ITMGContextGetInstance()->GetPTT()->StopRecording();
```

Unity SDKのクイック導入

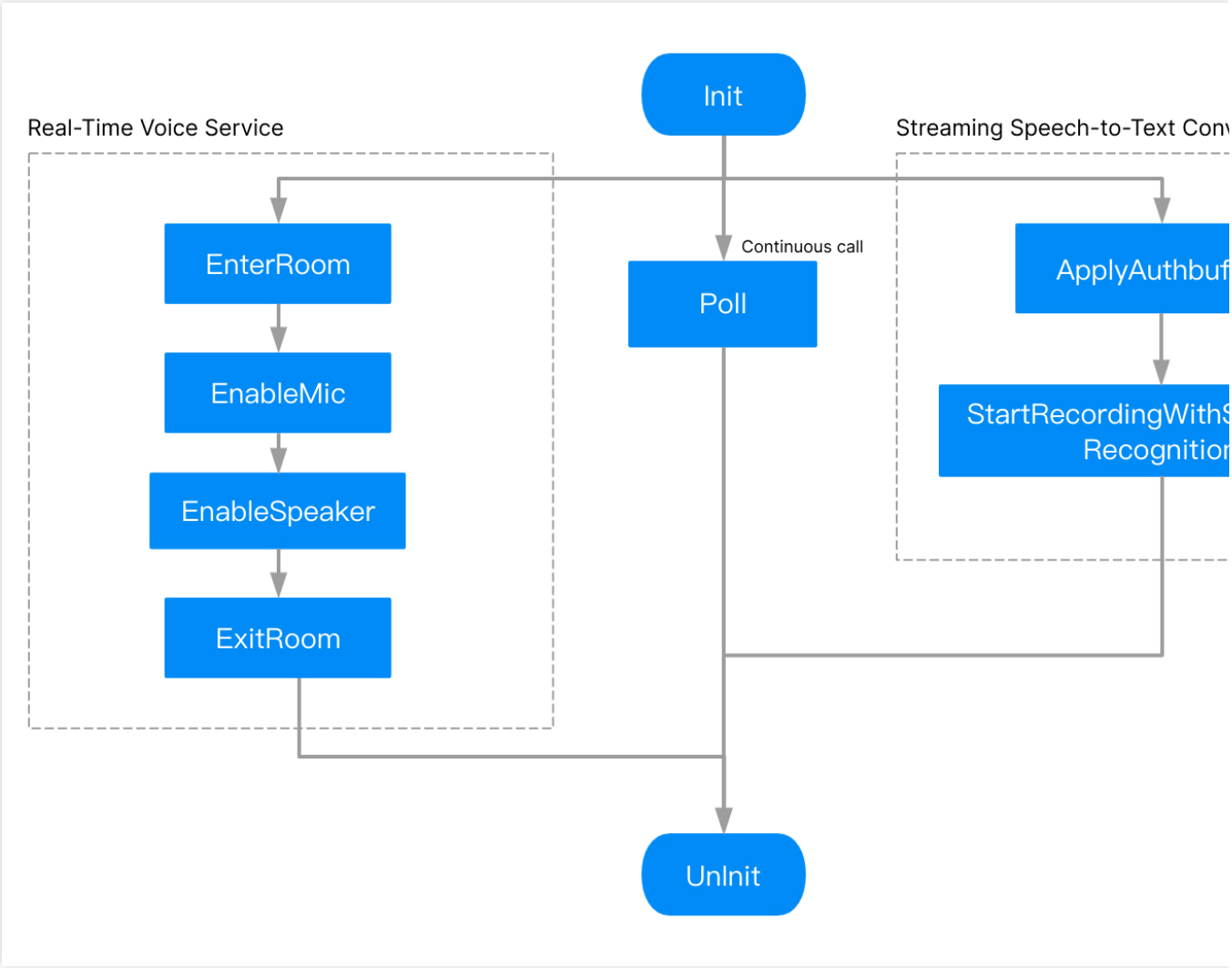
最終更新日：：2024-01-18 15:42:47

開発者がGME製品APIのデバッグ・アクセスを行いやすいように、ここで、Unity開発に適用されるクイックアクセス技術ドキュメントを説明させていただきます。
GMEクイックスタート文書は、ユーザーのアクセスを助けるための最も主要なアクセスインターフェースのみを提供しています。

GME利用上の重要事項

GMEは2つの部分に分かれます。リアルタイム音声サービス、音声メッセージおよびテキスト変換サービスを提供しており、これらのサービスの利用はInitやPollなどのコアインターフェースに依存しています。
关于 Init 接口
例えば、リアルタイムの音声サービスを使用する同時に音声メッセージ・サービスも使用する場合、**Init初期化インターフェースを1回だけ呼び出す必要があります**。

インターフェース呼び出しのフローチャート



統合の手順

SDKの統合

プロジェクトにSDKを統合するには、Unity SDK統合ドキュメントをご参照ください。

コアインターフェース

GMEの初期化，接口：Init
定期的なPoll呼び出しによるコールバックのトリガー，接口：Poll
入退室の通知監視，監听：QAVEnterRoomComplete

リアルタイム音声

1. リアルタイム音声ルームへの参加，接口：EnterRoom
2. マイクのオン/オフ，接口：EnableMic
3. スピーカーのオン/オフ，接口：EnableSpeaker
4. 音声ルーム退出，接口：ExitRoom

音声メッセージ

1. 認証の初期化，接口：ApplyPTTAuthbuffer
2. ストリーミング音声認識の起動，接口：StartRecordingWithStreamingRecognition
3. レコーディング停止，接口：StopRecording

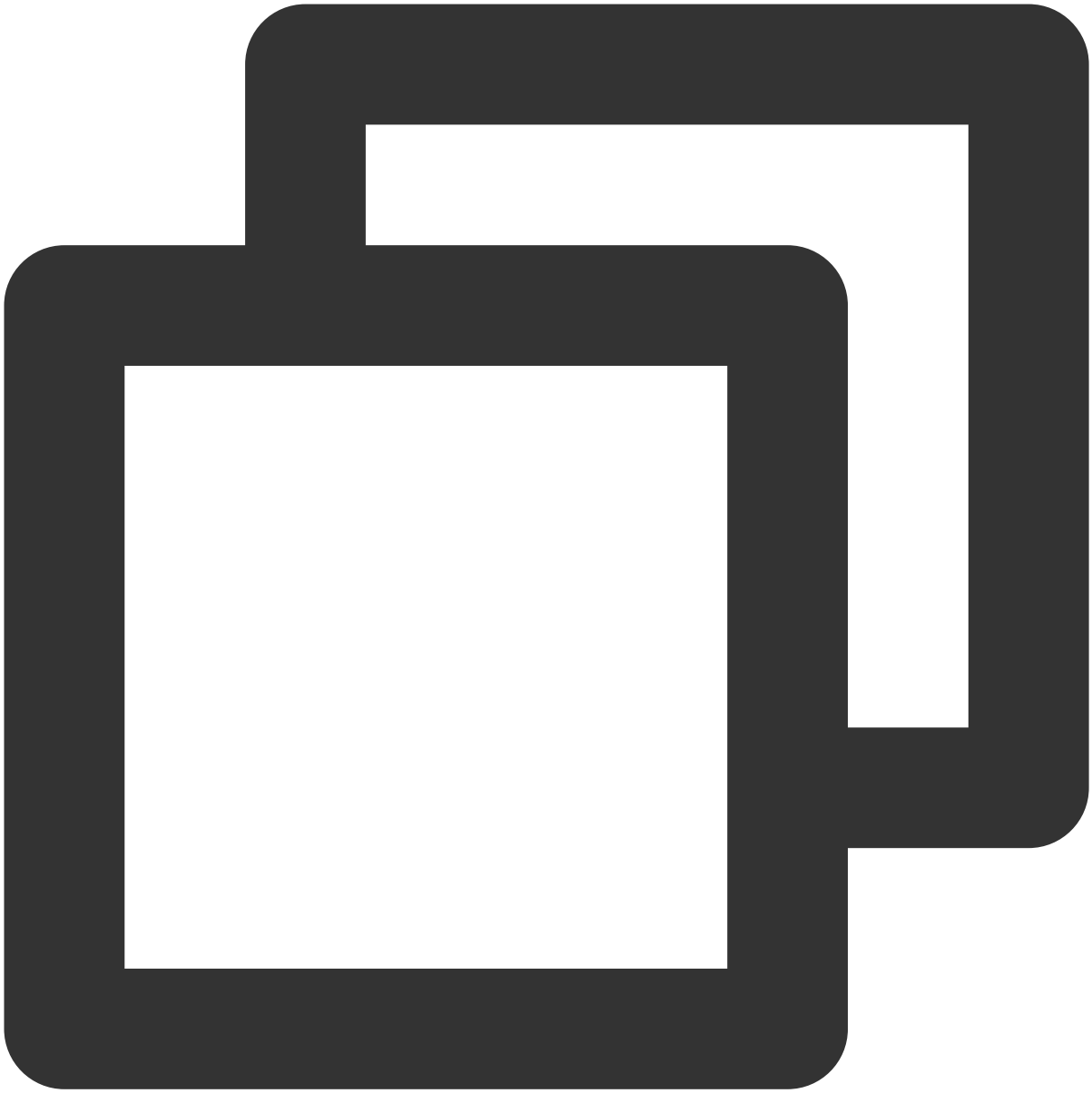
GMEの未初期化，接口：UnInit

コアインターフェースのアクセス

1. SDKのダウンロード

ダウンロード案内ページにアクセスして、必要な クライアンSDKをダウンロードします。

2. ヘッダーファイルを取り込む

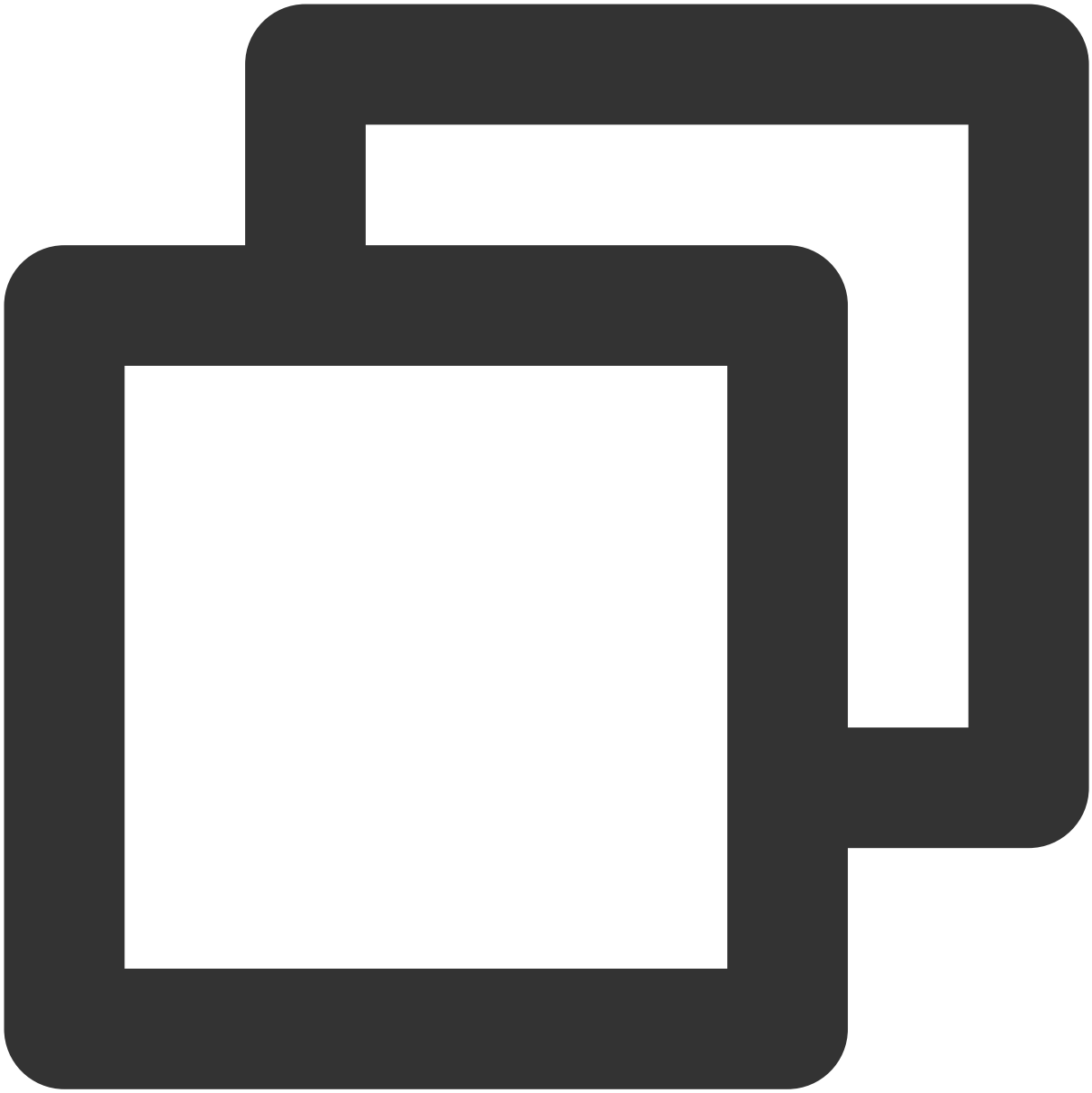


```
using GME;
```

3. Contextインスタンスの取得

QAVContext.GetInstance()の直接呼び出しでインスタンスを取得するのではなく、ITMGContextのメソッドでContextのインスタンスを取得してください。

サンプルコード

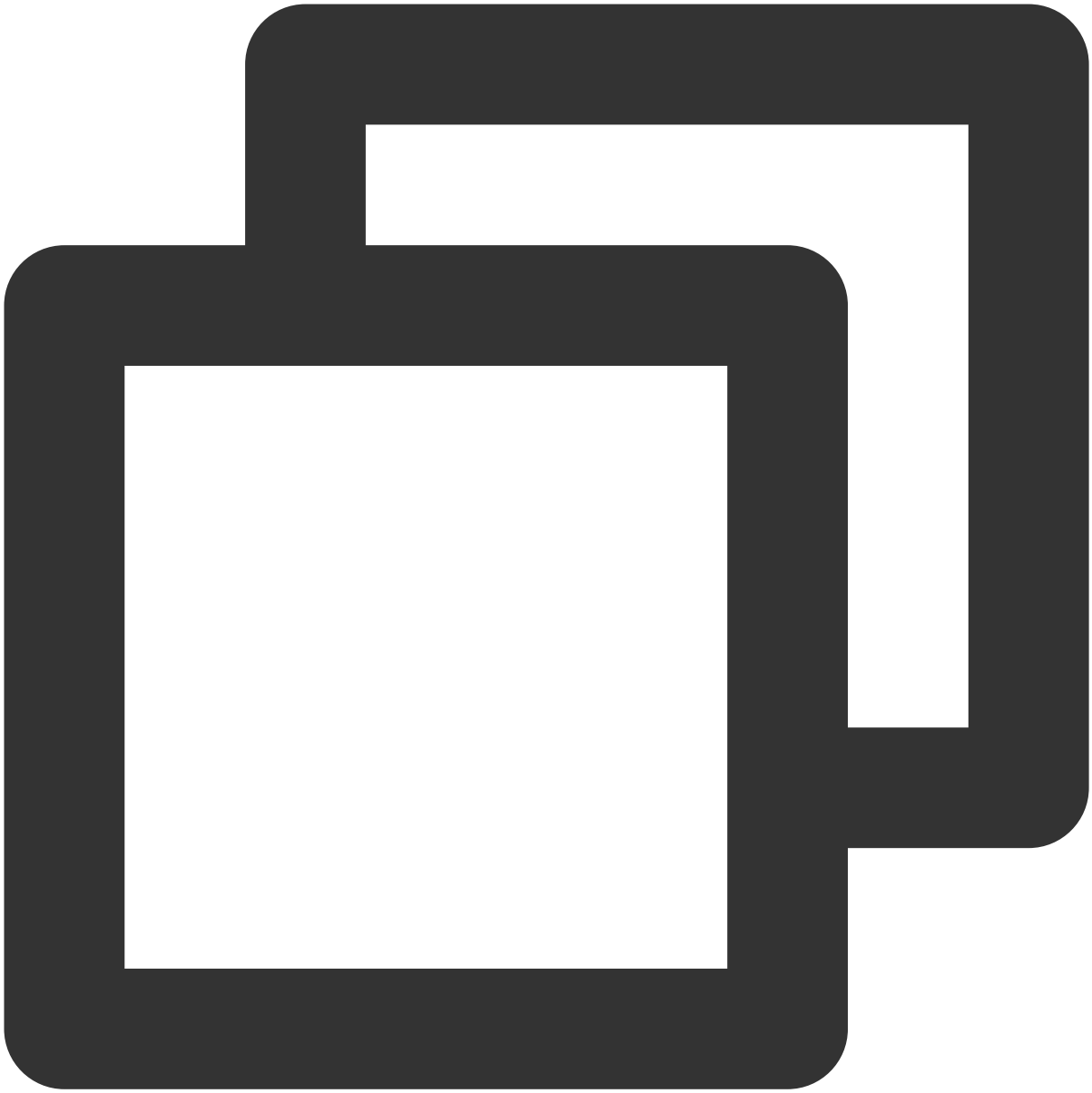


```
int ret = ITMGContext.GetInstance().Init(sdkAppId, openID);
```

4. SDKを初期化する

初期化前のSDKは初期化されていない状態です。リアルタイム音声サービス、音声メッセージサービスおよびボイスツーテキスト変換サービスを使用するには、**インターフェースInitを使用してSDKを初期化する必要があります**。Initインターフェースを呼び出すスレッドは、他のインターフェースと同じスレッドである必要があります。すべてのメインスレッドでインターフェースを呼び出すことをお勧めします。

インターフェースのプロトタイプ

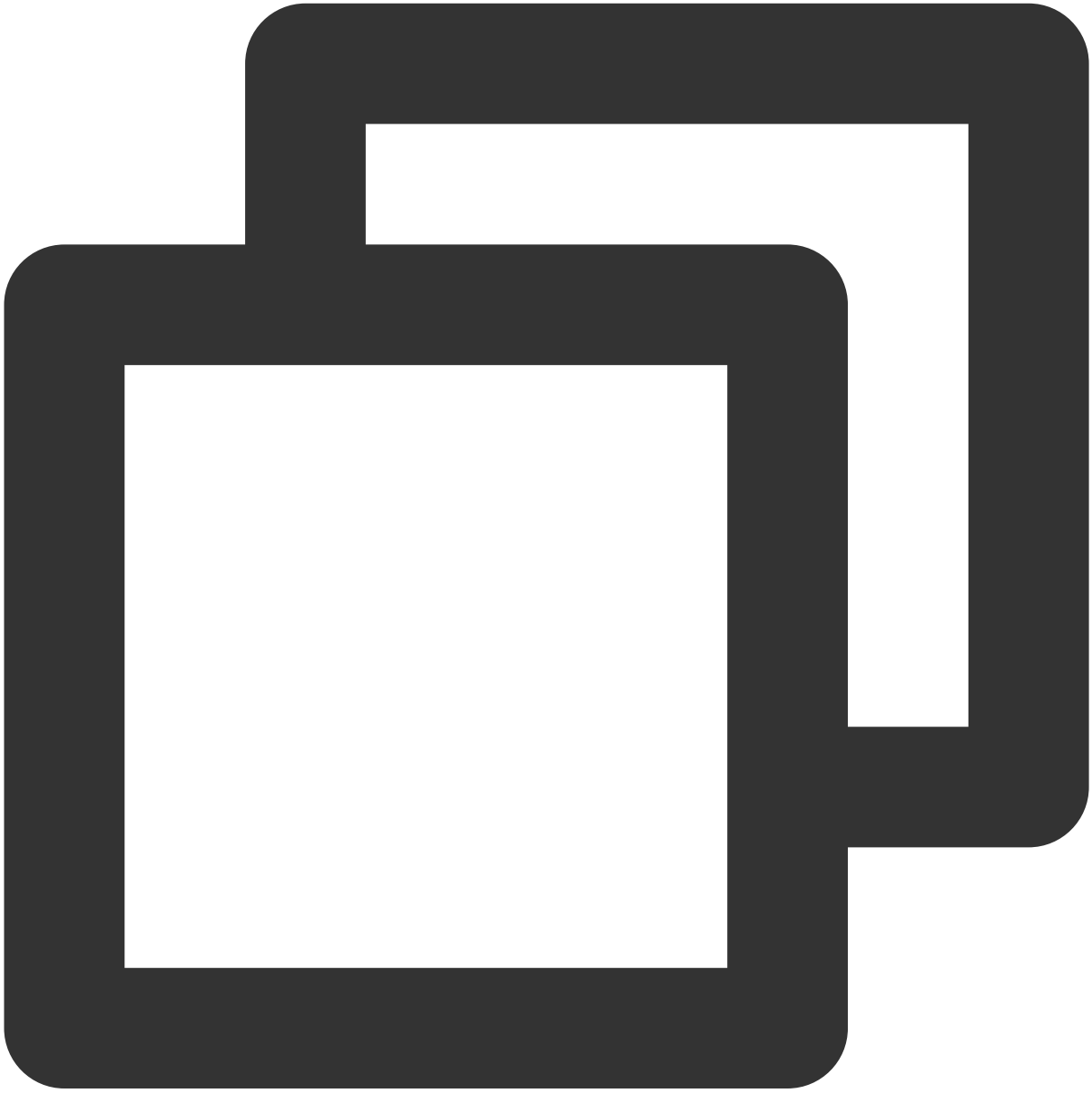


```
//class ITMGContext
public abstract int Init(string sdkAppID, string openID);
```

パラメータ	タイプ	意味
sdkAppId	string	Tencent Cloud ConsoleのGMEサービスが提供するAppIDです。取得についてはサービス開始ガイドラインをご参照ください。
openID	string	openIDはInt64型（stringに変換して渡す）のみに対応しており、ルールはApp開発者

		が独自に定め、App内で重複しなければよい。文字列をOpenidとして渡す必要がある場合は、チケットを提出をして開発者に連絡してください。

サンプルコード



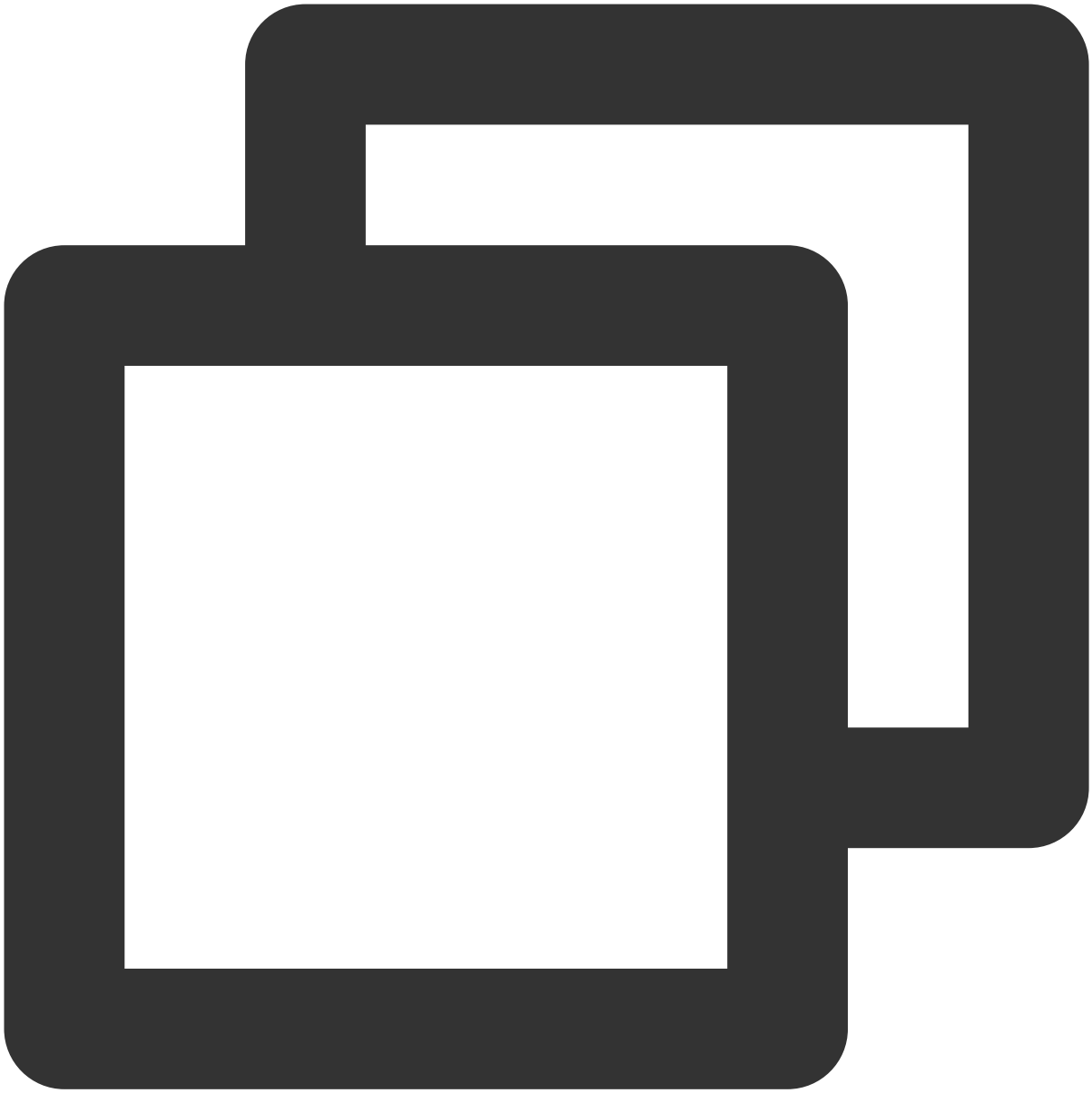
```
int ret = ITMGContext.GetInstance().Init(sdkAppId, openID);
//戻り値で初期化が成功したかどうかを判断する
if (ret != QAVError.OK)
 {
     Debug.Log("SDK初期化失敗:"+ret);
```

```
        return;
    }
```

5. イベントコールバックのトリガー

updateで周期的にPollを呼び出すことで、イベントのコールバックをトリガできます。PollはGMEのメッセージポンプであり、GMEはイベントのコールバックをトリガするためにPollインターフェースを定期的に呼び出す必要があります。Pollが呼び出されないと、SDKサービス全体が異常に動作します。詳細については、Sample ProjectのEnginePollHelperファイルをご参照ください。

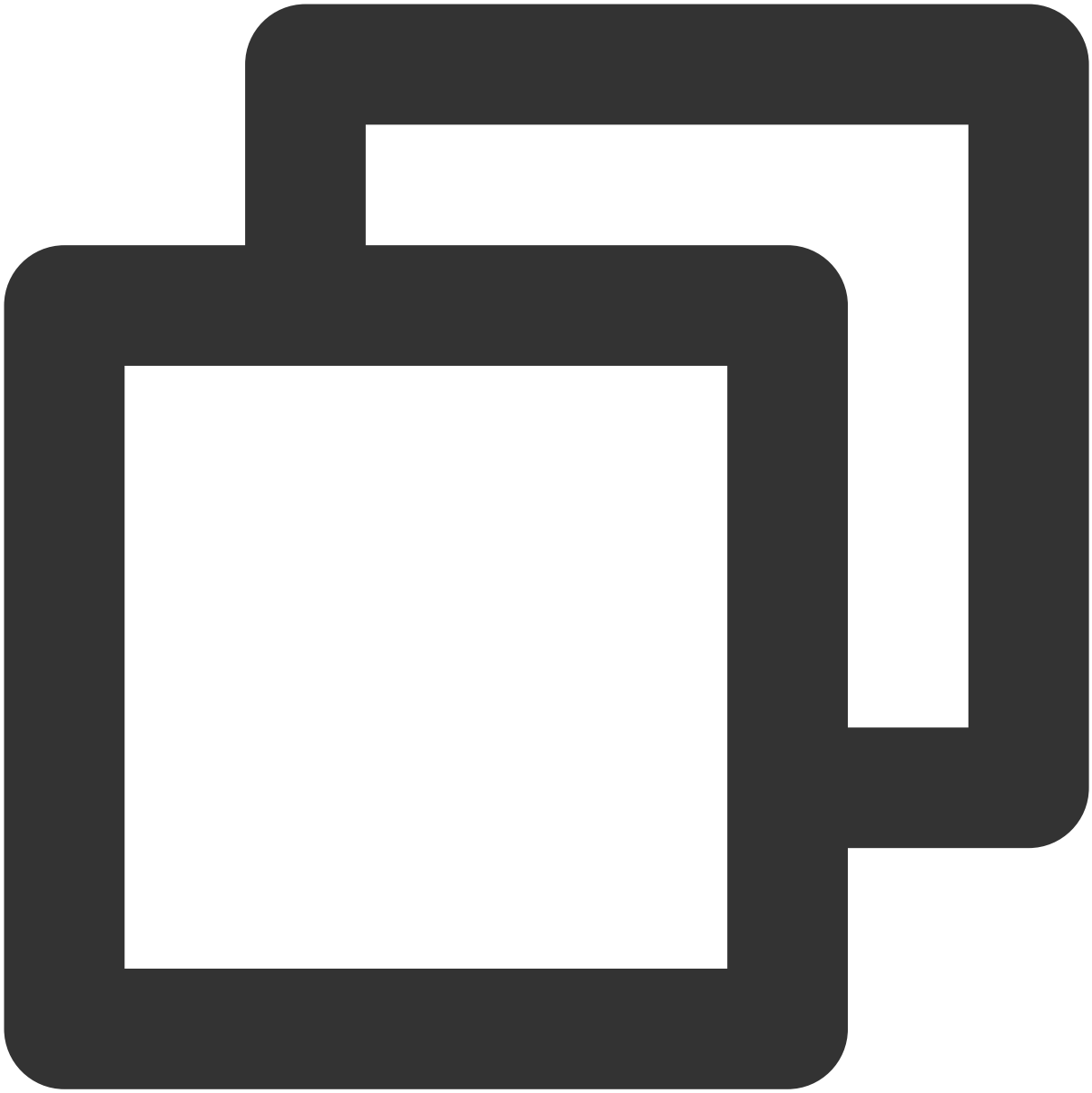
サンプルコード



```
public void Update()
  {
      ITMGContext.GetInstance().Poll();
  }
```

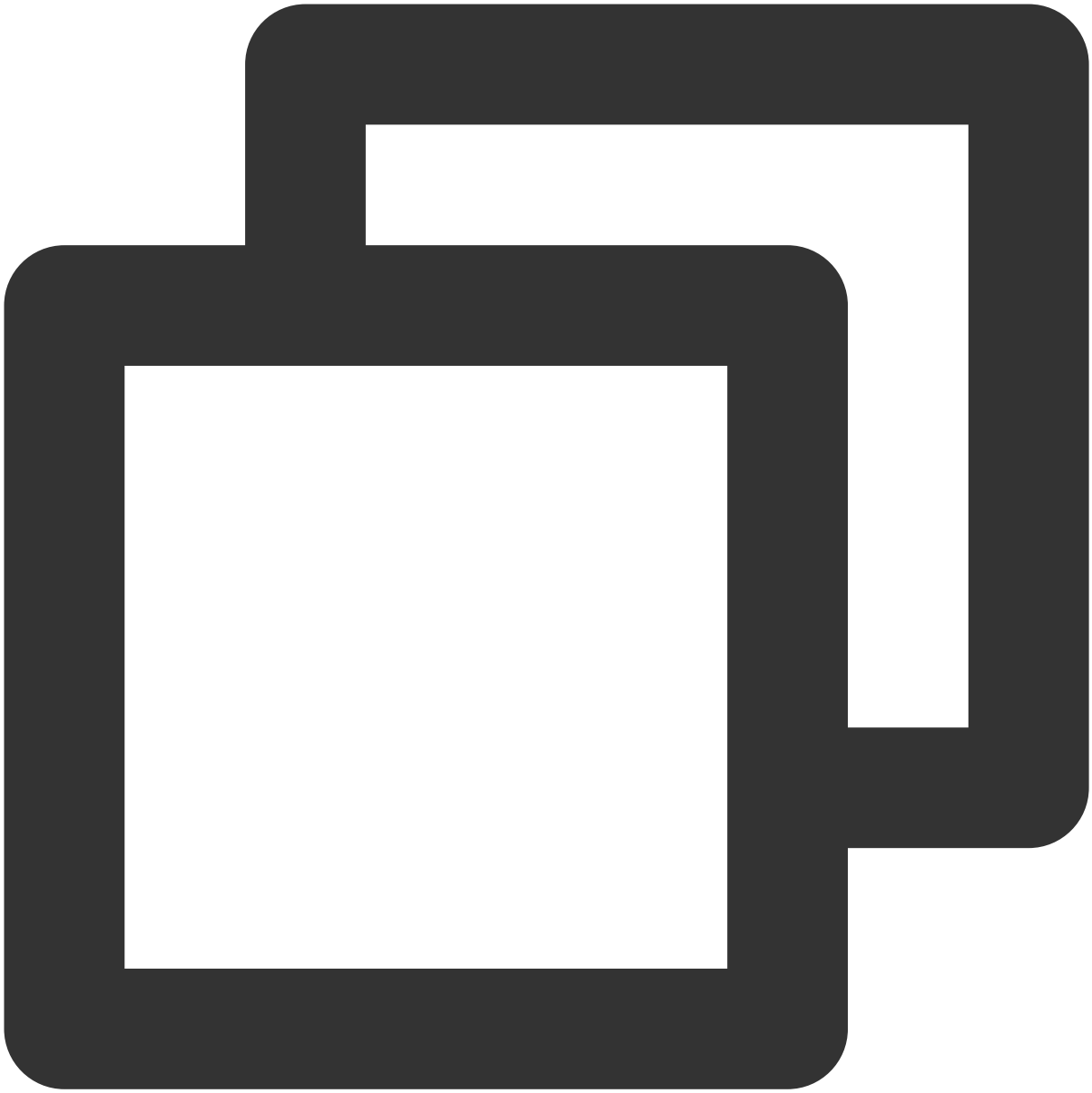
6. 入退室の通知監視

入室通知



```
//委託関数：
public delegate void QAVEnterRoomComplete(int result, string error_info);
//イベント関数：
public abstract event QAVEnterRoomComplete OnEnterRoomCompleteEvent;
```

退室通知

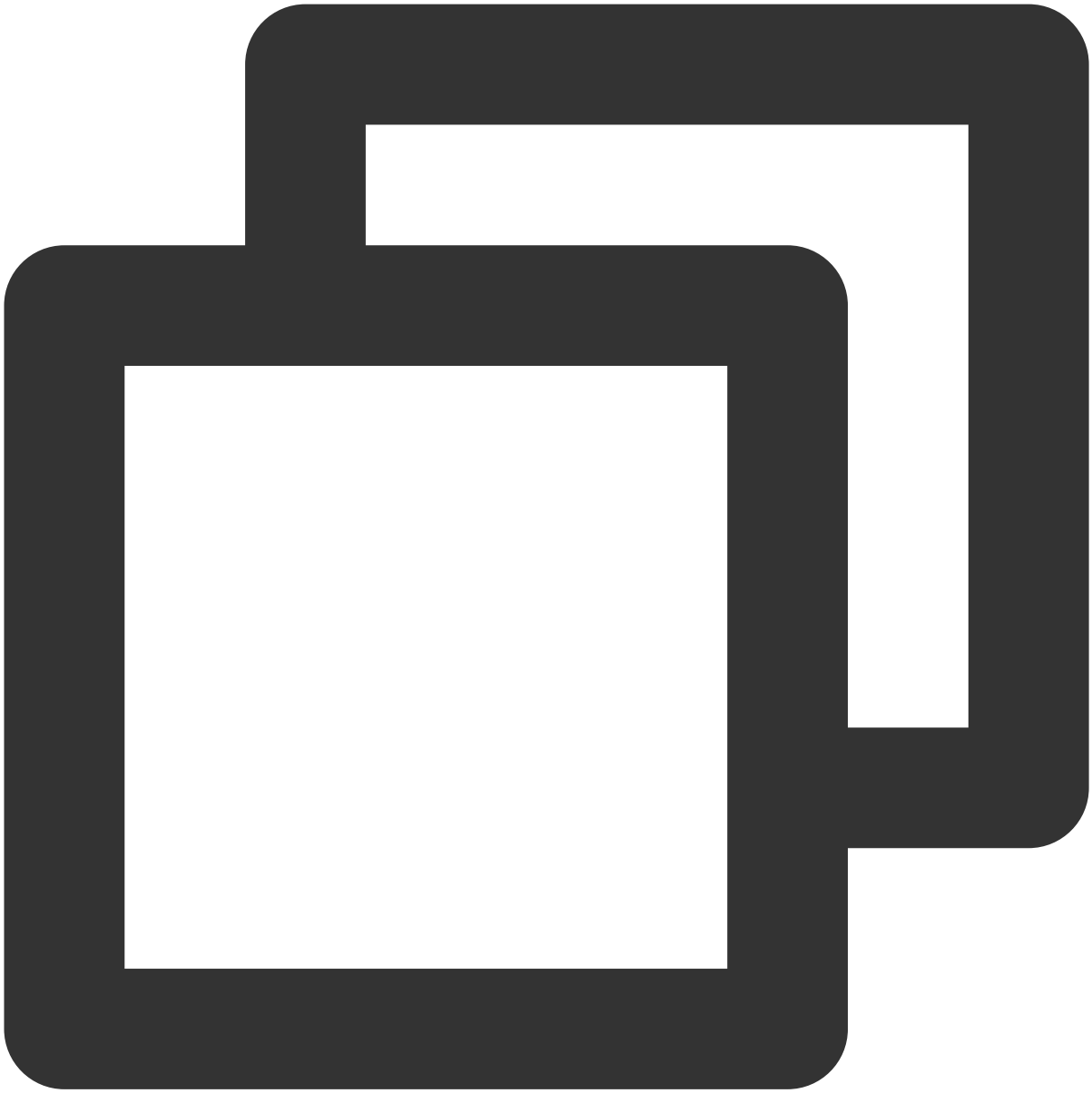


委託関数：
```
public delegate void QAVExitRoomComplete();
```
イベント関数：
```
public abstract event QAVExitRoomComplete OnExitRoomCompleteEvent;
```

7. ローカル認証計算

AuthBufferを生成し、関連機能の暗号化と認証に使用します。本格なリリースについてバックグラウンドのデプロイキーを使用してください。バックグラウンドのデプロイについては、認証キーをご参照ください。

インターフェースのプロトタイプ

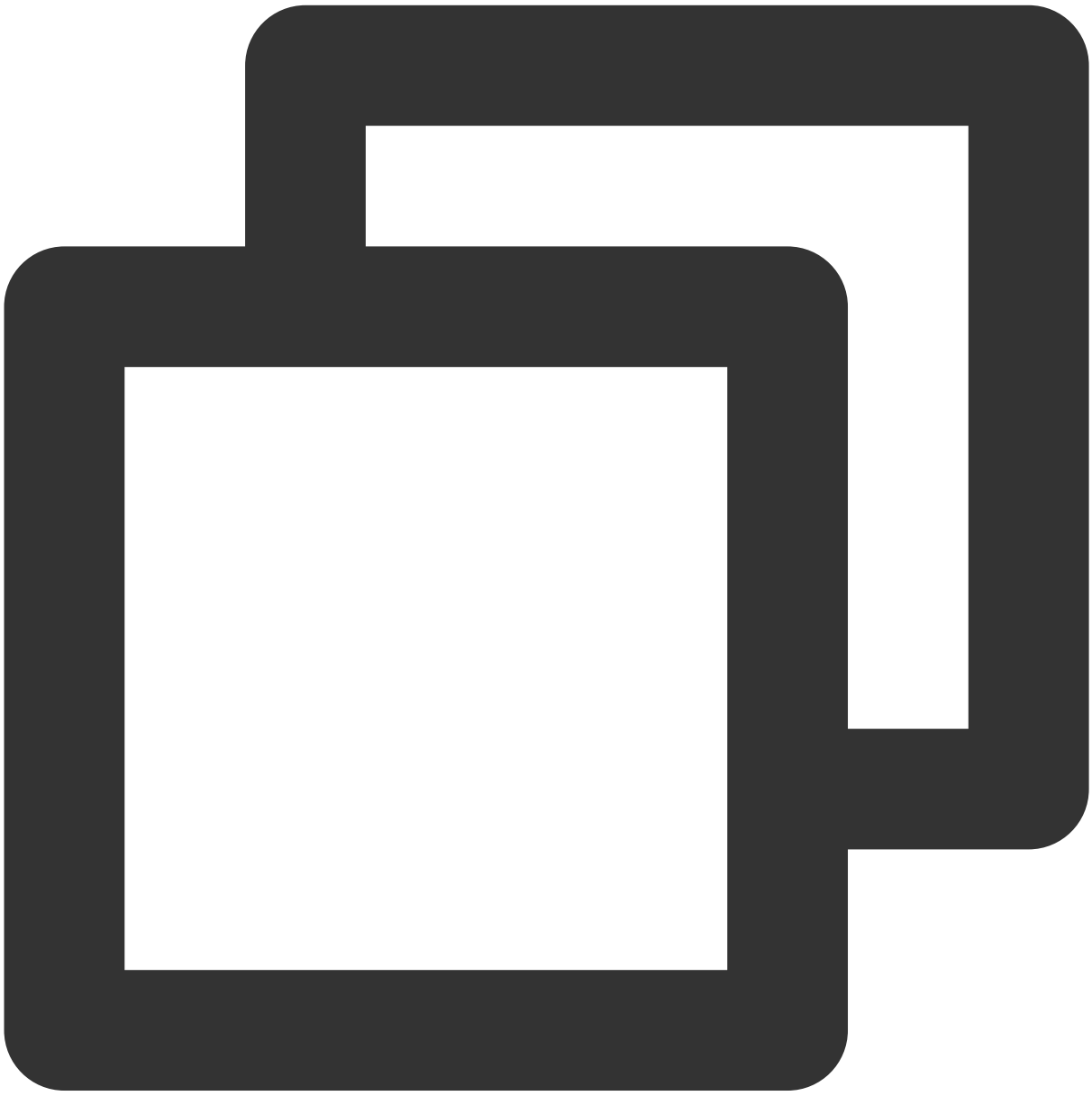


```
QAVAuthBuffer GenAuthBuffer(int appId, string roomId, string openId, string key)
```

パラメータ	タイプ	意味
appId	int	Tencent Cloudコンソールからの AppId番号。
roomId	string	ルーム番号であり、最大127文字まで対応しています（オフライン音声ルーム番号のパラメータをnullに設定しなければなりません）。

openId	string	ユーザーID。Initの場合のopenIdと同じです。
key	string	Tencent Cloudコンソールからの権限キー。

サンプルコード



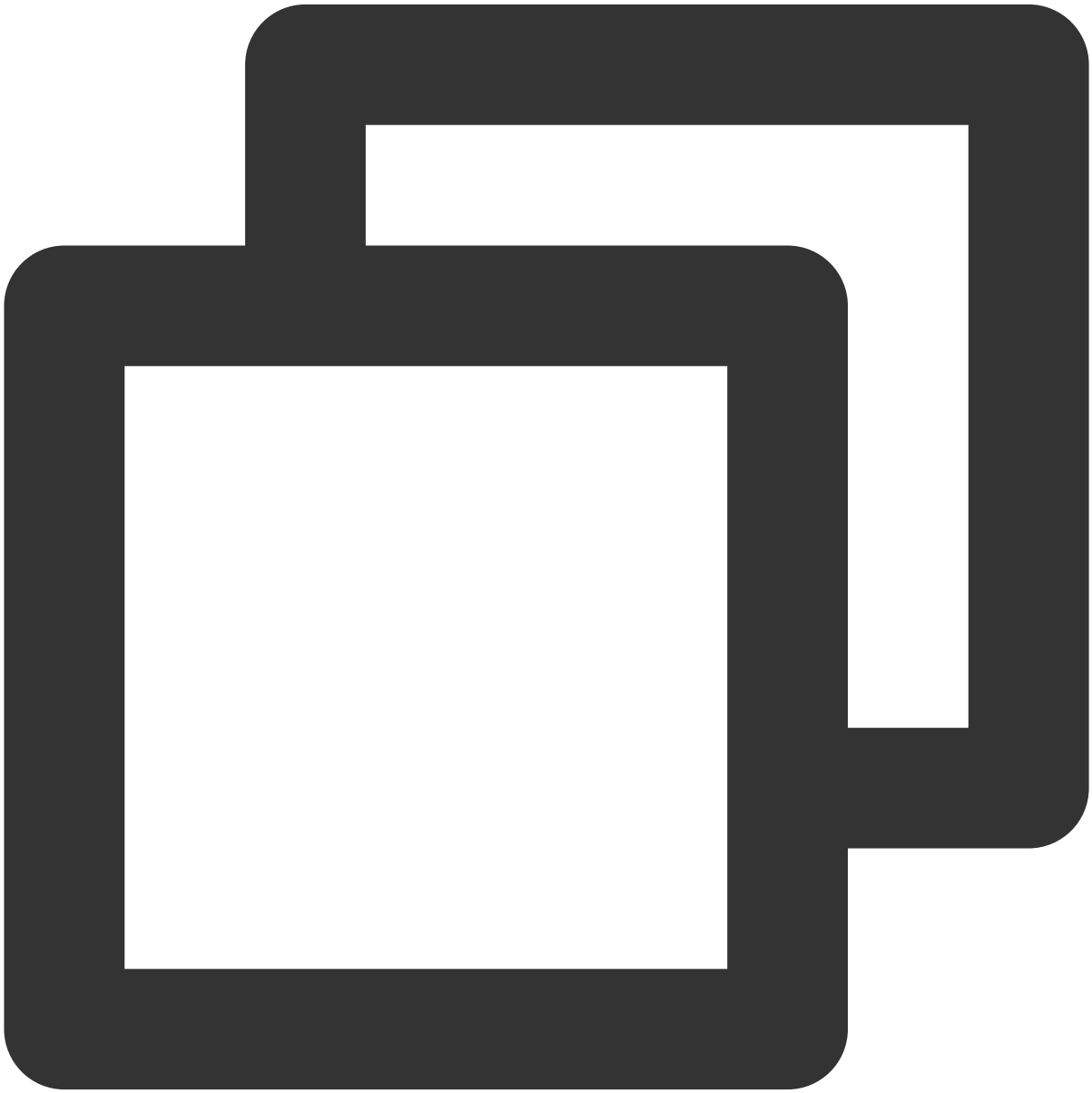
```
public static byte[] GetAuthBuffer(string AppID, string RoomID,string OpenId, strin
    return QAVAuthBuffer.GenAuthBuffer(int.Parse(AppID), RoomID, OpenId, AuthKey);
}
```

リアルタイム音声アクセス

1. ルームに参加

生成した認証情報を用いてルームに参加します。ルームに参加するとき、デフォルトでマイクとスピーカーはオフです。戻り値がAV_OKの場合はルーム参加が成功したことでなく、呼び出しが成功したことを意味します。

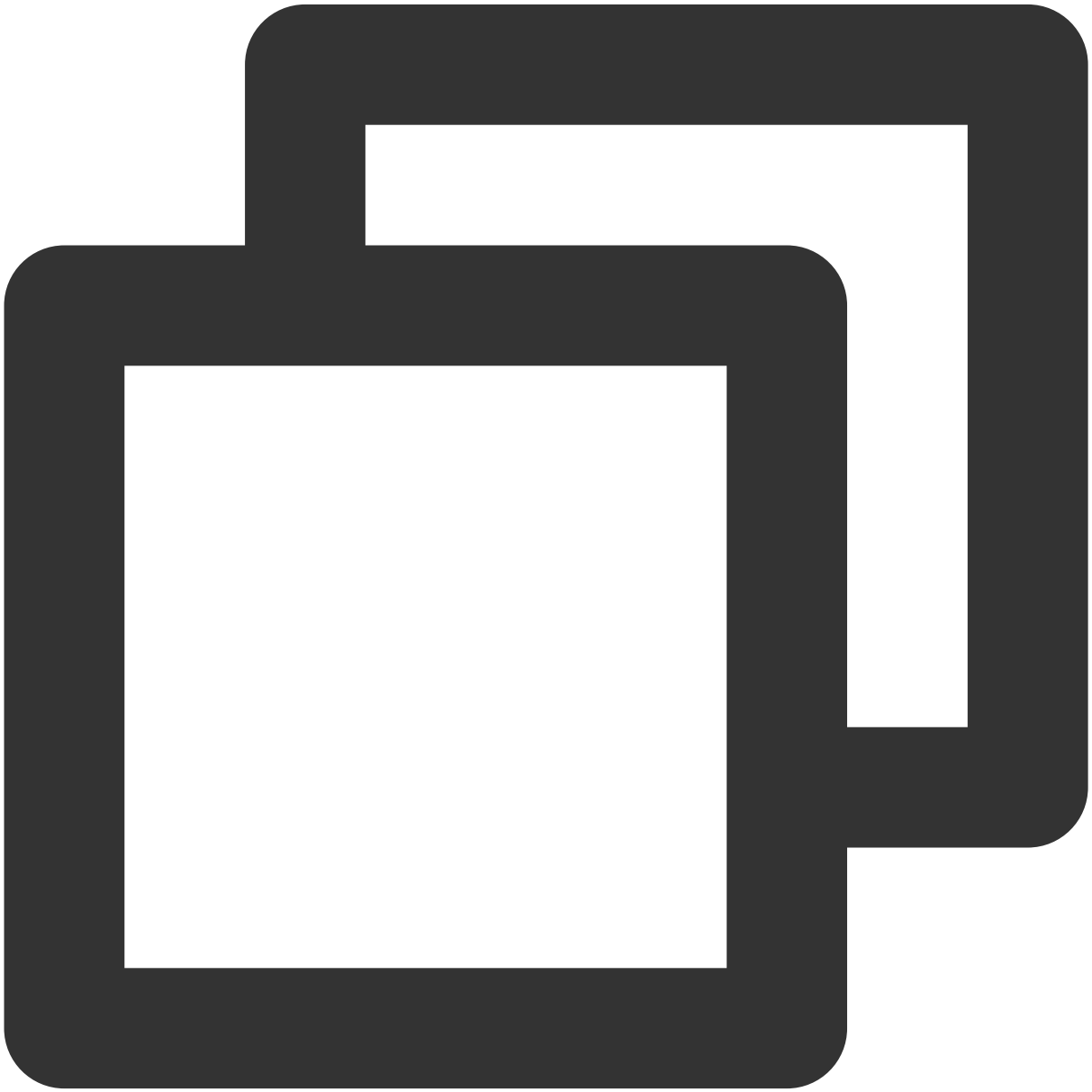
インターフェースのプロトタイプ



```
ITMGContext EnterRoom(string roomId, int roomType, byte[] authBuffer)
```

パラメータ	タイプ	意味
roomId	String	ルーム番号、127文字まで入力可能
roomType	ITMGRoomType	ITMGRoomType.ITMG_ROOM_TYPE_FLUENCYのみ入力すればよい
authBuffer	byte[]	認証コード

サンプルコード



```
ITMGContext.GetInstance().EnterRoom(strRoomId, ITMGRoomType.ITMG_ROOM_TYPE_FLUENCY,
```

入室イベントのコールバック

ルーム参加が完了するとコールバックにより入室結果が返され、入室結果イベントを監視して処理が行われます。コールバックが成功した場合は、その時点で入室が成功し、**課金**が開始されます。

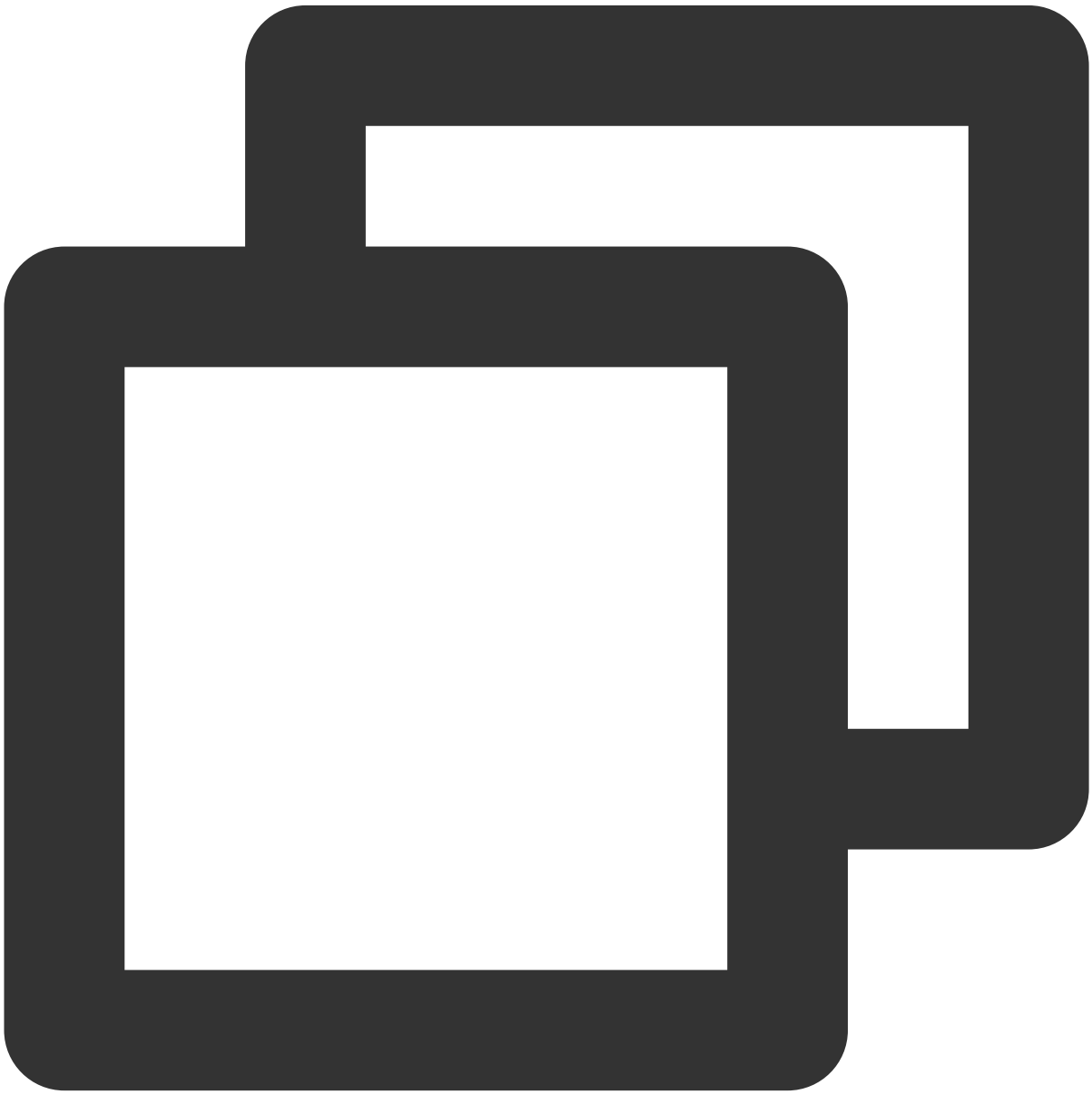
计费问题参考

購入ガイド。

課金に関するよくあるご質問。

リアルタイム音声を使用した後、クライアントの接続が切れた場合課金は継続されますか。

サンプルコードコールバック処理の関連参照コード



```
//イベントを監視します：
ITMGContext.GetInstance().OnEnterRoomCompleteEvent += new QAVEnterRoomComplete(OnEn
//監視処理：
void OnEnterRoomComplete(int err, string errInfo)
{
if (err != 0) {
  ShowLoginPanel("エラーコード:" + err + "エラーメッセージ:" + errInfo);
  return;
}
else{
  //入室に成功
}
}
```

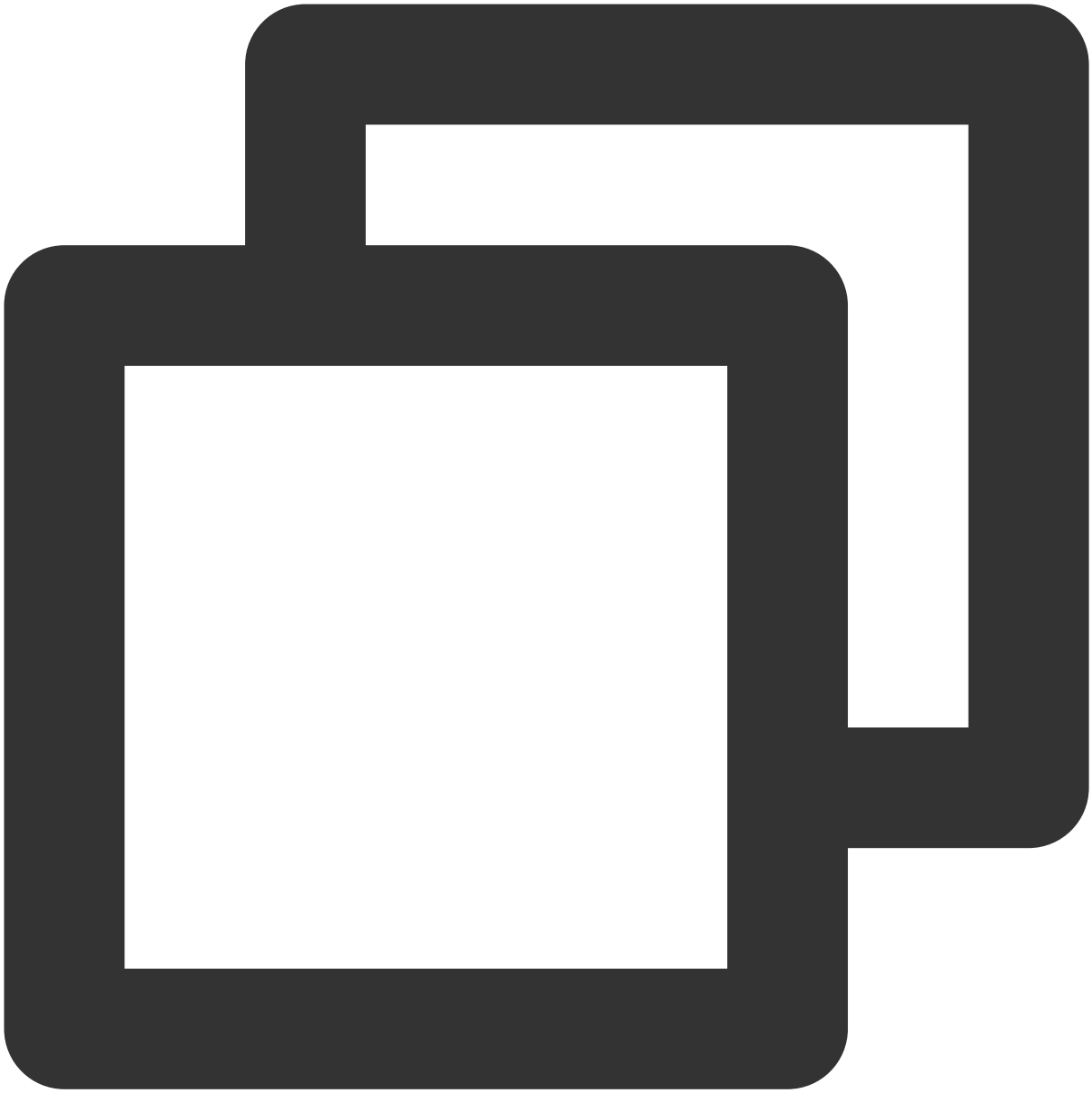
エラーコード

エラーコードの値	原因と解決策
7006	次の理由で認証に失敗しました： AppIDが存在しないか、エラーです authbuff認証エラーです 認証期限切れです openIdが仕様に準拠していません
7007	他のルームにいます
1001	ルーム参加中でこの操作を繰り返しています。コールバックが戻るまで、ルーム参加インターフェースを呼び出さないことをお勧めします
1003	ルームに参加してルームにいますが、もう1回ルーム参加インターフェースを呼び出しました
1101	SDKが初期化されていること、openIdが規則に準拠していること、またはインターフェースが同じスレッドで呼び出されていること、およびPollインタフェースが正常に呼び出されていることを確認してください

2. マイクのオン/オフ

このインターフェースは、マイクのオン/オフに使用されます。入室する際、マイクとスピーカーはデフォルトでオフになっています。

サンプルコード



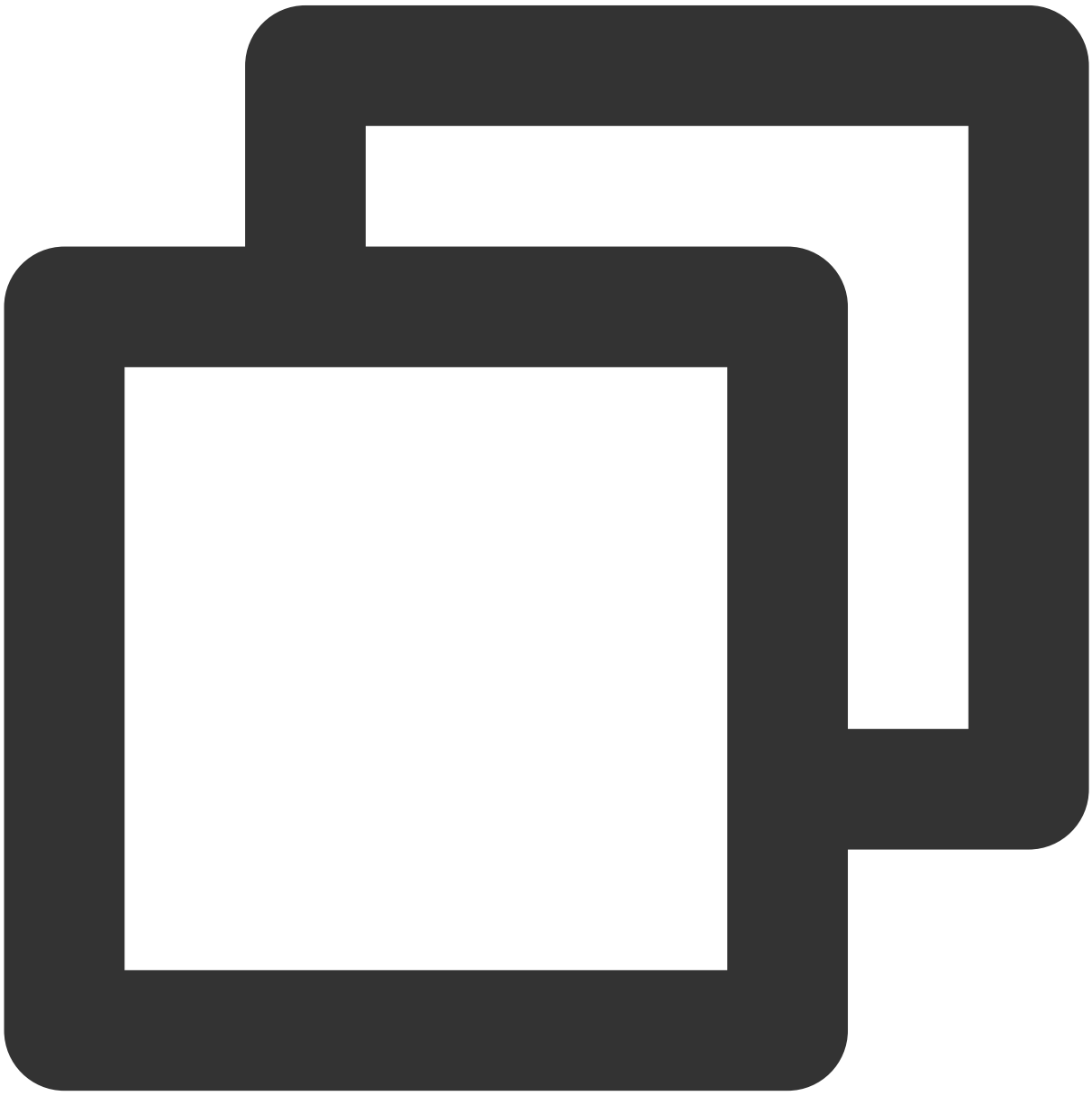
```
//イベントを監視します：
ITMGContext.GetInstance().OnEnterRoomCompleteEvent += new QAVEnterRoomComplete(OnEn
//監視処理：
void OnEnterRoomComplete(int err, string errInfo)
{
  if (err != 0) {
    ShowLoginPanel("エラーコード:" + err + "エラーメッセージ:" + errInfo);
    return;
  }
  else{
    //入室に成功
```

```
        // マイクの起動
        ITMGContext.GetInstance().GetAudioCtrl().EnableMic(true);
    }
}
```

3. スピーカーのオン/オフ

このインターフェースは、スピーカーのオン/オフに使用されます。

サンプルコード

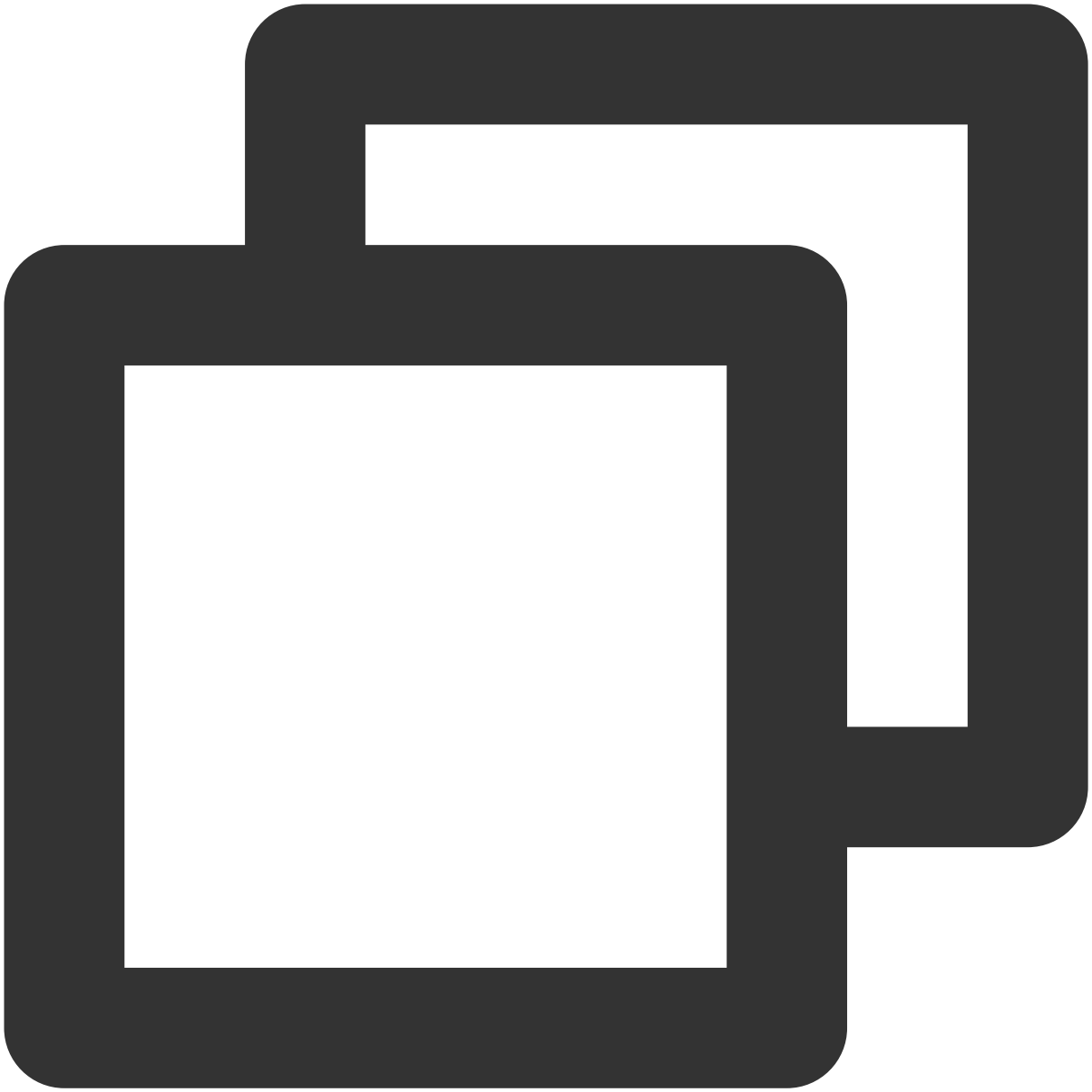


```
//イベントを監視します：
ITMGContext.GetInstance().OnEnterRoomCompleteEvent += new QAVEnterRoomComplete(OnEn
//監視処理：
void OnEnterRoomComplete(int err, string errInfo)
{
  if (err != 0) {
    ShowLoginPanel("エラーコード:" + err + "エラーメッセージ:" + errInfo);
    return;
  }
  else{
    //入室に成功
    //スピーカーをオンにする
    ITMGContext.GetInstance().GetAudioCtrl().EnableSpeaker(true);
  }
}
```

4. 退室

このインターフェースを呼び出すと、退室することができます。処理の実行は退室のコールバックを待つ必要があります。

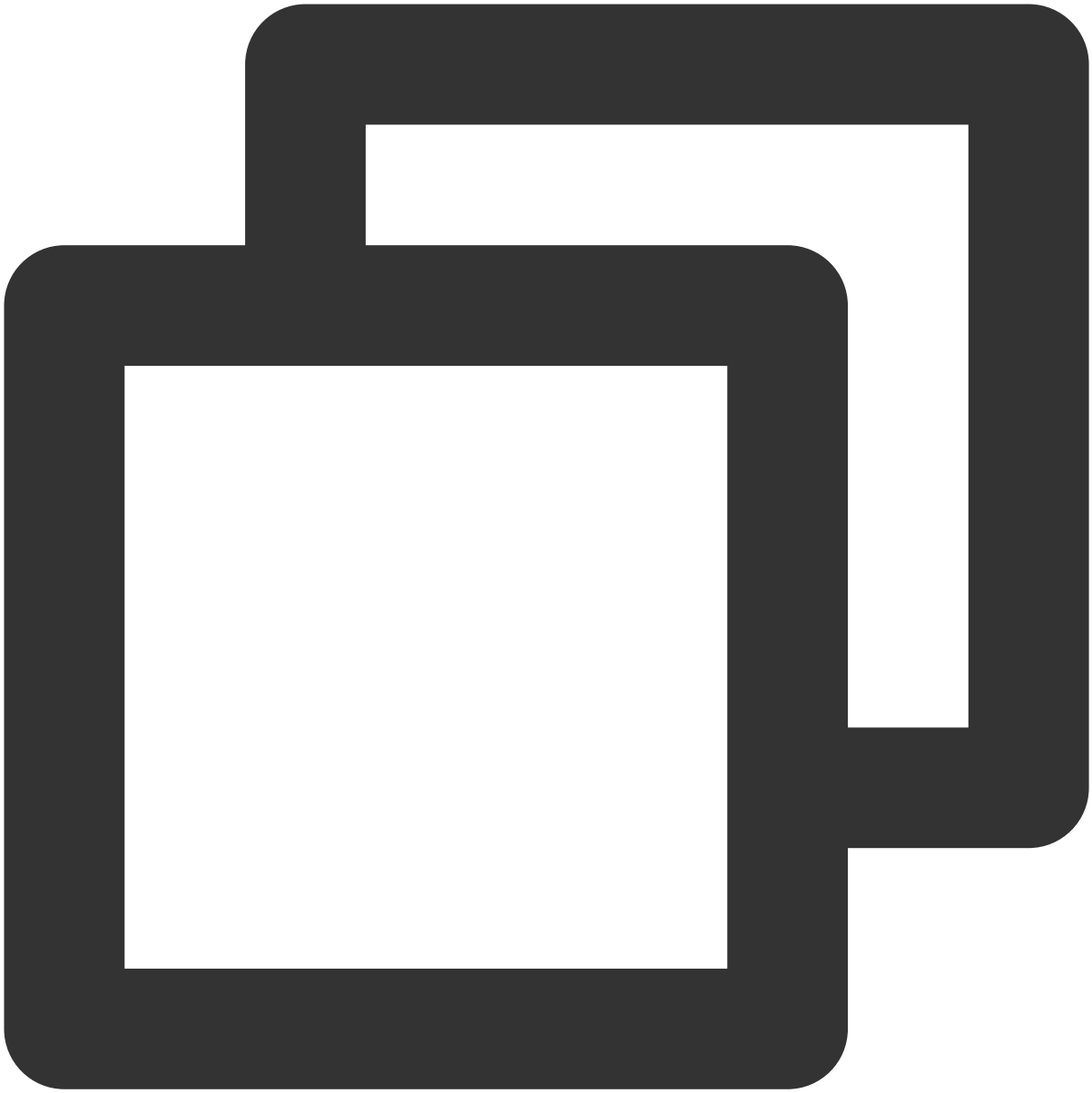
サンプルコード



```
ITMGContext.GetInstance().ExitRoom();
```

退室コールバック

退室してからはコールバックが発生します、サンプルコードは次の通りです：



イベントを監視します。

```
ITMGContext.GetInstance().OnExitRoomCompleteEvent += new QAVExitRoomComplete(OnExit
```

監視処理：

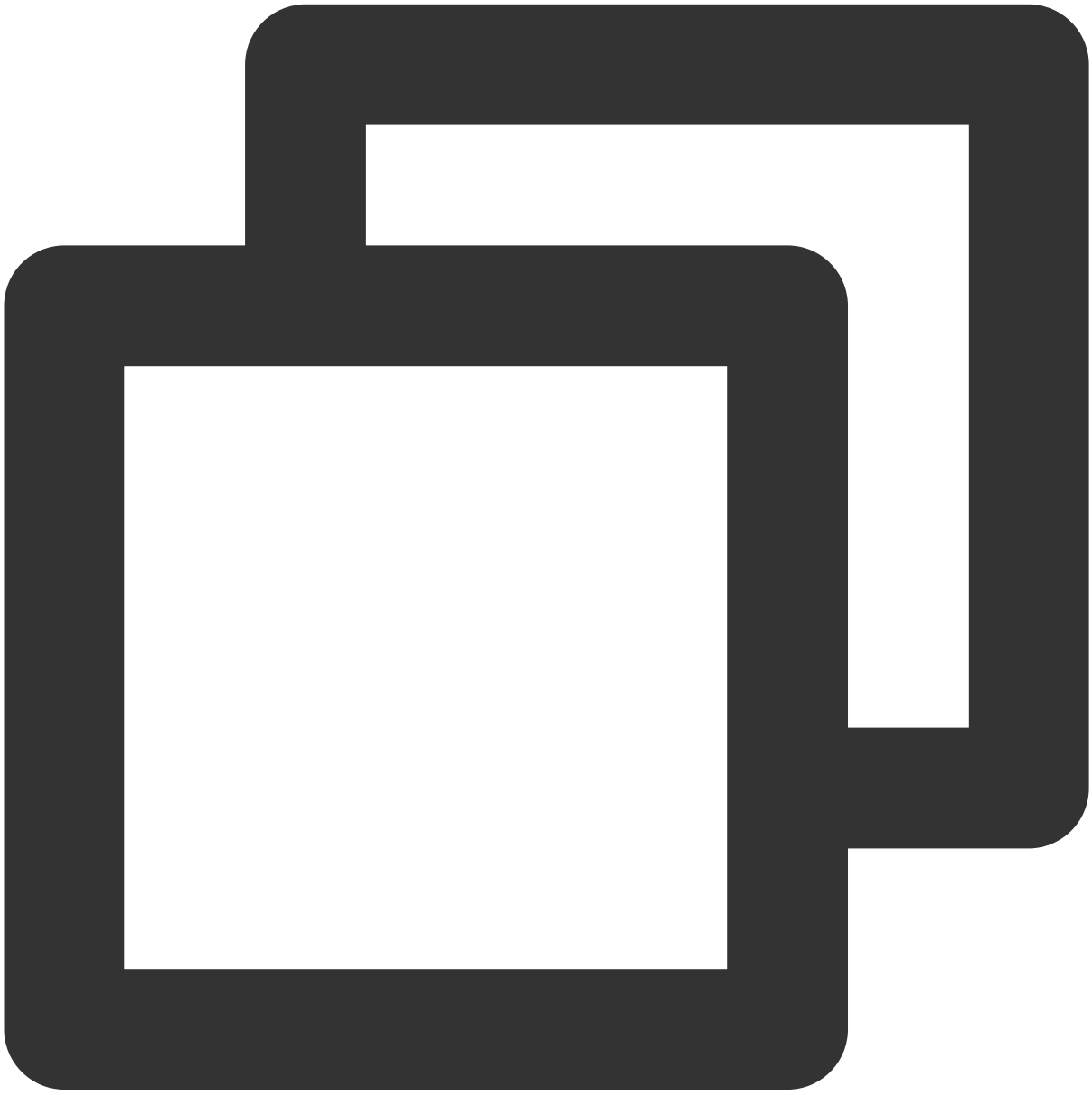
```
void OnExitRoomComplete(){
//退室した後の処理
}
```

音声メッセージの導入

1. 認証初期化

SDKを初期化してから認証の初期化を呼び出します。authBufferの取得については、前記のリアルタイム音声の認証情報インターフェースgenAuthBufferをご参照ください。

インターフェースのプロトタイプ

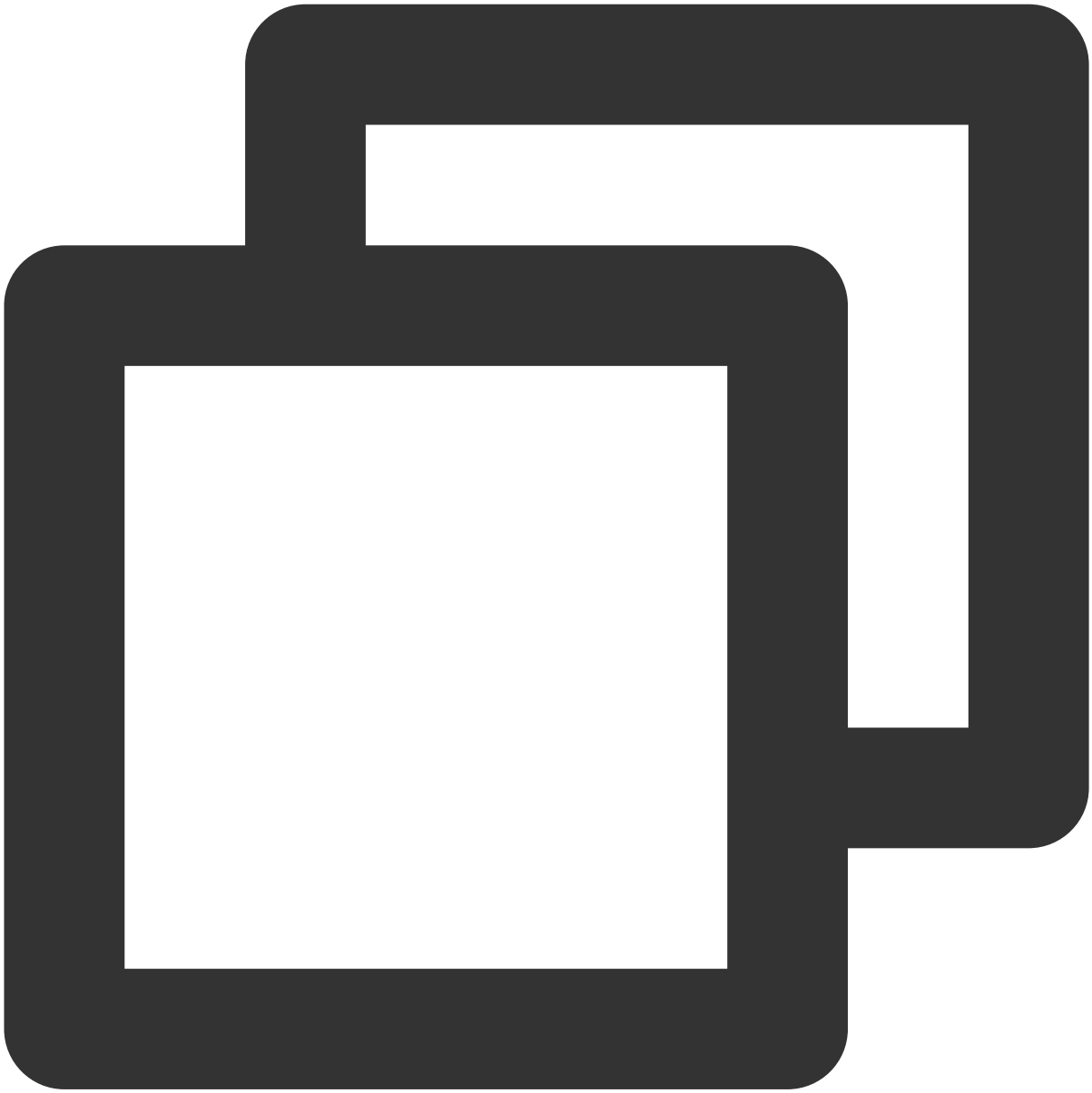


```
ITMGPTT int ApplyPTTAuthbuffer (byte[] authBuffer)
```

パラメータ	タイプ	意味

authBuffer	String	認証

サンプルコード

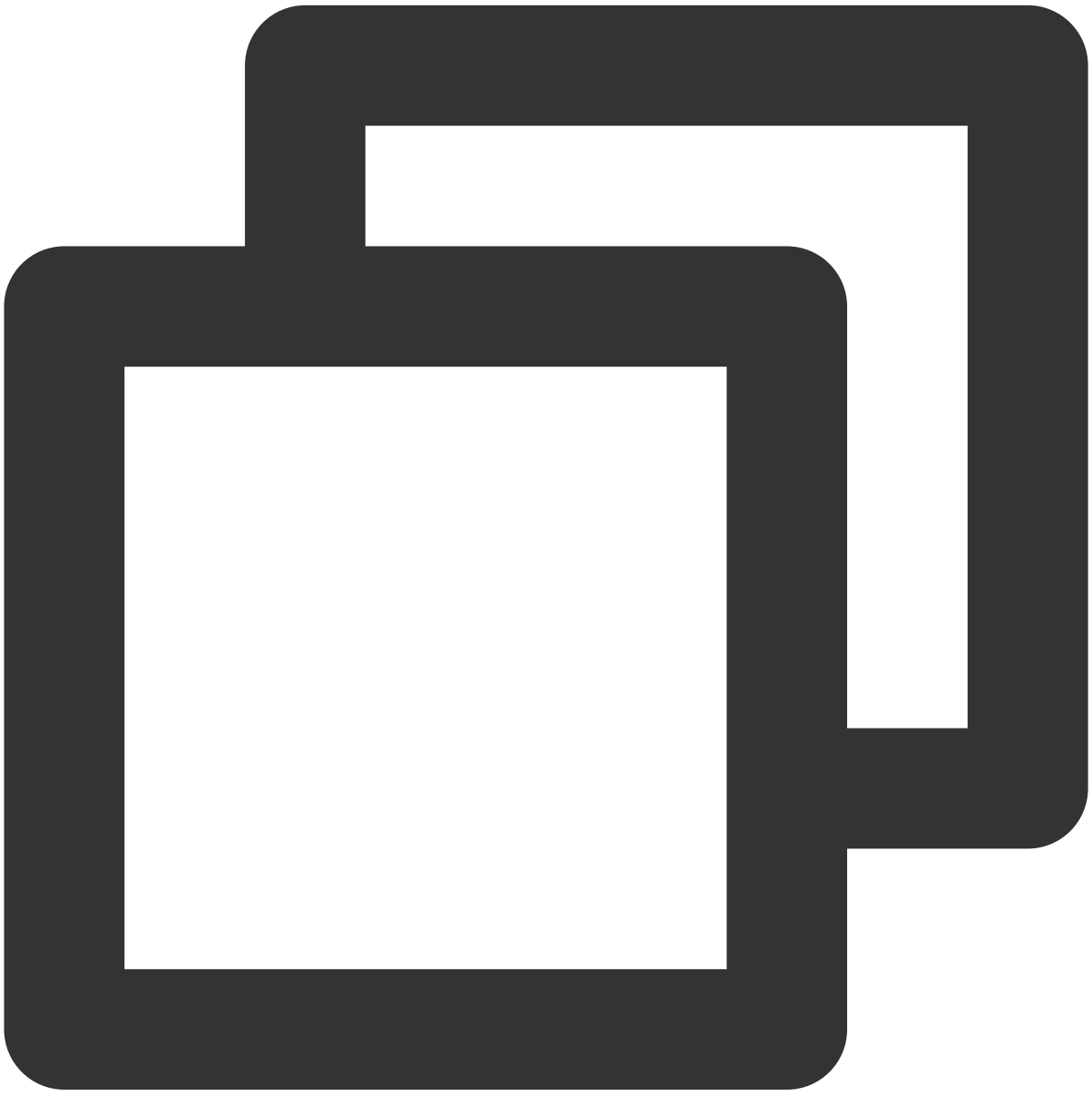


```
UserConfig.SetAppID(transform.Find ("appId").GetComponent<InputField> ().text);
UserConfig.SetUserID(transform.Find ("userId").GetComponent<InputField> ().text);
UserConfig.SetAuthKey(transform.Find("authKey").GetComponent<InputField>().text);
byte[] authBuffer = UserConfig.GetAuthBuffer(UserConfig.GetAppID(), UserConfig.GetU
ITMGContext.GetInstance().GetPttCtrl().ApplyPTTAuthbuffer(authBuffer);
```

2. ストリーミング音声認識を起動

このインターフェースは、ストリーミング音声識別の開始に使われています。コールバックにおいて、音声はリアルタイムでテキストに変換されて返されます。**録音の停止にはStopRecordingを呼び出します**。停止後にコールバックが発生します。

インターフェースのプロトタイプ

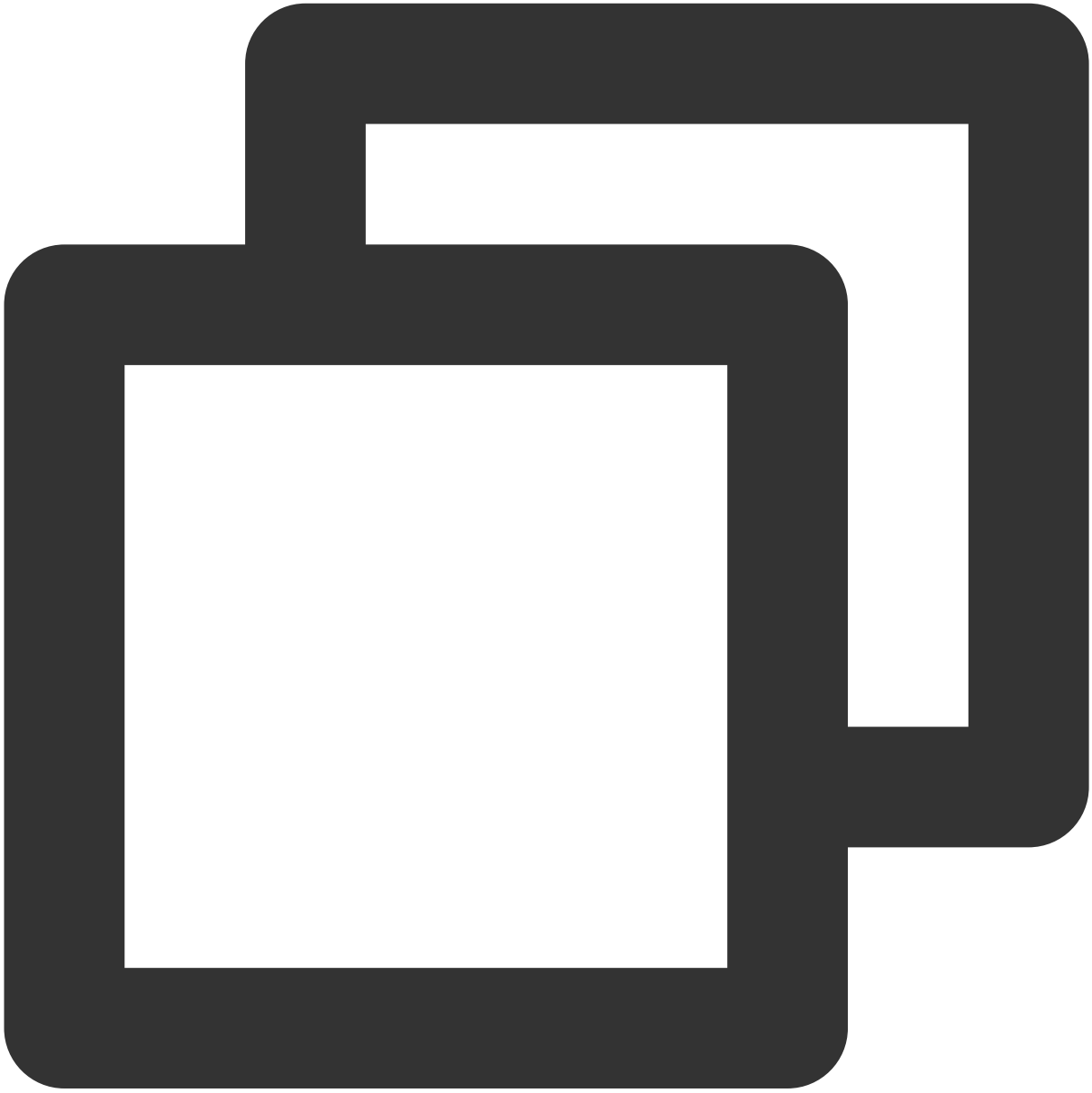


```
ITMGPTT int StartRecordingWithStreamingRecognition(string filePath)
```

パラメータ	タイプ	意味

filePath	String	ボイスの保存パス

サンプルコード



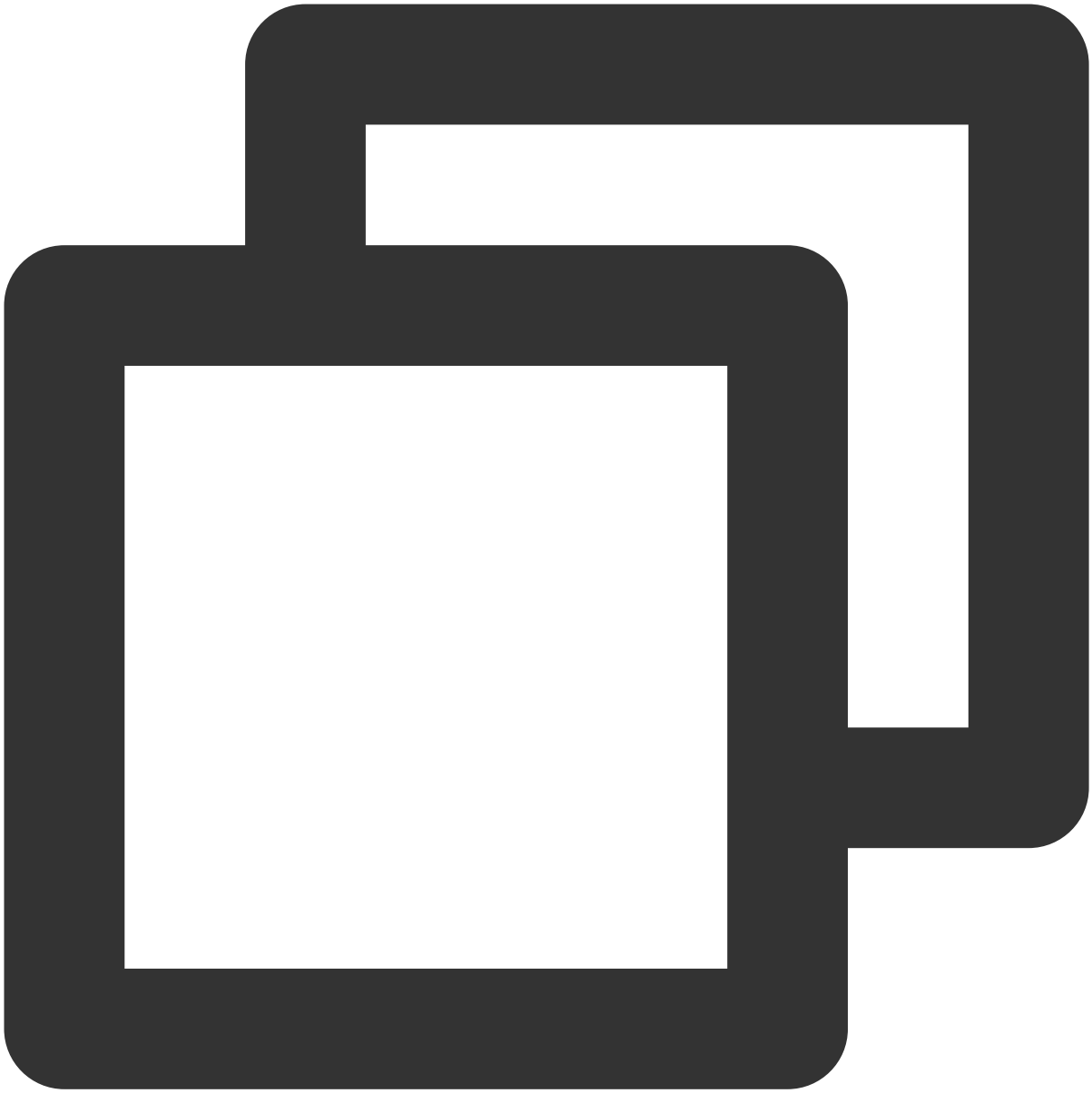
```
string recordPath = Application.persistentDataPath + string.Format("/{0}.silk", sUi
int ret = ITMGContext.GetInstance().GetPttCtrl().StartRecordingWithStreamingRecogni
```

ストリーミング音声識別コールバック

ストリーミング音声識別を開始した後、OnStreamingSpeechCompleteまたはOnStreamingSpeechisRunning通知でコールバックメッセージを監視する必要があります。イベントメッセージは次の2つがあります。
`OnStreamingSpeechComplete` はレコーディングを停止して認識が完了した後にテキストを返します。これは、発話が完了した後に認識されたテキストを返すことに相当します。
`OnStreamingSpeechisRunning` は録音中に認識されたテキストをリアルタイムで返すことであり、発話しながら認識された文字を返すことに相当します。
OnEvent関数で、必要に応じて適切なイベントメッセージを判断します。渡されるパラメータには次の4つの情報が含まれます。

メッセージ名称	意味
result	ストリーミングボイス認識が完了したかどうかを判断するための戻りコード
text	ボイステキスト変換で認識されたテキスト
file_path	録音を保存するローカルアドレス
file_id	録音はバックグラウンドのURLアドレスにあり、録音はサーバーで90日間保存されます

サンプルコード



```
//イベントを監視します：
ITMGContext.GetInstance().GetPttCtrl().OnStreamingSpeechComplete +=new QAVStreaming
ITMGContext.GetInstance().GetPttCtrl().OnStreamingSpeechisRunning += new QAVStreami
//監視処理：
void OnStreamingSpeechComplete(int code, string fileid, string filepath, string res
  // ストリーミングAutomatic Speech Recognitionのコールバックの起動
}
void OnStreamingRecisRunning(int code, string fileid, string filePath, string resul
  if (code == 0)
  {
      setBtnText(mStreamBtn, "ストリーミング");
```

```
        InputField field = transform.Find("recordFilePath").GetComponent<InputField>(
        field.text = filePath;
            field = transform.Find("downloadUrl").GetComponent<InputField>();
        field.text = "Stream is Running";
            field = transform.Find("convertTextResult").GetComponent<InputField>();
        field.text = result;
        showWarningText("レコーディング中");
    }
}
```

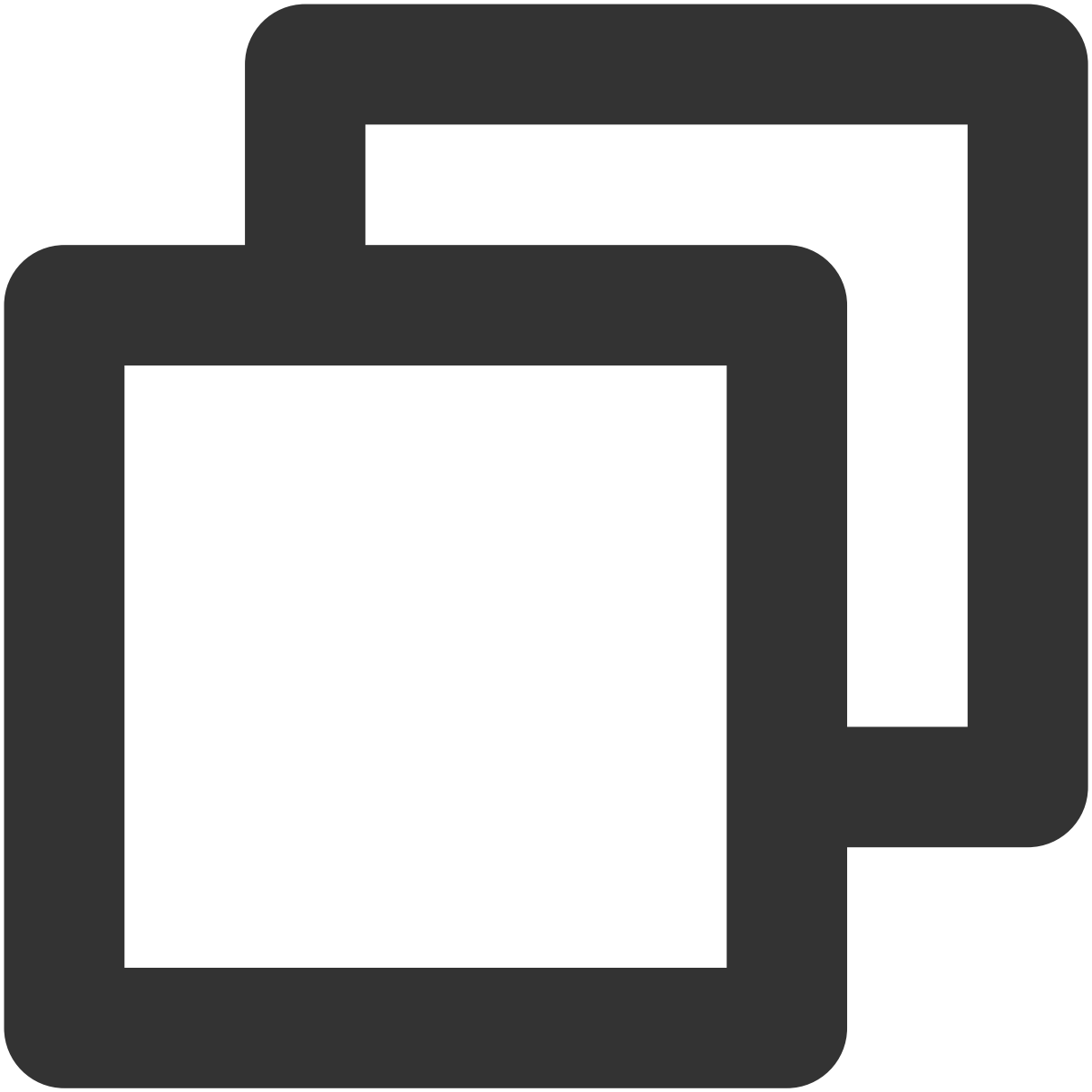
エラーコード

エラーコード	意味	処理方式
32775	ストリーミング音声をテキストに変更できませんが、録音は成功しました	UploadRecordedFileインターフェースを呼び出して録音をアップロードし、SpeechToTextインターフェースを呼び出して音声を文字に変換します
32777	ストリーミング音声をテキストに変更できませんが、録音とアップロードは成功しました	返されたメッセージには正常にアップロードしたバックグラウンドURLがあり、SpeechToTextインターフェースを呼び出して音声から文字への変換操作を行います
32786	ストリーミング音声をテキストに変更できませんでした	ストリーミングレコーディングステータスでは、ストリーミングレコーディングインターフェースの実行結果が返されるまで待ってください

3. 録音を停止

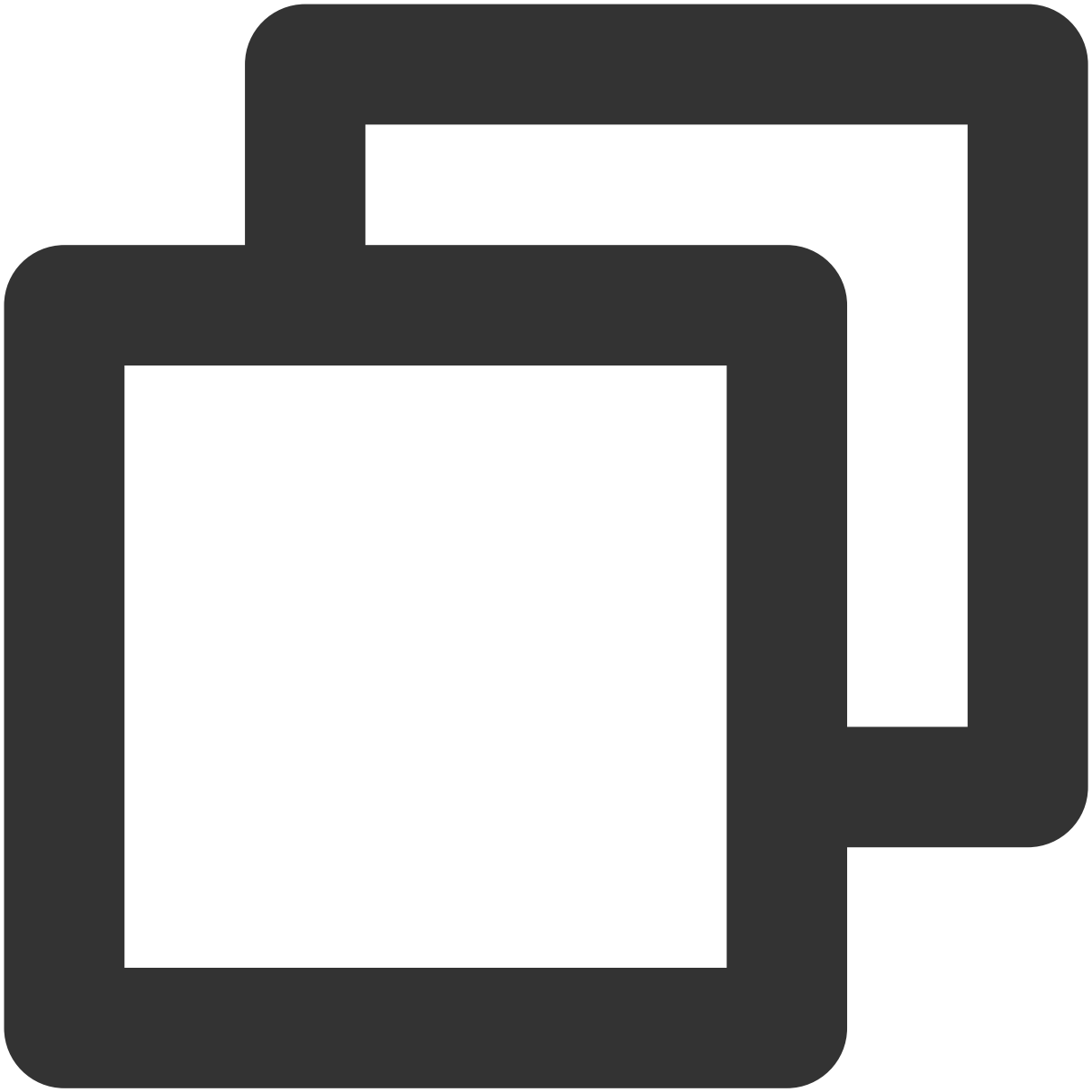
このインターフェースは、録音の停止に使われています。このインターフェースが非同期インターフェースであるため、録音を停止した後には録音完了のコールバックがあります。コールバックが成功してから、録音ファイルが利用できるようになります。

インターフェースのプロトタイプ



```
ITMGPTT int StopRecording()
```

サンプルコード



```
ITMGContext.GetInstance().GetPttCtrl().StopRecording();
```

Unreal SDKのクイック導入

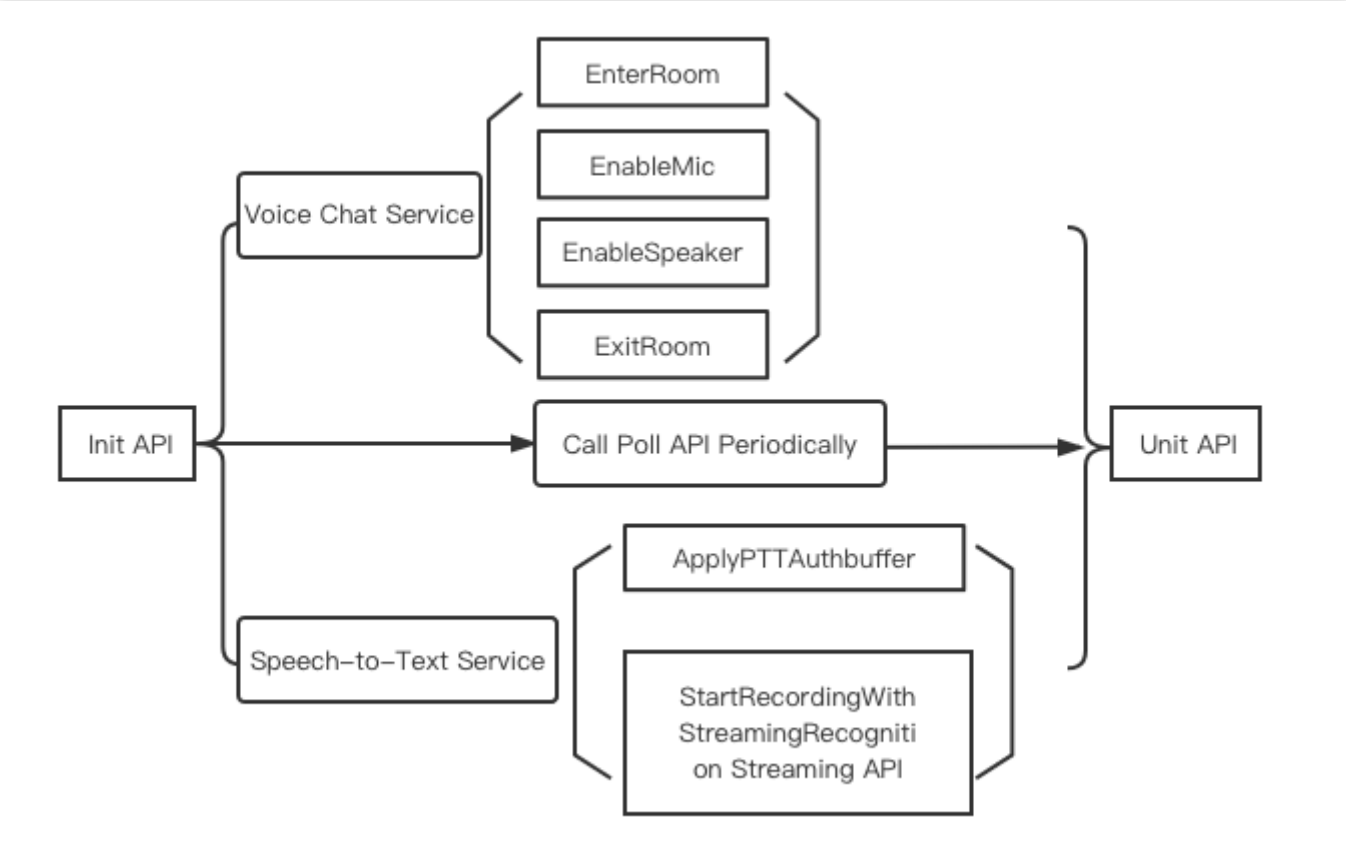
最終更新日：：2024-01-18 15:42:48

開発者がGME製品APIのデバッグアクセスを行いやすいように、ここで、Unreal Engine開発に適用されるクイックアクセスドキュメントを説明します。
GMEクイックスタート文書は、ユーザーのアクセスを助けるための最も主要なアクセスインターフェースのみを提供しています。

GME利用上の重要事項

GMEは2つの部分に分かれます。リアルタイム音声サービス、音声メッセージおよびテキスト変換サービスを提供しており、これらのサービスの利用はInitやPollなどのコアインターフェースに依存しています。
关于 Init 接口
例えば、リアルタイムの音声サービスを使用する同時に音声メッセージ・サービスも使用する場合、**Init初期化インターフェースを1回だけ呼び出す必要があります**。

インターフェース呼び出しのフローチャート



統合の手順

SDKの統合

プロジェクトにSDKを統合するには、Unreal SDK統合ドキュメントをご参照ください。

コアインターフェース

GMEの初期化，接口：Init
定期的なPoll呼び出しによるコールバックのトリガー，接口：Poll
入退室の通知監視，监听：QAVEnterRoomComplete

リアルタイム音声

1. リアルタイム音声ルームへの参加，接口：EnterRoom
2. マイクのオン/オフ，接口：EnableMic
3. スピーカーのオン/オフ，接口：EnableSpeaker
4. 音声ルーム退出，接口：ExitRoom

音声メッセージ

1. 認証の初期化，接口：ApplyPTTAuthbuffer
2. ストリーミング音声認識の起動，接口：StartRecordingWithStreamingRecognition

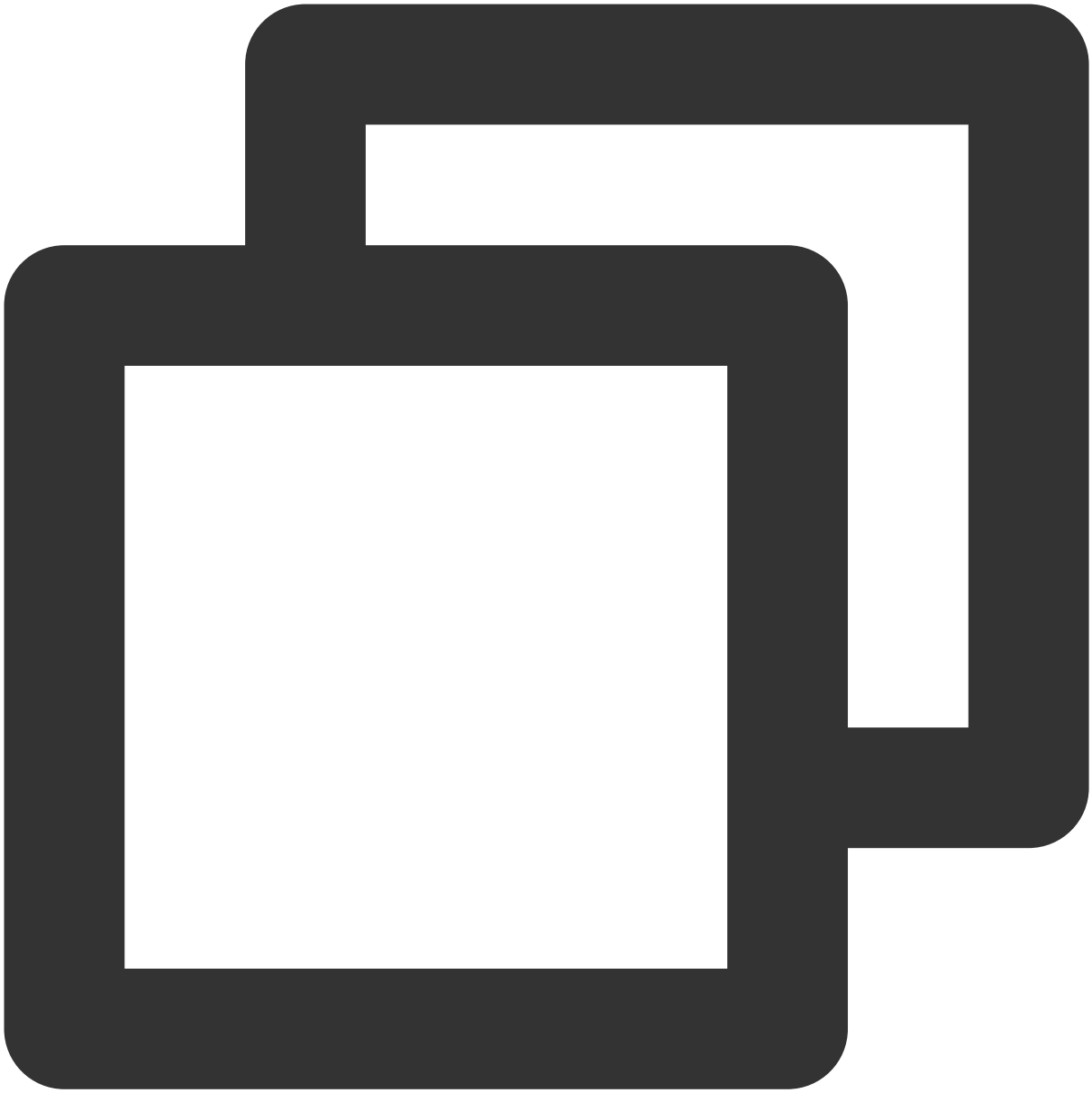
3. レコーディング停止，接口：StopRecording
GMEの録画を停止，接口：UnInit

コアインターフェースのアクセス

1. SDKのダウンロード

ダウンロード案内ページにアクセスして、必要なクライアンSDKをダウンロードします。

2. ヘッダーファイルを取り込む

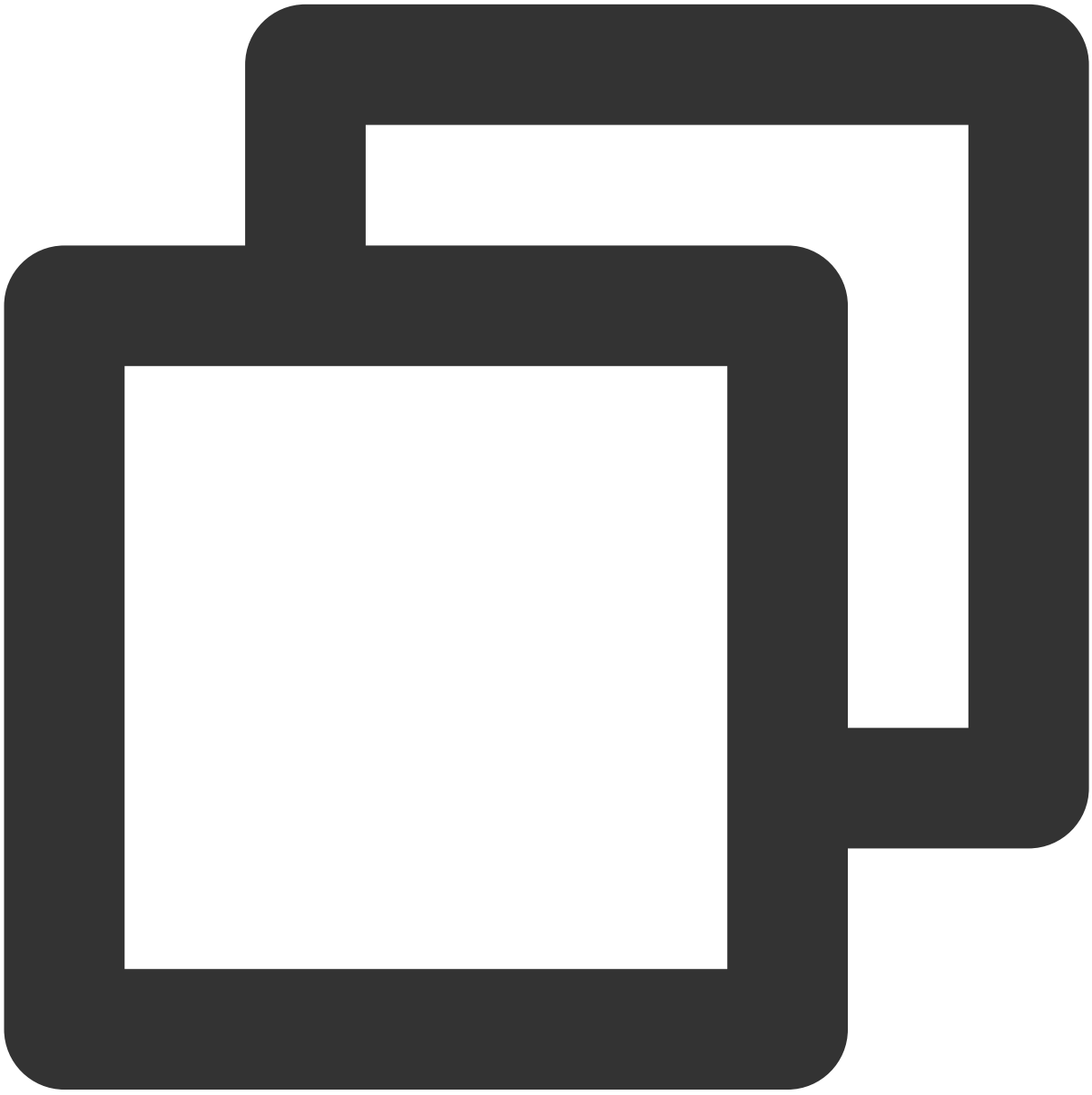


```
#include "tmg_sdk.h"

class UEDEMO1_API AUEDemoLevelScriptActor : public ALevelScriptActor, public ITMGDe
{
public:
...
private:
...
}
```

3. シングルトンの設定

EnterRoom関数を呼び出す前に、ITMGContextの取得が必要です、全ての呼び出しはITMGContextから始まっており、ITMGDelegateによってコールバックでアプリケーションに返送されています、事前に設定してください。

サンプルコード

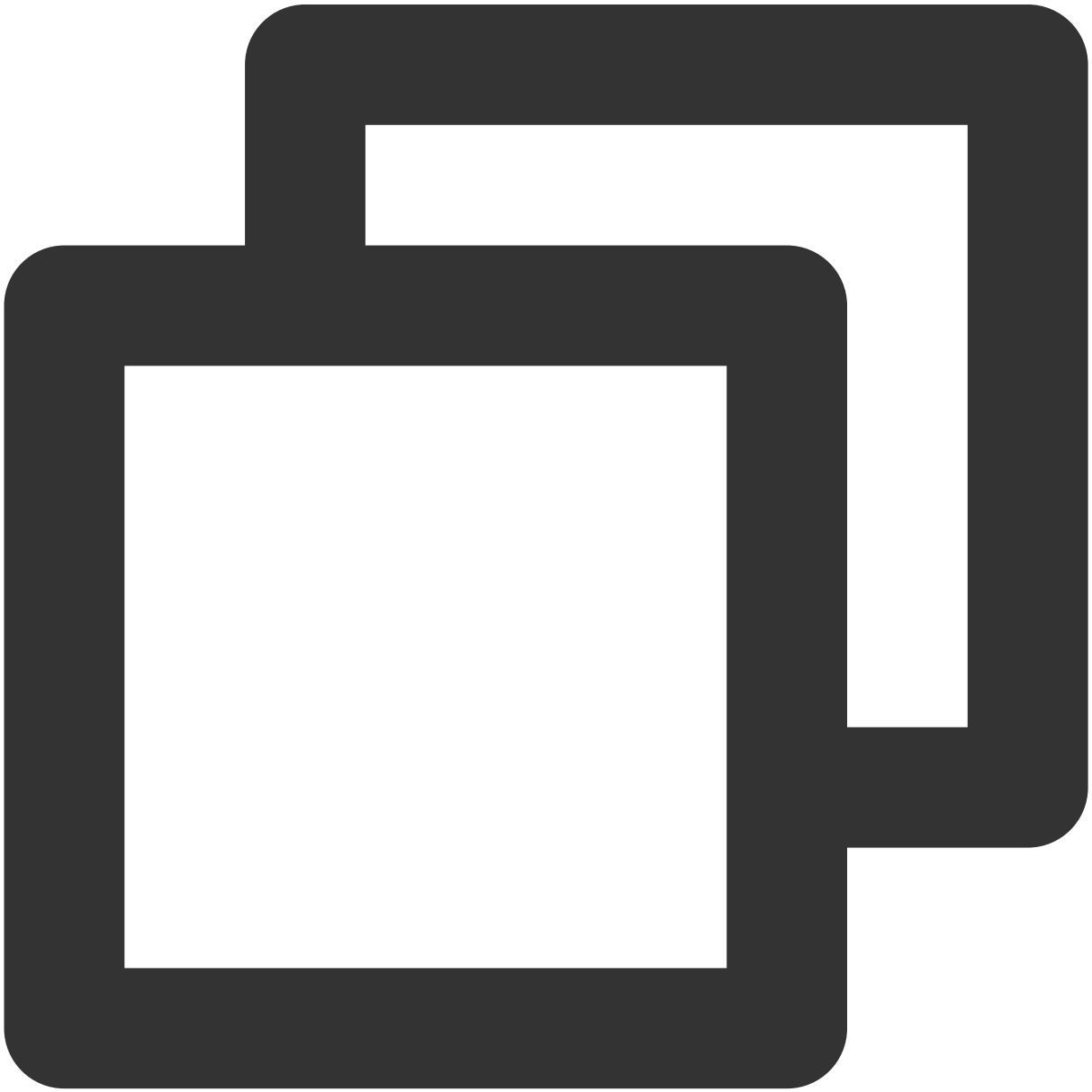


```
ITMGContext* context = ITMGContextGetInstance();
context->SetTMGDelegate(this);
```

4. SDKを初期化する

このインターフェースはGMEサービスの初期化に使用され、アプリケーションの初期化時にアプリケーション側で呼び出すことをお勧めします。
OpenIdはユーザーを一意に識別するために使用されます。現在はINT64のみがサポートされています。ルールはApp開発者が独自に設定し、App内で重複しないようにしてください。
ユーザーがログインアカウントを切り替えた場合、Uninitを呼び出してから新しいOpenIdを使用してGMEサービスを再Initしてください。

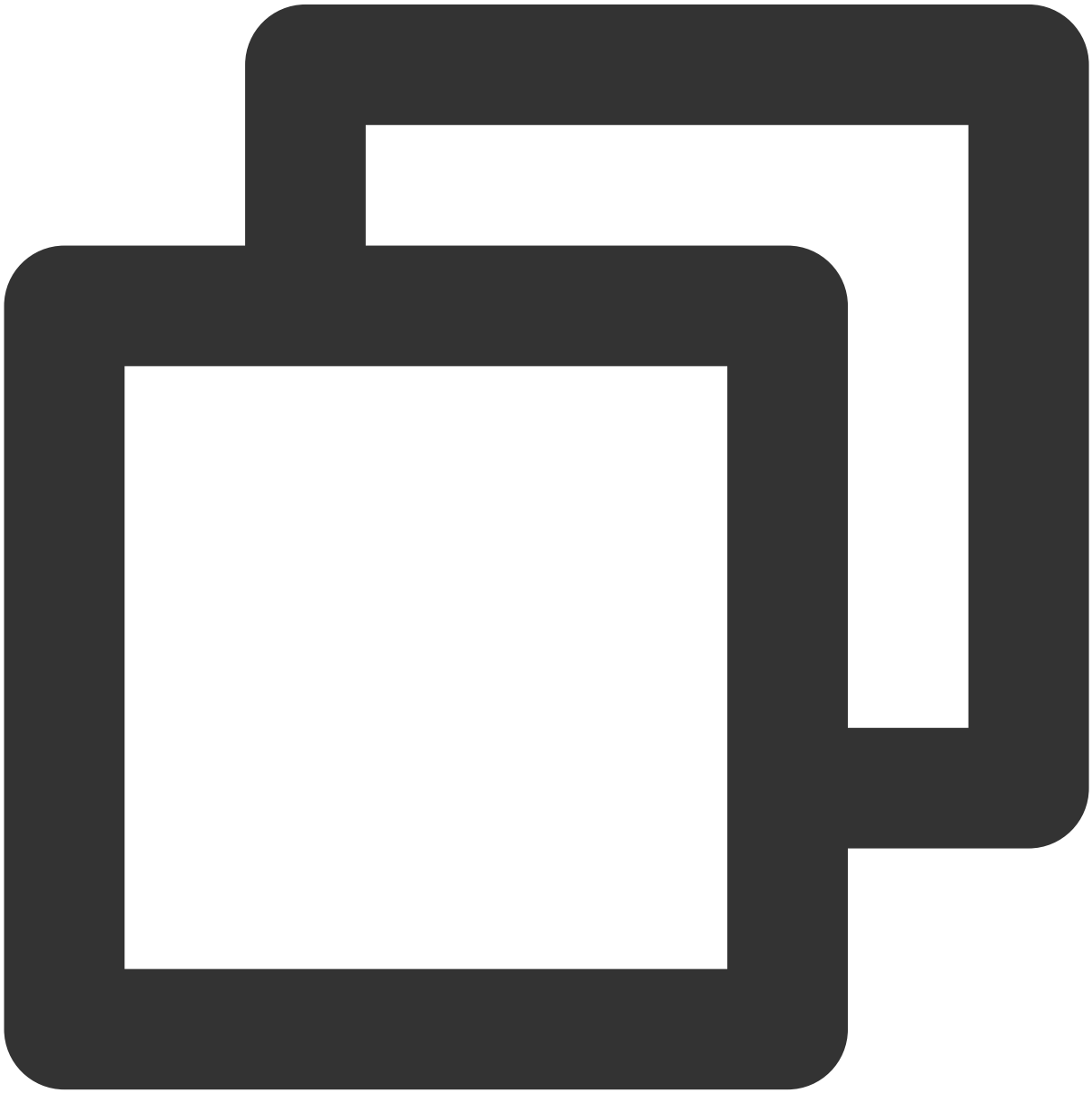
関数のプロトタイプ



```
//class ITMGContext
ITMGContext virtual int Init(const char* sdkAppId, const char* openId)
```

パラメータ	タイプ	意味
sdkAppId	const char*	Tencent CloudコンソールのGME サービスが提供するAppId。
OpenId	const char*	OpenIdはInt64型のみをサポートします（stringに変換されて渡されます）。

サンプルコード

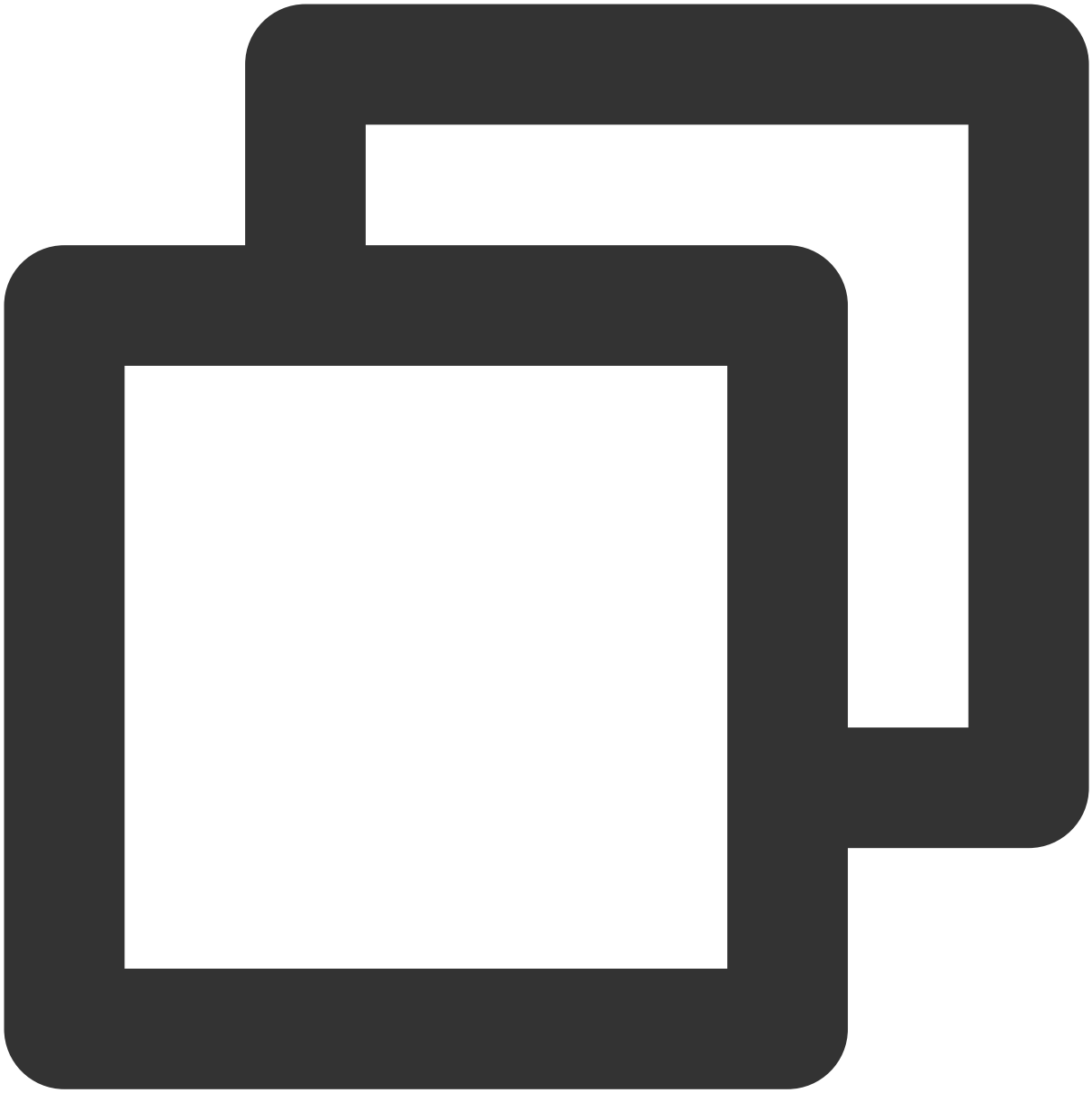


```
std::string appid = TCHAR_TO_UTF8(CurrentWidget->editAppID->GetText().ToString().op
std::string userId = TCHAR_TO_UTF8(CurrentWidget->editUserID->GetText().ToString().
ITMGContextGetInstance()->Init(appid.c_str(), userId.c_str());
```

5. イベントコールバックのトリガー

updateで周期的にPollを呼び出すことで、イベントのコールバックをトリガできます。GMEは定期的にPoll APIを呼び出してイベントコールバックをトリガしてください。Pollが呼び出されないと、SDKサービス全体が異常に動作します。
詳細については、DemoのUEDemoLevelScriptActor.cppファイルをご参照ください。

サンプルコード

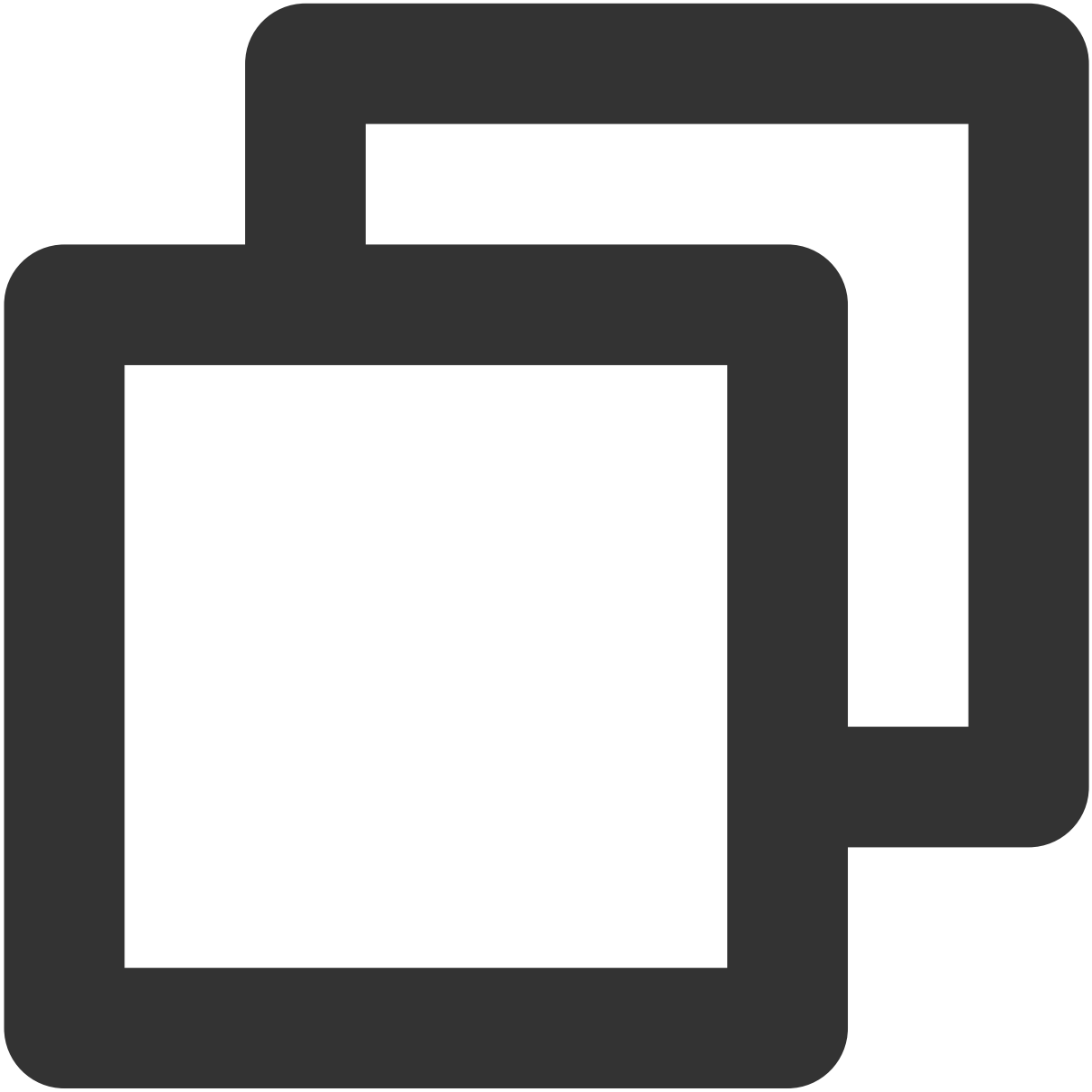


```
//ヘッダファイルにおける宣言
virtual void Tick(float DeltaSeconds);

void AUEDemoLevelScriptActor::Tick(float DeltaSeconds) {
  Super::Tick(DeltaSeconds);
  ITMGContextGetInstance()->Poll();
}
```

6. コールバックの設定

インターフェースなどは、Delegate方式でアプリケーションにコールバック通知を送信します、メッセージタイプはITMG_MAIN_EVENT_TYPEを参考してください、Windowsプラットフォームにおけるdataはjson文字列の形であり、具体的なkey-valueは、説明ドキュメントを参考してください。

サンプルコード



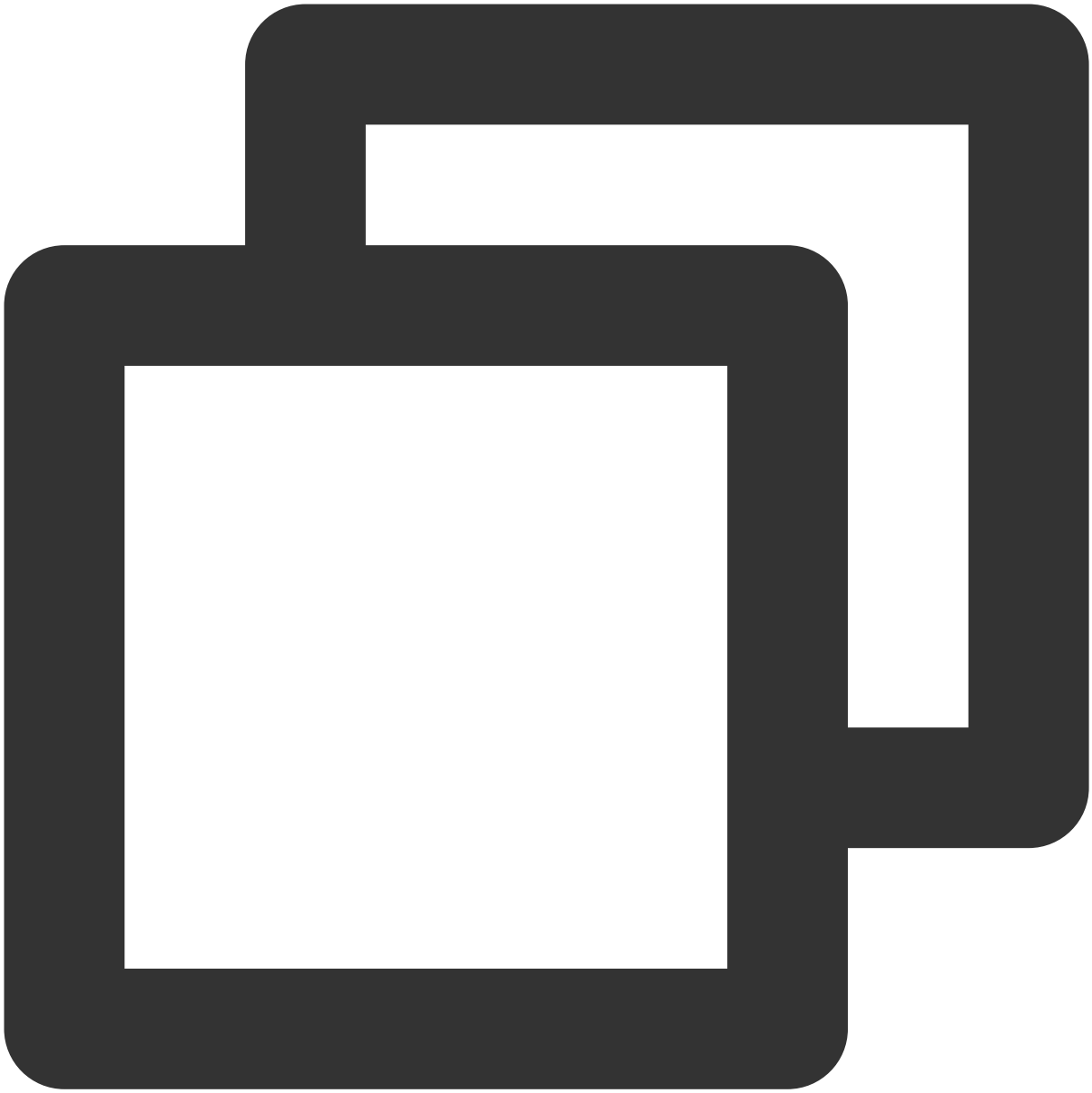
```
//関数の実現：
//UEDemoLevelScriptActor.h:
class UEDEMO1_API AUEDemoLevelScriptActor : public ALevelScriptActor, public SetTMG
{
public:
```

```
  void OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char* data);
}

//UEDemoLevelScriptActor.cpp:
void AUEDemoLevelScriptActor::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char* d
  //ここで、eventTypeの判断と操作を行います
}
```

7. 認証情報

関連する機能の暗号化と認証のためのAuthBufferを生成します。音声メッセージやテキスト変換の認証を取得する場合、ルーム番号のパラメータをnullに設定してください。

関数のプロトタイプ

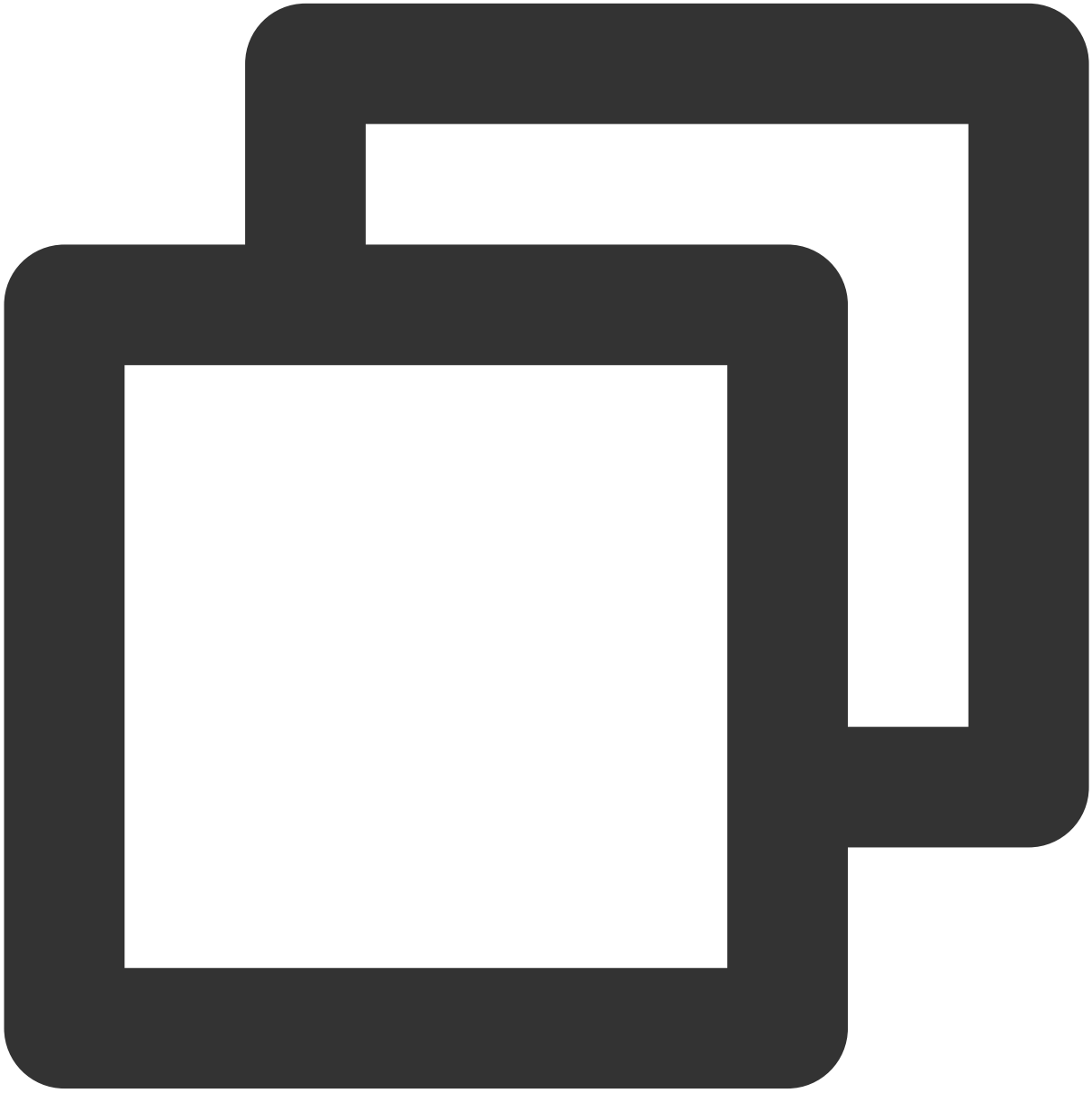


```
int  QAVSDK_AuthBuffer_GenAuthBuffer(unsigned int dwSdkAppID, const char* strRoomID
  const char* strKey, unsigned char* strAuthBuffer, unsigned int bufferLength);
```

参数	タイプ	意味
dwSdkAppID	int	Tencent Cloudコンソールからの AppId番号
strRoomID	char*	ルーム番号です。最大127バイトまでをサポートします
strOpenID	char*	ユーザーID。Initの場合のopenIDと同じです。

strKey	char*	Tencent Cloudコンソールからの権限キー
strAuthBuffer	char*	返されたauthbuff
bufferLength	int	渡されるauthbuffの長さです。推奨値は500です

サンプルコード



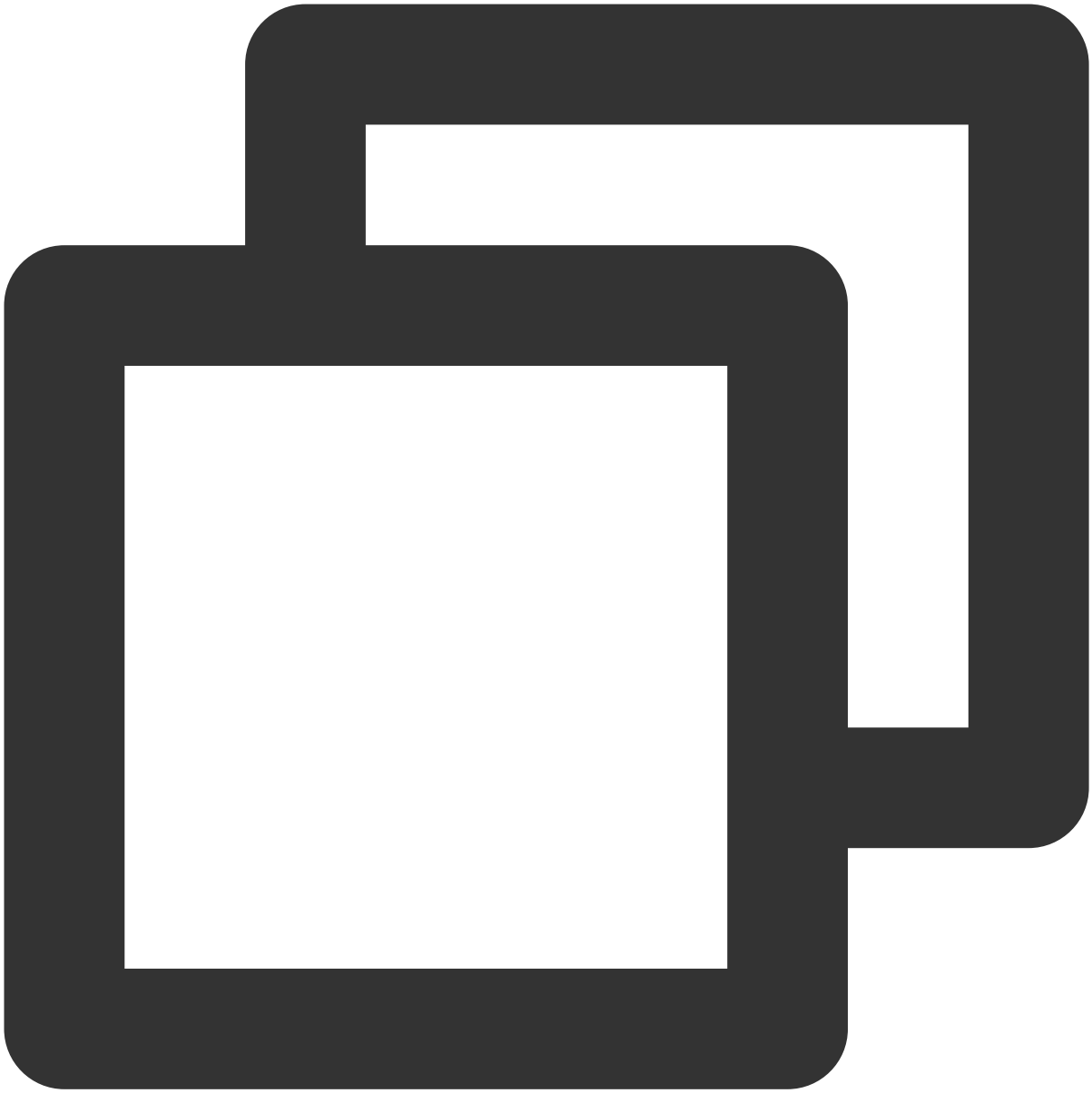
```
unsigned int bufferLen = 512;
unsigned char retAuthBuff[512] = {0};
QAVSDK_AuthBuffer_GenAuthBuffer(atoi(SDKAPPID3RD), roomId, "10001", AUTHKEY,retAuth
```

リアルタイム音声アクセス

1. ルームに参加

生成された認証情報を用いて入室します。ルームに参加するとき、デフォルトでマイクとスピーカーはオフです。インターフェースの戻り値が0の場合は、正常に入室することでなく、呼び出しインタフェースが正常に終了したことを意味します。
ルームオーディオタイプについては、音質選択をご参照ください。

関数のプロトタイプ

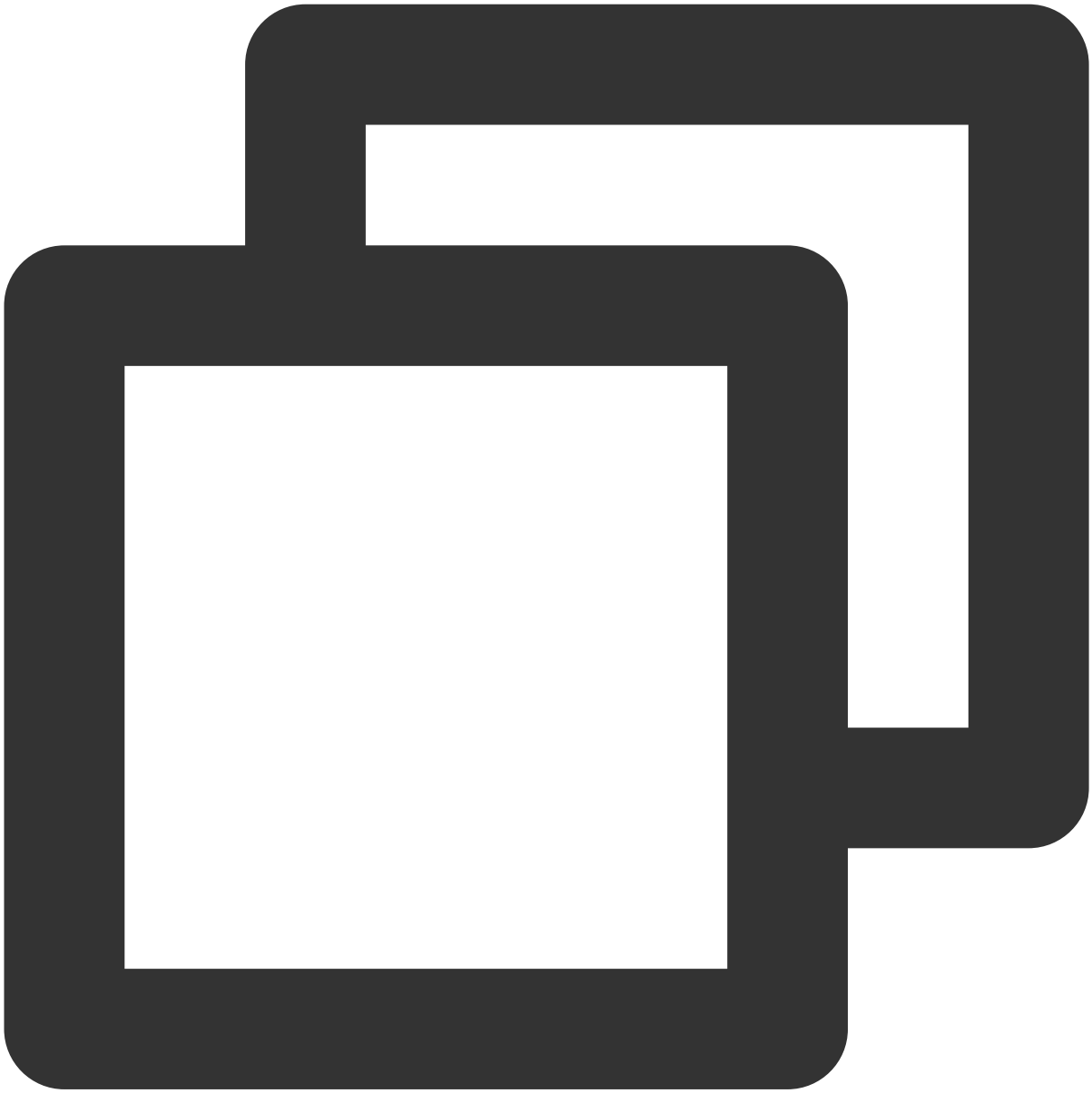


```
ITMGContext virtual int EnterRoom(const char*  roomID, ITMG_ROOM_TYPE roomType, con
```

パラメータ	タイプ	意味
roomID	char*	ルーム番号であり、最大127バイトまでをサポートします
roomType	ITMG_ROOM_TYPE	ルームオーディオタイプ
authBuffer	char*	認証コード

buffLen	int	認証コードの長さ

サンプルコード



```
ITMGContext* context = ITMGContextGetInstance();
context->EnterRoom(roomID, ITMG_ROOM_TYPE_FLUENCY, (char*)retAuthBuff,bufferLen);
```

入室イベントのコールバック

入室が完了すると入室通知が届き、監視処理関数で判断して処理します。err=0のコールバックは成功を意味し、つまりこの時点の入室が成功しました。**課金**が開始されます。本日の合計通話時間が700分未満の場合は無料です。

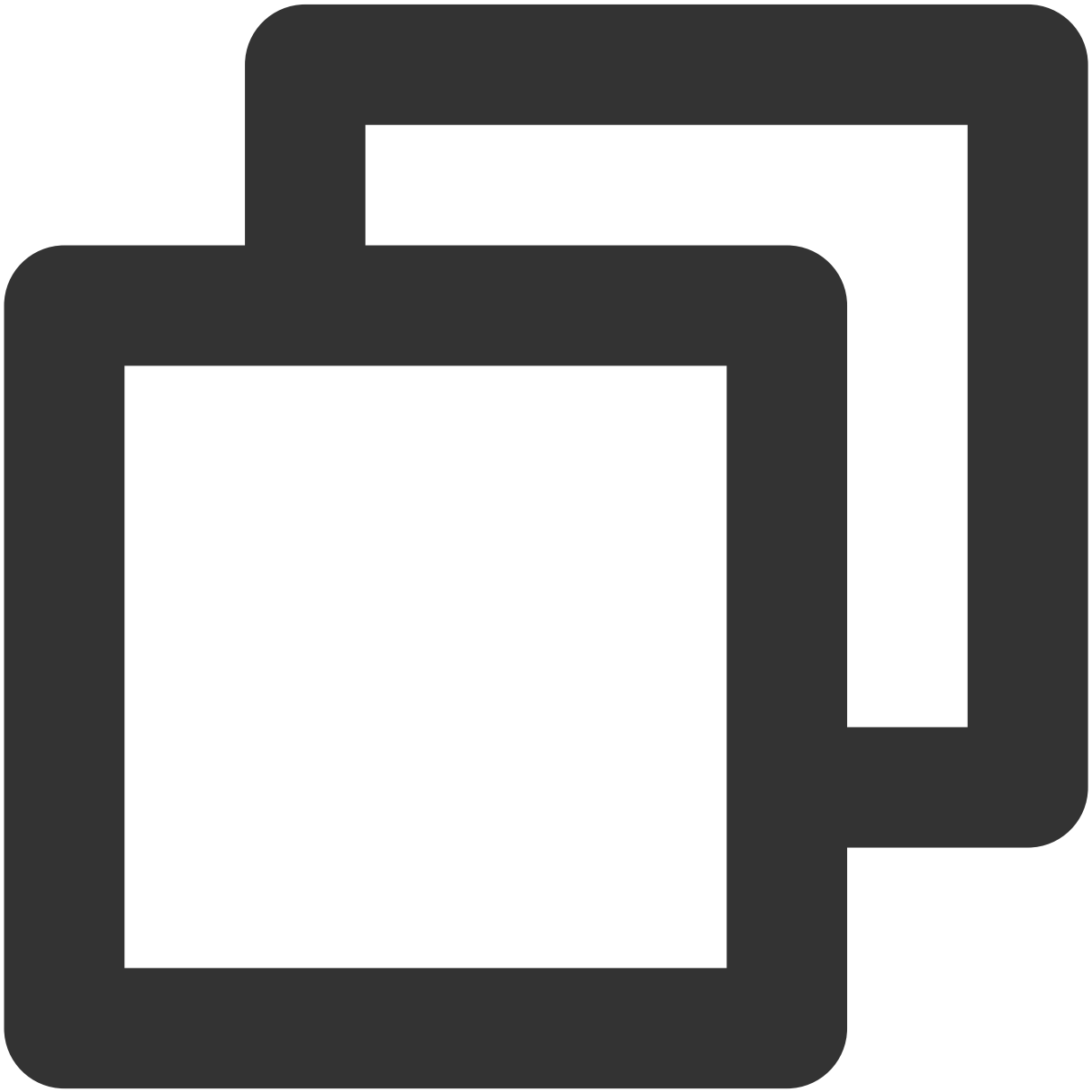
计费问题参考

購入ガイド。

課金に関するよくあるご質問。

リアルタイム音声を使用した後、クライアントの接続が切れた場合課金は継続されますか？

サンプルコードコールバック処理の関連参照コード



```
void UBaseViewController::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char *data)
```

```
FString jsonData = FString(UTF8_TO_TCHAR(data));
TSharedPtr<FJsonObject> JsonObject;
TSharedRef<TJsonReader<>> Reader = TJsonReaderFactory<>::Create(FString(UTF8_TO_TCH
FJsonSerializer::Deserialize(Reader, JsonObject);


if (eventType == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM) {
  int32 result = JsonObject->GetIntegerField(TEXT("result"));
  FString error_info = JsonObject->GetStringField(TEXT("error_info"));
  if (result == 0) {
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, TEXT("Enter
  }
  else {
    FString msg = FString::Printf(TEXT("Enter room failed. result=%d, info = %ls"),
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, *msg);
  }
  onEnterRoomCompleted(result, error_info);
}
}
```

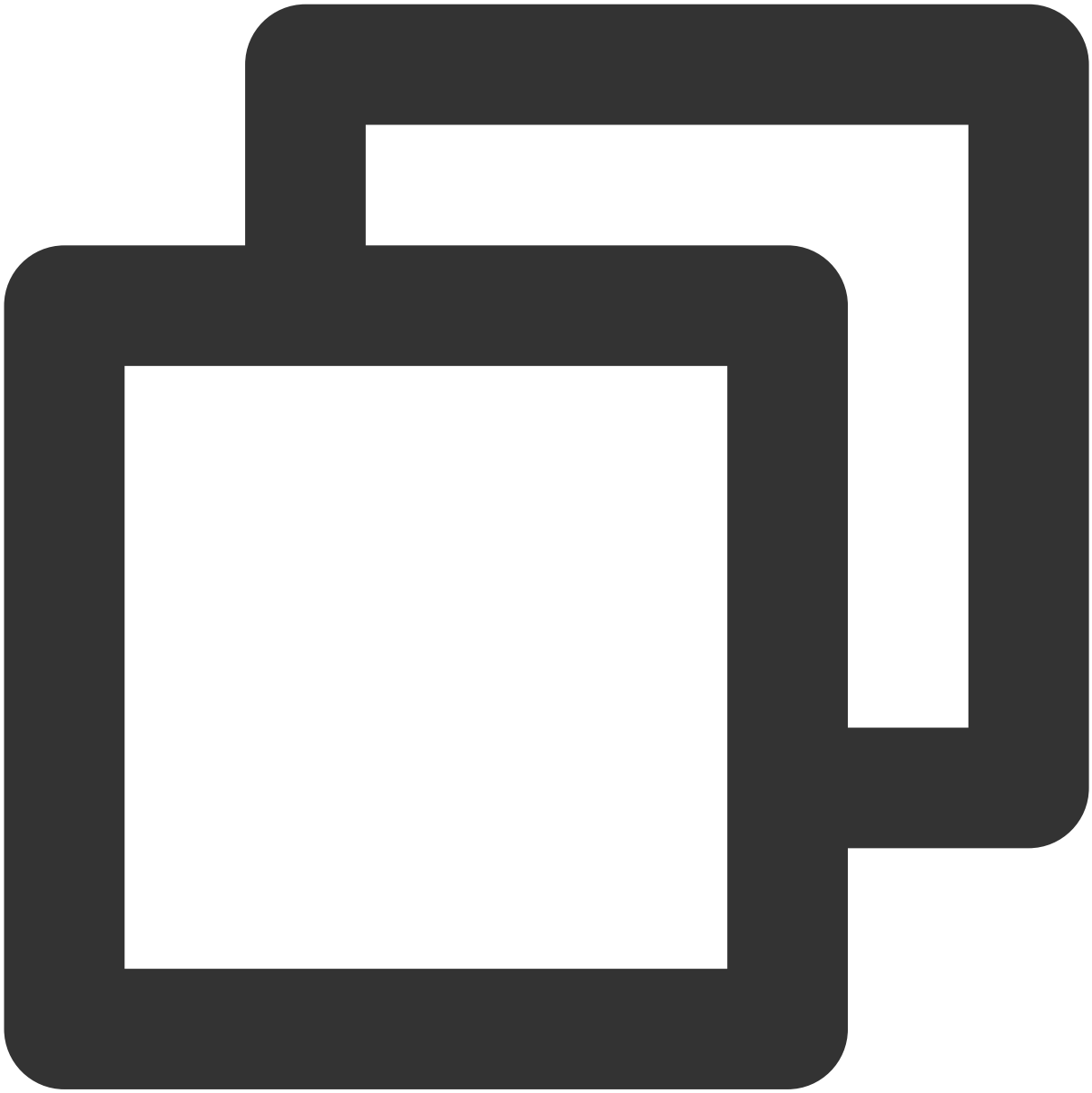
エラーコード

エラーコードの値	原因と解決策
7006	次の理由で認証に失敗しました： AppIDが存在しないか、エラーです authbuff認証エラーです 認証期限切れです openIdが仕様に準拠していません
7007	他のルームにいます
1001	ルーム参加中でこの操作を繰り返しています。コールバックが戻るまで、ルーム参加インターフェースを呼び出さないことをお勧めします
1003	ルームに参加してルームにいますが、もう1回ルーム参加インターフェースを呼び出しました
1101	SDKが初期化されていること、openIdが規則に準拠していること、またはインターフェースが同じスレッドで呼び出されていること、およびPollインタフェースが正常に呼び出されていることを確認してください

2. マイクのオン/オフ

このインターフェースは、マイクのオン/オフに使用されます。入室する際、マイクとスピーカーはデフォルトでオフになっています。

サンプルコード



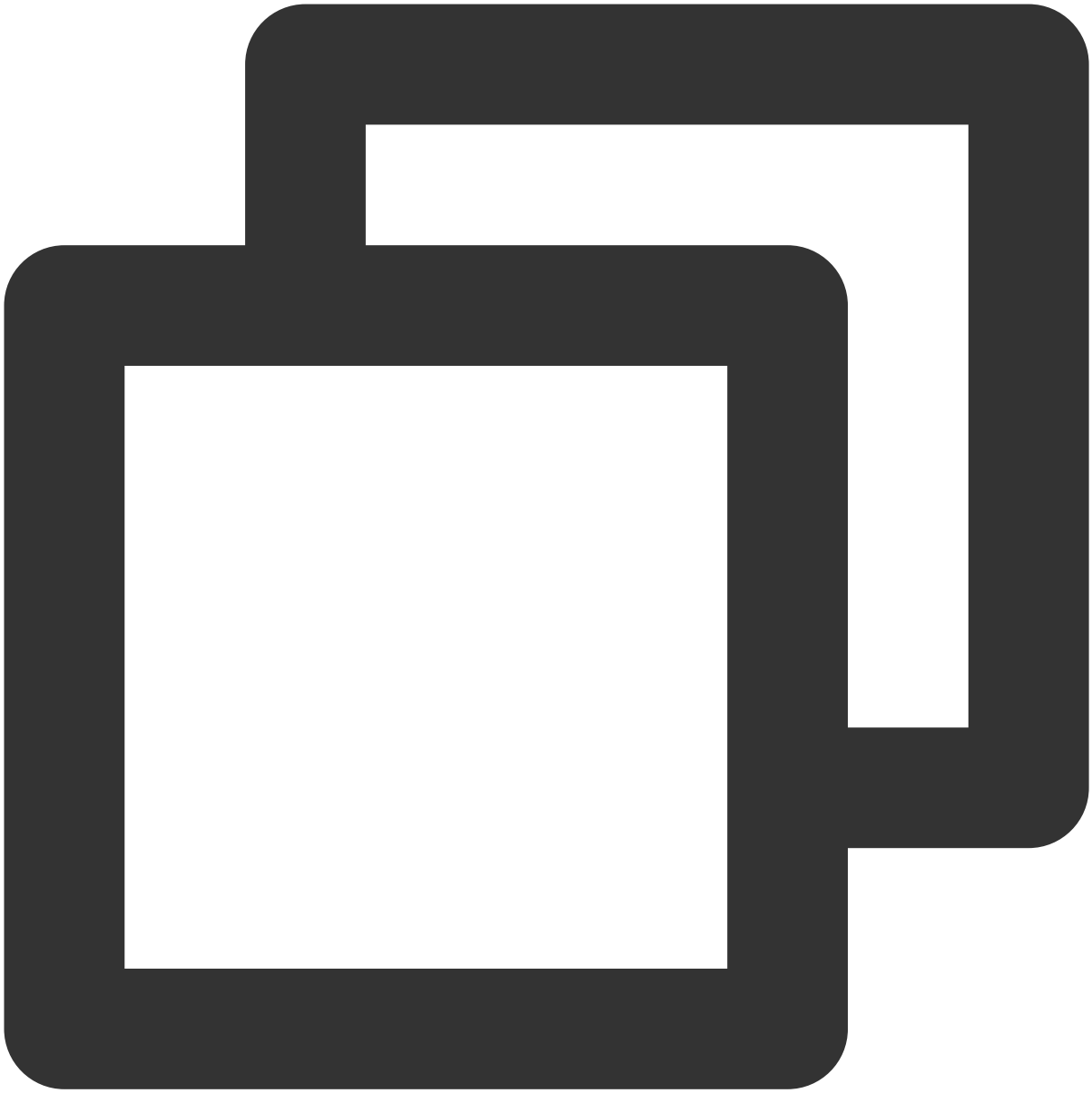
```
void UBaseViewController::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char *data)

  FString jsonData = FString(UTF8_TO_TCHAR(data));
  TSharedPtr<FJsonObject> JsonObject;
  TSharedRef<TJsonReader<>> Reader = TJsonReaderFactory<>::Create(FString(UTF8_TO_T
  FJsonSerializer::Deserialize(Reader, JsonObject);
```

```
  if (eventType == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM) {
    int32 result = JsonObject->GetIntegerField(TEXT("result"));
    FString error_info = JsonObject->GetStringField(TEXT("error_info"));
    if (result == 0) {
      GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, TEXT("Ent
      // マイクの起動
      ITMGContextGetInstance()->GetAudioCtrl()->EnableMic(true);
    }
    else {
      FString msg = FString::Printf(TEXT("Enter room failed. result=%d, info = %ls"
      GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, *msg);
    }
    onEnterRoomCompleted(result, error_info);
  }
}
```

3. スピーカーのオン/オフ

このインターフェースは、スピーカーのオン/オフに使用されます。

サンプルコード



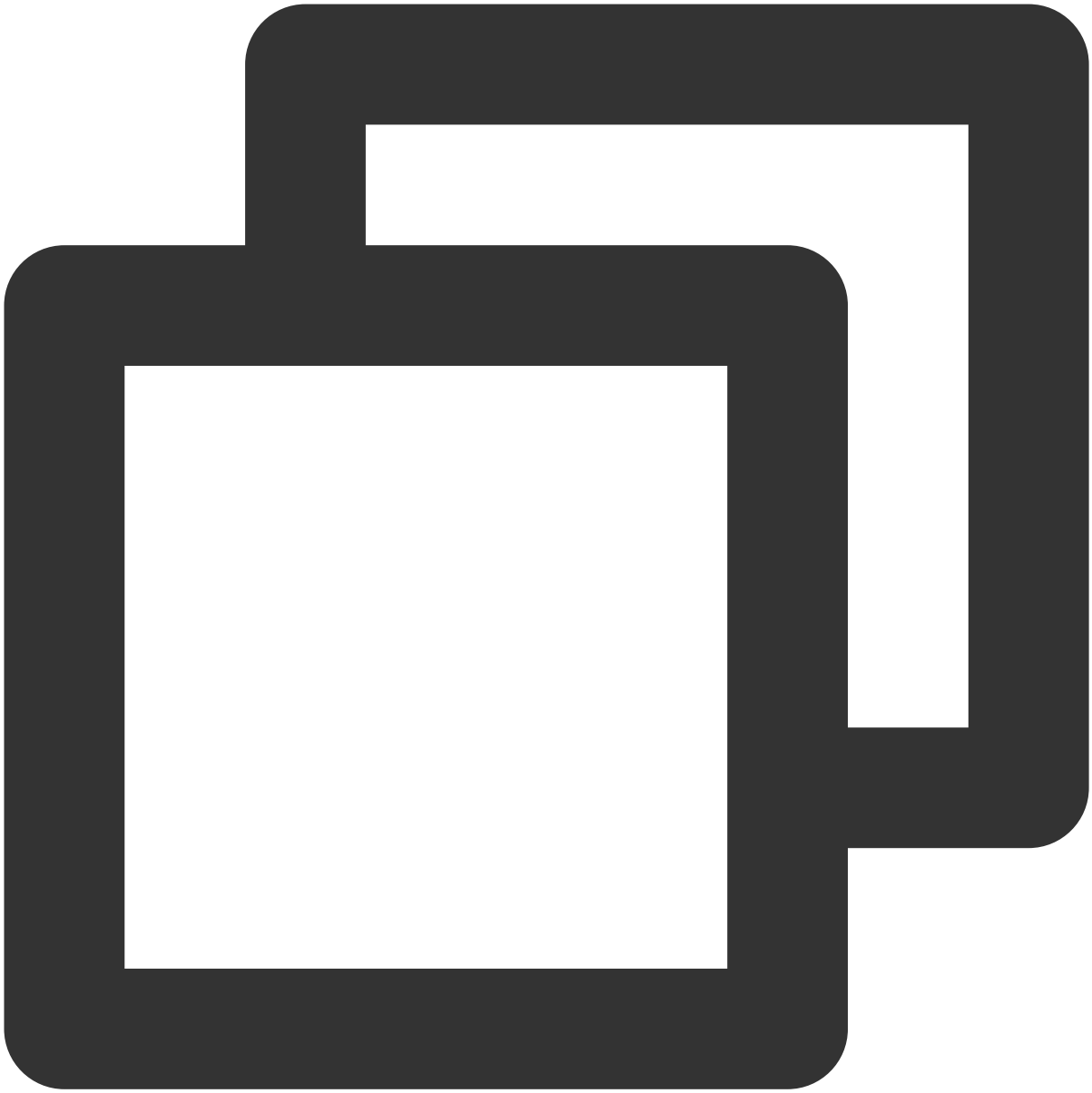
```
void UBaseViewController::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char *data)

  FString jsonData = FString(UTF8_TO_TCHAR(data));
  TSharedPtr<FJsonObject> JsonObject;
  TSharedRef<TJsonReader<>> Reader = TJsonReaderFactory<>::Create(FString(UTF8_TO_T
  FJsonSerializer::Deserialize(Reader, JsonObject);


  if (eventType == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM) {
    int32 result = JsonObject->GetIntegerField(TEXT("result"));
    FString error_info = JsonObject->GetStringField(TEXT("error_info"));
```

```
    if (result == 0) {
      GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, TEXT("Ent
      //スピーカーをオンにする
      ITMGContextGetInstance()->GetAudioCtrl()->EnableSpeaker(true);
    }
    else {
      FString msg = FString::Printf(TEXT("Enter room failed. result=%d, info = %ls"
      GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, *msg);
    }
    onEnterRoomCompleted(result, error_info);
  }
}
```

4. 退室

このインターフェースを呼び出すと、退室することができます。処理の実行は退室のコールバックを待つ必要があります。

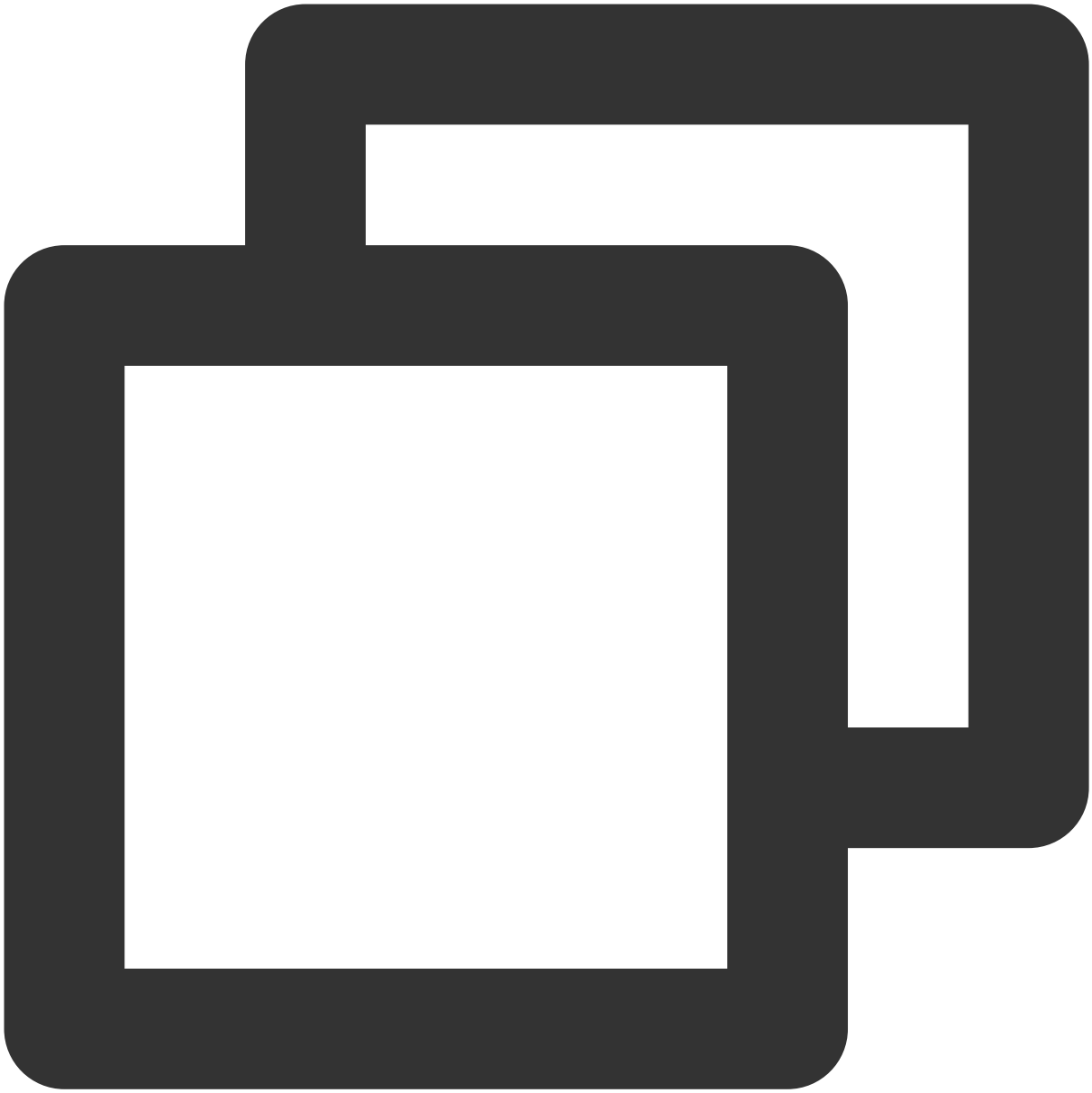
サンプルコード



```
ITMGContext* context = ITMGContextGetInstance();
context->ExitRoom();
```

退室コールバック

退室してからはコールバックが発生します、サンプルコードは次の通りです：



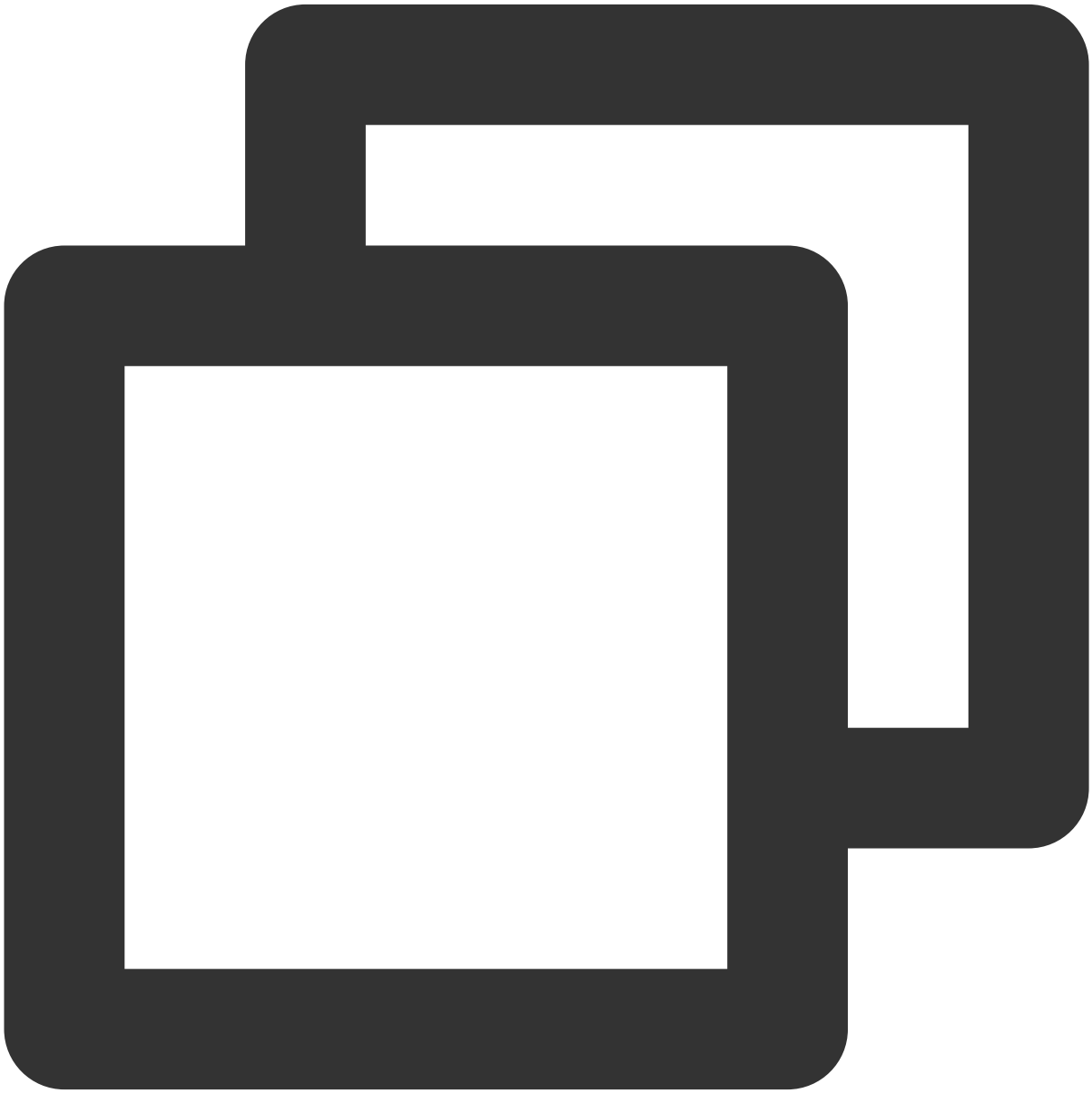
```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
  switch (eventType) {
            case ITMG_MAIN_EVENT_TYPE_EXIT_ROOM:
    {
    //処理します
    break;
    }
  }
}
```

音声メッセージの導入

1. 認証初期化

SDKを初期化してから認証の初期化を呼び出します。authBufferの取得については、前記のリアルタイム音声の認証情報インターフェースgenAuthBufferをご参照ください。

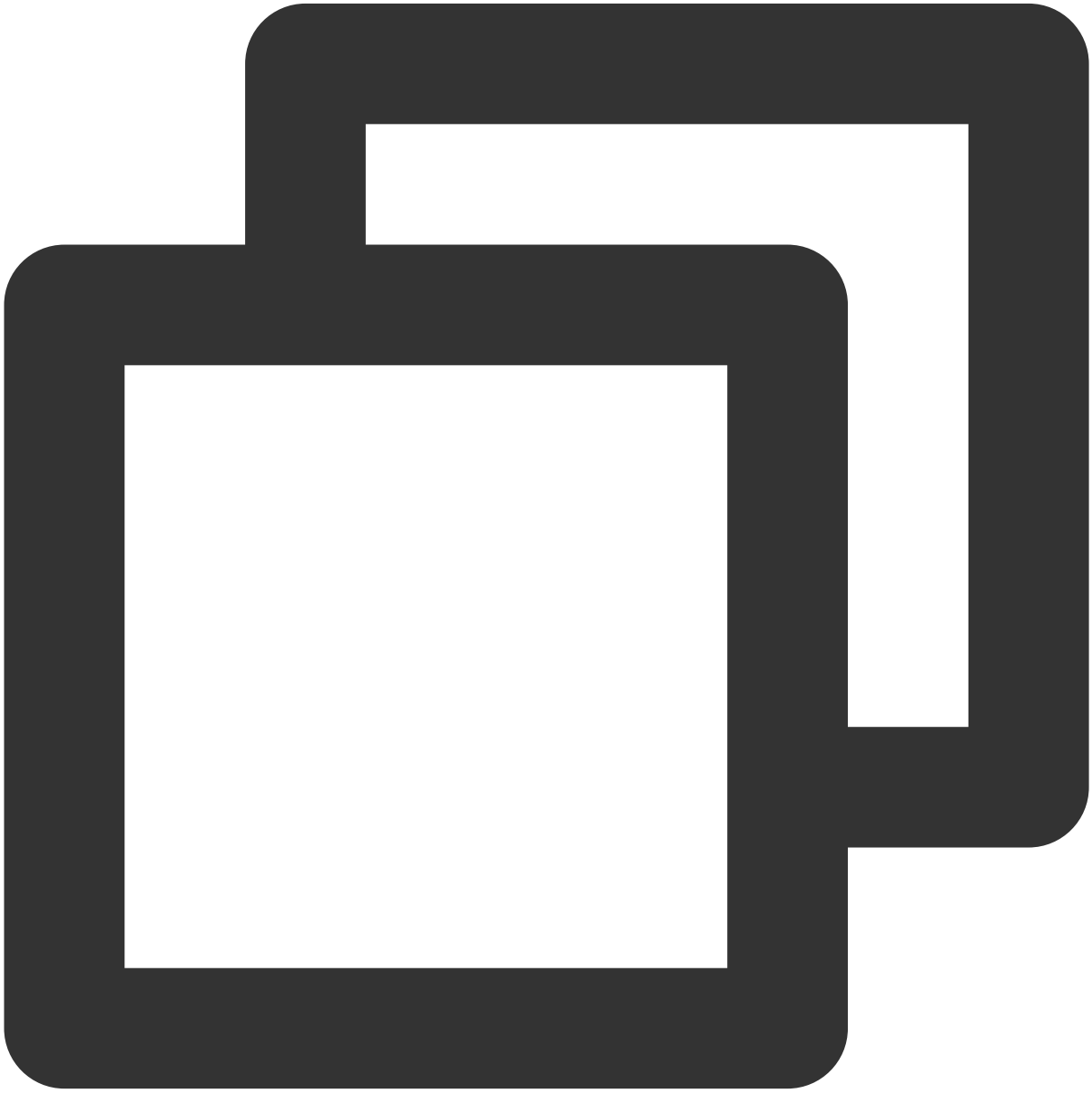
関数のプロトタイプ



```
ITMGPTT virtual int ApplyPTTAuthbuffer(const char* authBuffer, int authBufferLen)
```

パラメータ	タイプ	意味
authBuffer	char*	認証
authBufferLen	int	認証の長さ

サンプルコード

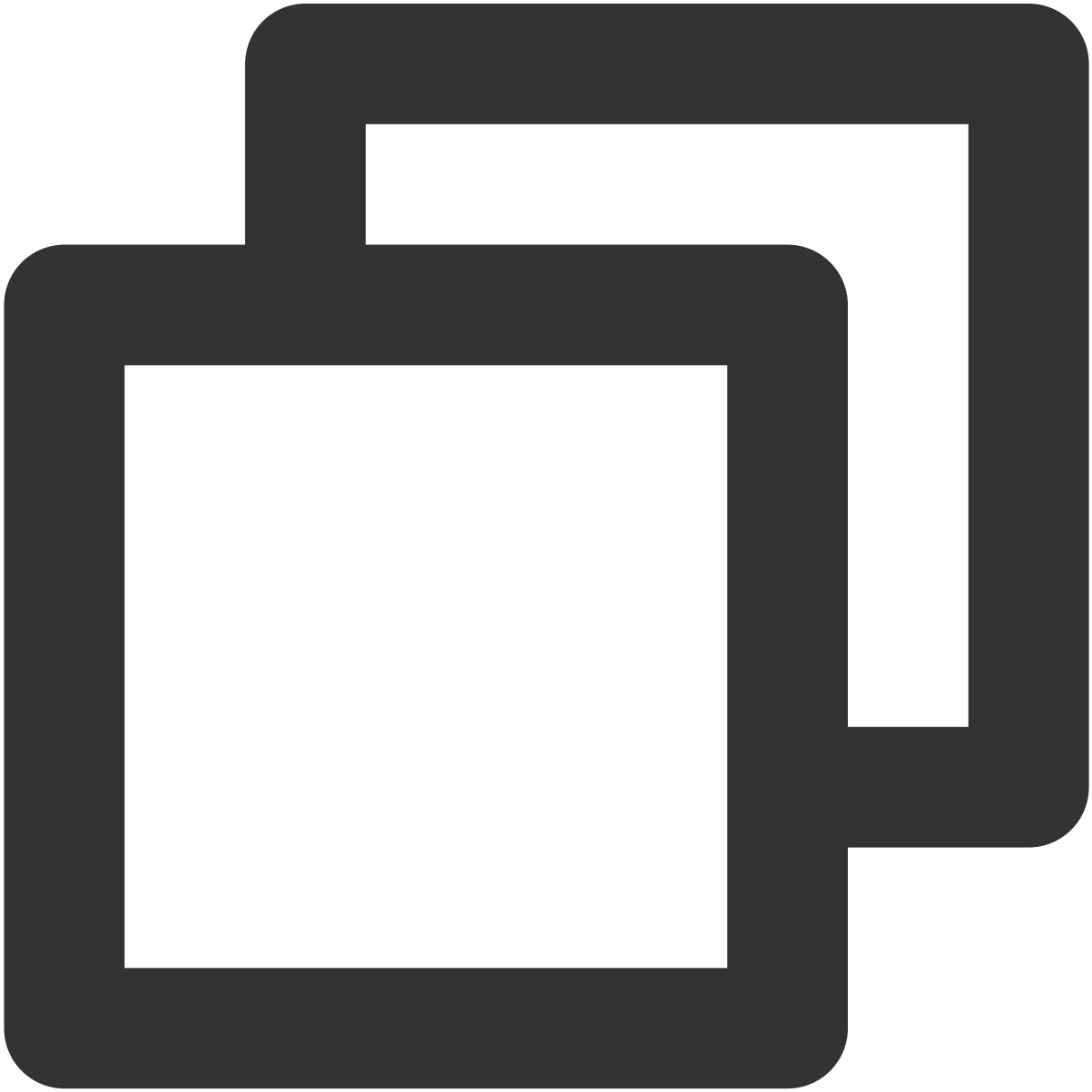


```
ITMGContextGetInstance()->GetPTT()->ApplyPTTAuthbuffer(authBuffer,authBufferLen);
```

2. ストリーミング音声認識を起動

このインターフェースは、ストリーミング音声識別の開始に使われています。コールバックにおいて、音声はリアルタイムでテキストに変換されて返されます。言語を指定し識別することができるし、音声から識別した情報を指定した言語に翻訳してから返すこともできます。**録音の停止にはStopRecording**を呼び出します。停止後にコールバックが発生します。

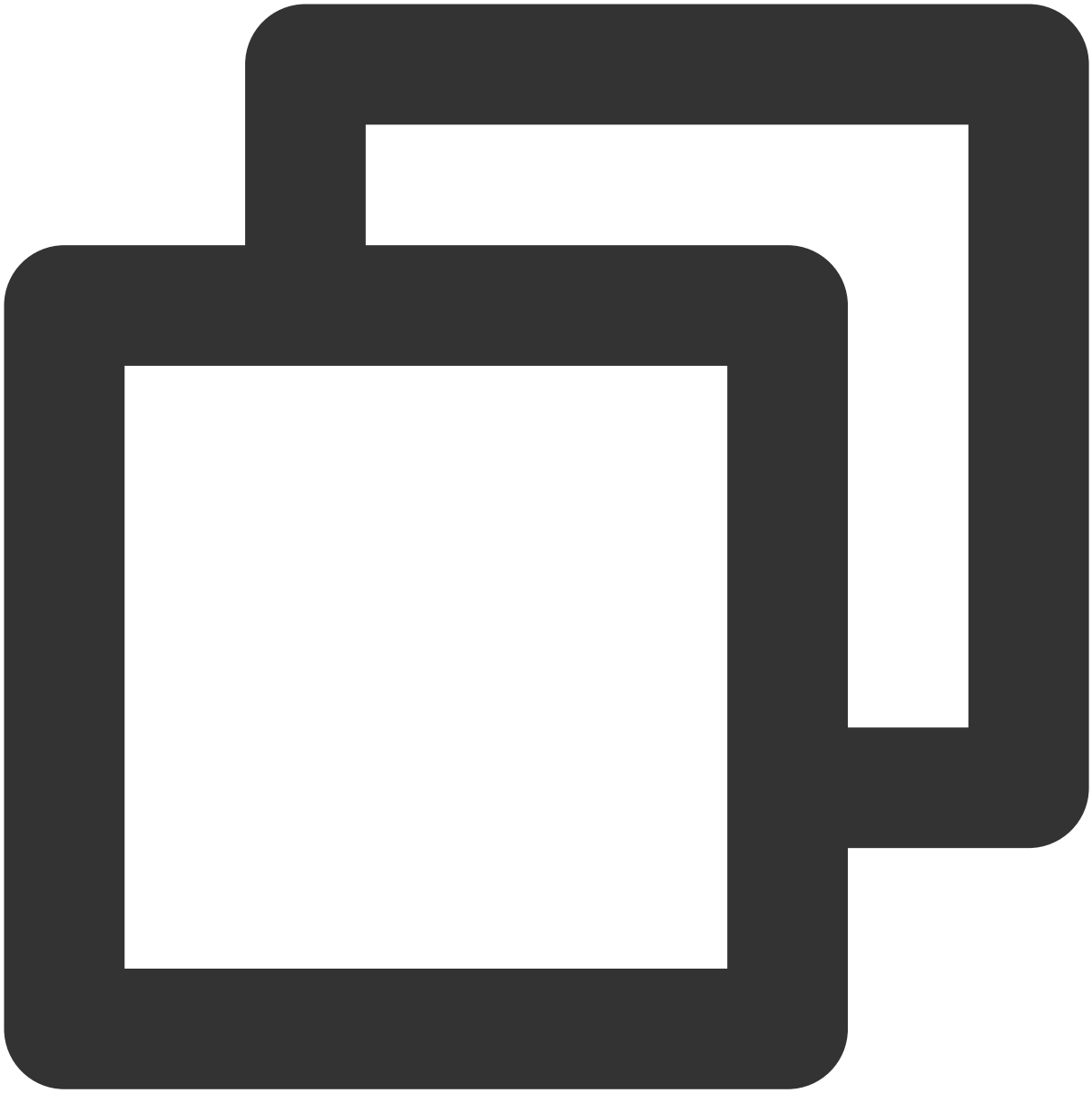
関数のプロトタイプ



```
ITMGPTT virtual int StartRecordingWithStreamingRecognition(const char* filePath)
ITMGPTT virtual int StartRecordingWithStreamingRecognition(const char* filePath,con
```

参数	タイプ	意味
filePath	char*	ボイスの保存パス
speechLanguage	char*	指定した言語のテキストに識別するパラメータです。パラメータについては、音声のテキスト変換の言語パラメータ参照リストをご参照ください
translateLanguage	char*	指定した言語のテキストに翻訳するパラメータです。パラメータについては、音声のテキスト変換の言語パラメータ参照リストをご参照ください。（このパラメータは一時的に無効です。speechLanguageと同じのパラメータを入力してください）

サンプルコード



```
ITMGContextGetInstance()->GetPTT()->StartRecordingWithStreamingRecognition(filePath
```

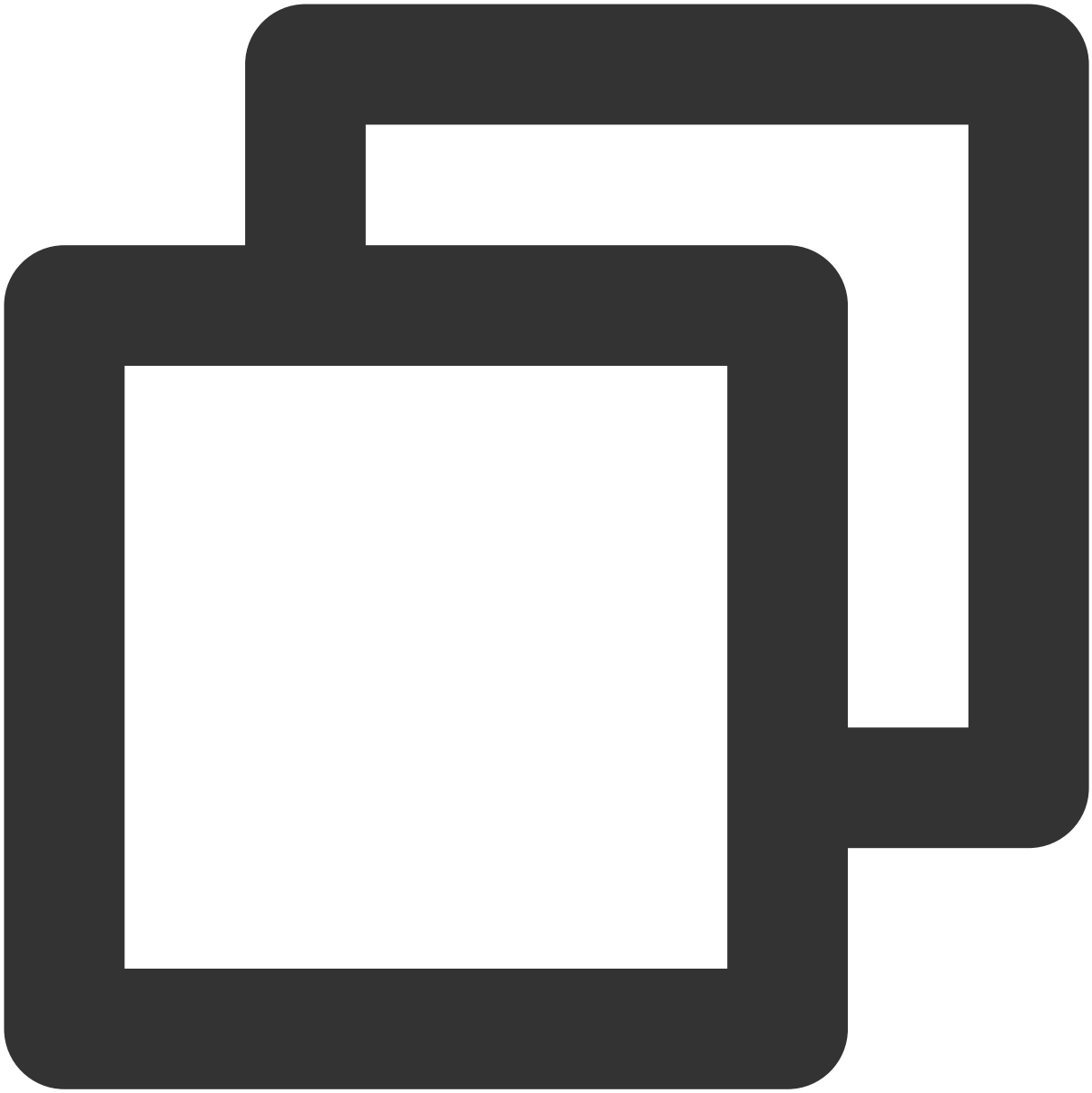
ストリーミング音声識別コールバック

ストリーミング音声認識を開始した後、コールバック関数OnEventでコールバックメッセージを受信する必要があります。イベントメッセージは `ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` があり、録音を停止して認識を完了した後にテキストを返します。これは、話が終わってから認識されたテキストを返すことに相当します。

OnEvent関数で、必要に応じて適切なイベントメッセージを判断します。渡されるパラメータには次の4つの情報が含まれます。

メッセージ名称	意味
result	ストリーミングボイス認識が完了したかどうかを判断するための戻りコード
text	ボイステキスト変換で認識されたテキスト
file_path	録音を保存するローカルアドレス
file_id	録音はバックグラウンドのURLアドレスにあり、録音はサーバーで90日間保存されます

サンプルコード



```
void UBaseViewController::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char *data)

FString jsonData = FString(UTF8_TO_TCHAR(data));
TSharedPtr<FJsonObject> JsonObject;
TSharedRef<TJsonReader<>> Reader = TJsonReaderFactory<>::Create(FString(UTF8_TO_TCH
FJsonSerializer::Deserialize(Reader, JsonObject);
...
else if(eventType == ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE)
  {
      int32 nResult = JsonObject->GetIntegerField(TEXT("result"));
      FString text = JsonObject->GetStringField(TEXT("text"));
```

```
        FString fileid = JsonObject->GetStringField(TEXT("file_id"));
        FString file_path = JsonObject->GetStringField(TEXT("file_path"));
        onPttStreamRecognitionCompleted(nResult,file_path, fileid, text);
    }
    else if(eventType == ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING)
    {
        int32 nResult = JsonObject->GetIntegerField(TEXT("result"));
        FString text = JsonObject->GetStringField(TEXT("text"));
        FString fileid = TEXT("STREAMINGRECOGNITION_IS_RUNNING");
        FString file_path = JsonObject->GetStringField(TEXT("file_path"));
        onPttStreamRecognitionisRunning(nResult,file_path, fileid, text);
    }
}
```

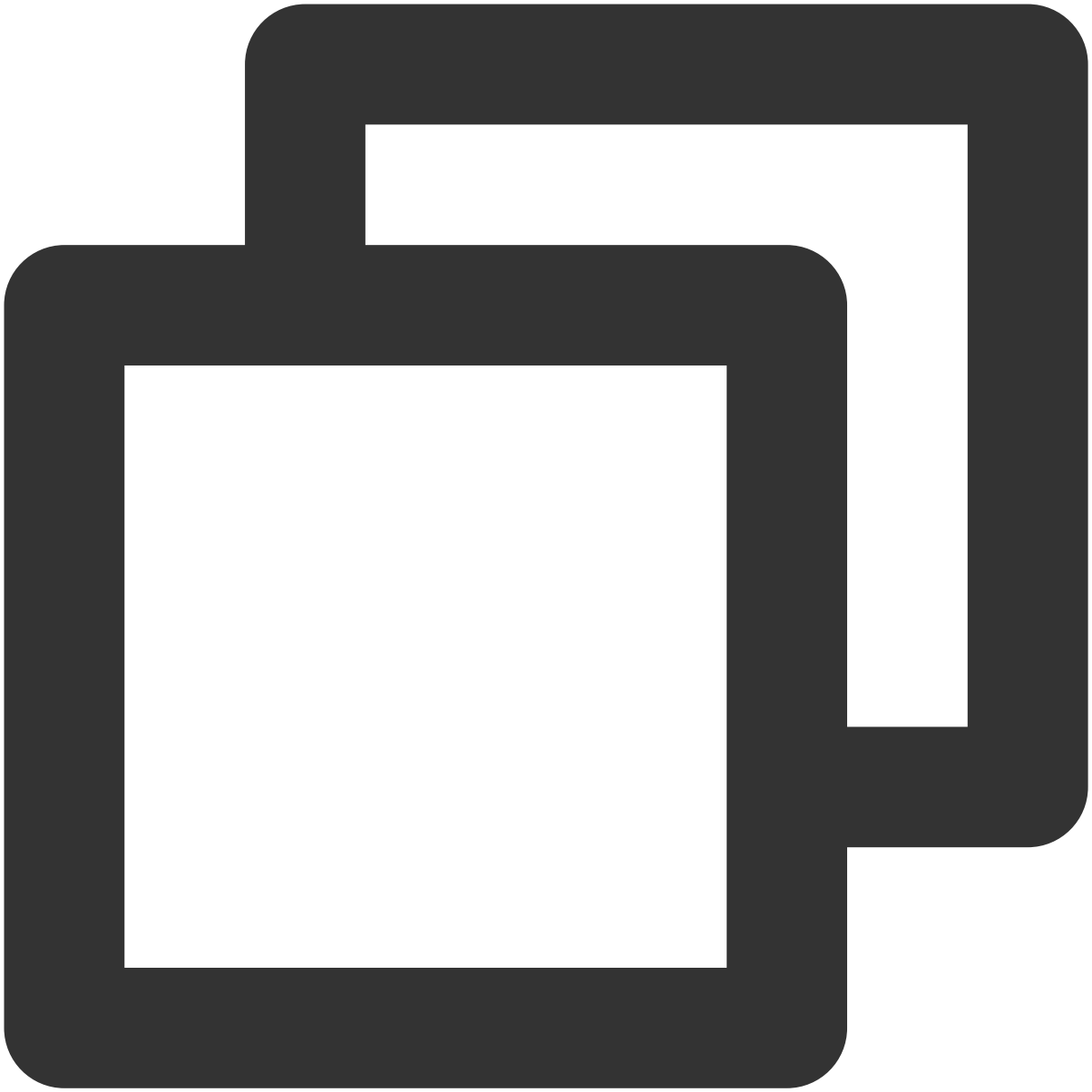
エラーコード

エラーコード	意味	処理方式
32775	ストリーミング音声をテキストに変更できませんが、録音は成功しました	UploadRecordedFileインターフェースを呼び出して録音をアップロードし、SpeechToTextインターフェースを呼び出して音声を文字に変換します
32777	ストリーミング音声をテキストに変更できませんが、録音とアップロードは成功しました	返されたメッセージには正常にアップロードしたバックグラウンドURLがあり、SpeechToTextインターフェースを呼び出して音声から文字への変換操作を行います
32786	ストリーミング音声をテキストに変更できませんでした	ストリーミングレコーディングステータスでは、ストリーミングレコーディングインターフェースの実行結果が返されるまで待ってください

3. 録音を停止

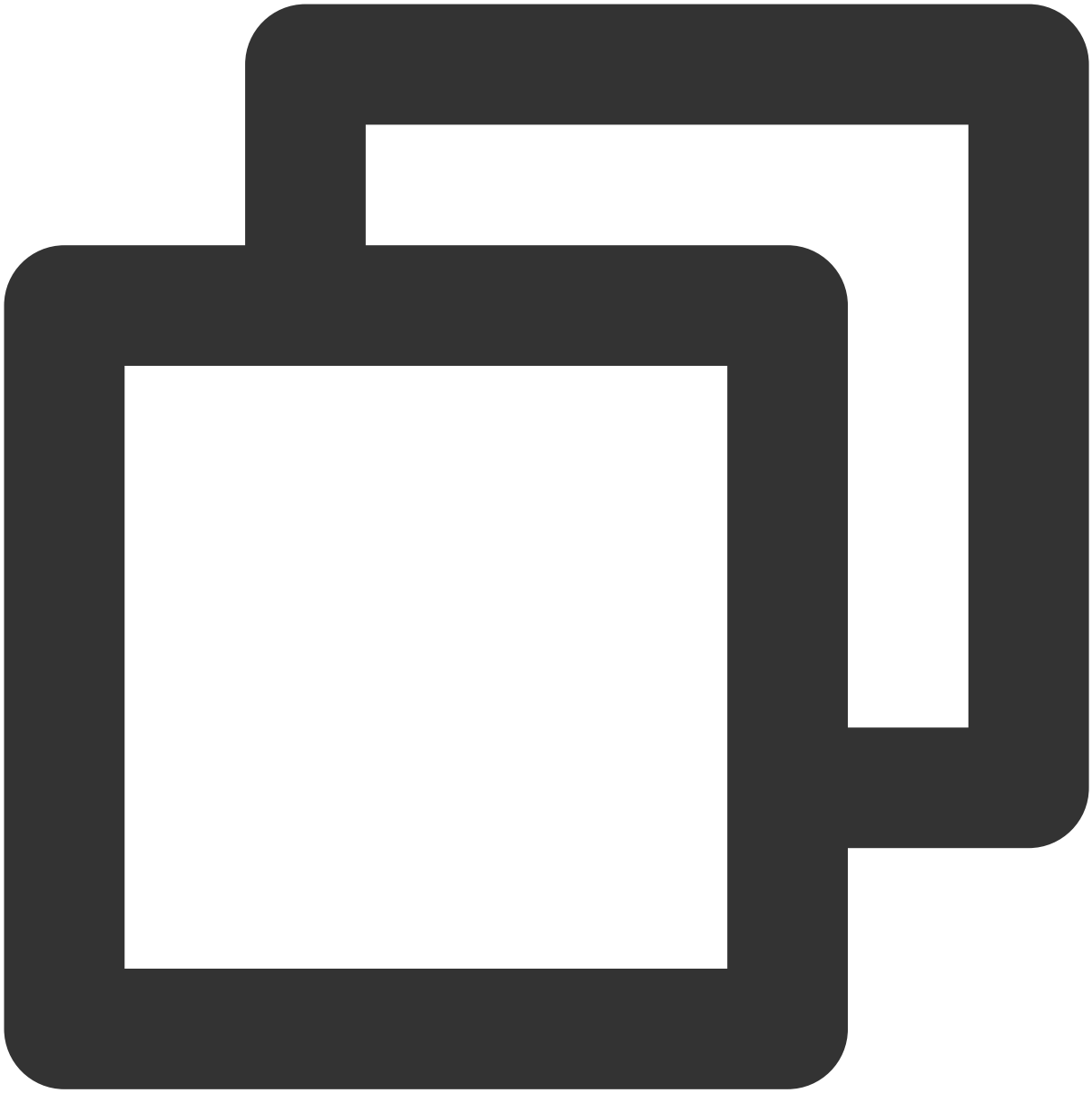
このインターフェースは、録音の停止に使われています。このインターフェースが非同期インターフェースであるため、録音を停止した後には録音完了のコールバックがあります。コールバックが成功してから、録音ファイルが利用できるようになります。

関数のプロトタイプ



```
ITMGPTT virtual int StopRecording()
```

サンプルコード



```
ITMGContextGetInstance()->GetPTT()->StopRecording();
```

Quick Integration of Sample Project
Unreal Sample Projectのクイックスタート

最終更新日：：2024-01-18 15:42:47

このドキュメントでは、GME Unreal Sample Projectをすばやく実行し、プロジェクトにプロジェクトサンプルコードを導入する方法について説明します。

Unreal Sample Projectのクイックスタート

環境要件

UnrealEngine 4.22以降のバージョンを推奨します。
Microsoft Visual Studio。
UnrealEngineプロジェクトを実行できる構成環境。

前提条件

事前にGMEリアルタイム音声、音声メッセージサービスを有効化し、AppIdとKeyを取得してください。GMEサービスの有効化については、サービスの有効化をご参照ください。**appIdはコンソールのAppID、authKeyはコンソールの秘密鍵に対応します。**

操作手順

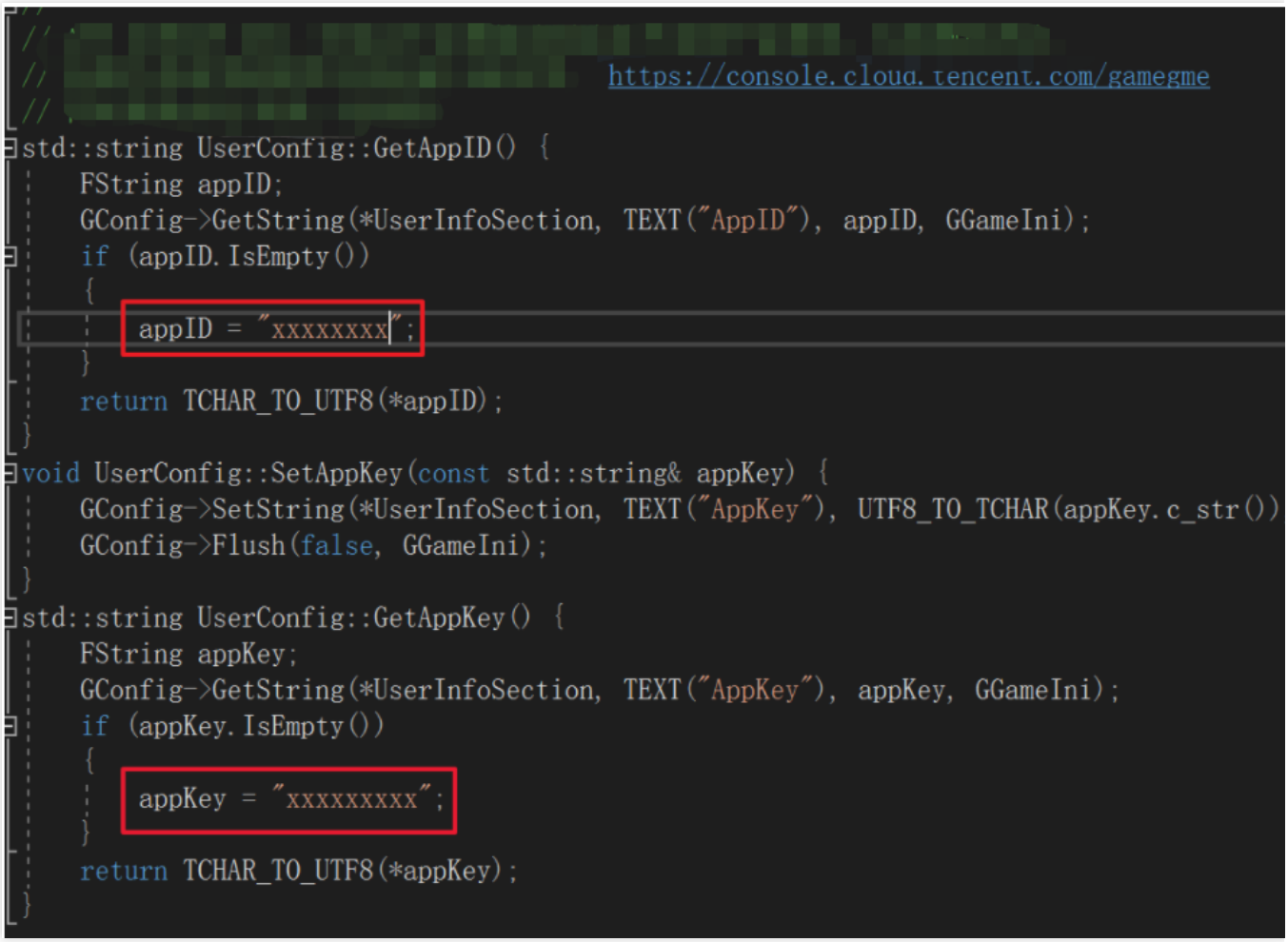
ステップ1：プロジェクトのダウンロード

ダウンロードガイドでUnreal Sample Projectをダウンロードしてください。UE5とUE4ではDemoの設定が異なるため、対応するエンジンバージョンのSample Projectをダウンロードする必要があります。

OS/Engine	Update Time	SDK Download	Sample Project Download	Documents
Unity	January 18, 2023	Download	Download	Quick Integratio Unity
Unreal Engine 4.x	January 18, 2023	Download	Download	Quick Integratio Unreal Engine
Unreal Engine 5.x	January 18, 2023	Download	Download	Quick Integratio Unreal Engine
Cocos2D	January 18, 2023	Download	Download	Getting Started
Windows	January 18, 2023	Download	Download	Native SDK Quic
iOS	January 18, 2023	Download	Download	Quick Integratio SDK
Android	January 18, 2023	Download	Download	Quick Integratio SDK
macOS	January 18, 2023	Download	Download	Quick Integratio SDK
Web	2022-06-20	Download	Download	API Documentat

ステップ2：プロジェクトの設定

ダウンロード後にプロジェクトディレクトリを開き、パスSource\\UEDemo1からUserConfig.cppを探し、図中の赤枠内のappID及びappKeyを申請したGMEコンソールサービス管理-アプリケーション設定のAppID及び権限キーに変更します。

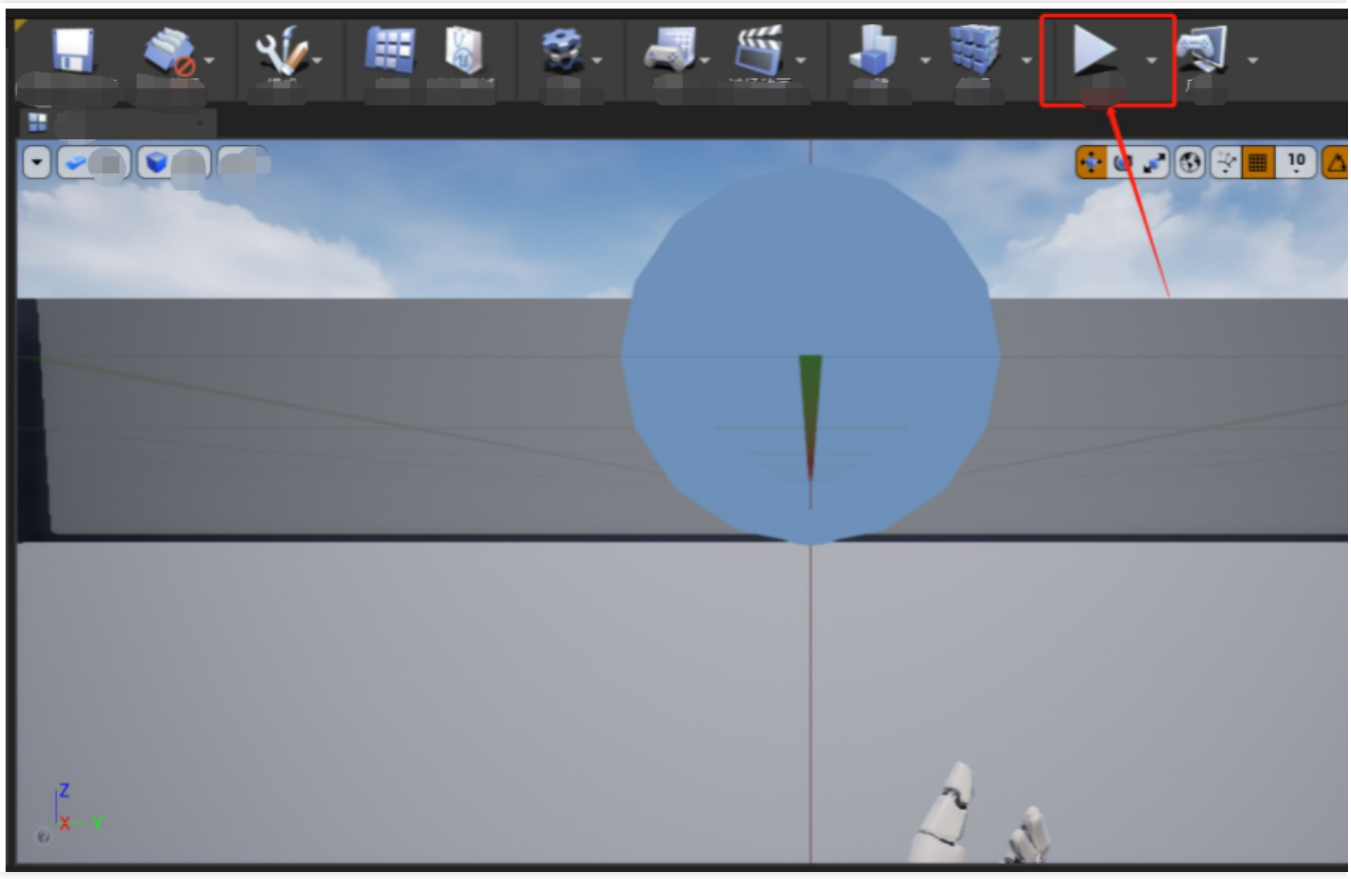


ステップ3：コンパイルと実行

1）プログラムの実行

「エディタを実行」ボタン

をクリックして、実行プログラムに移行します。



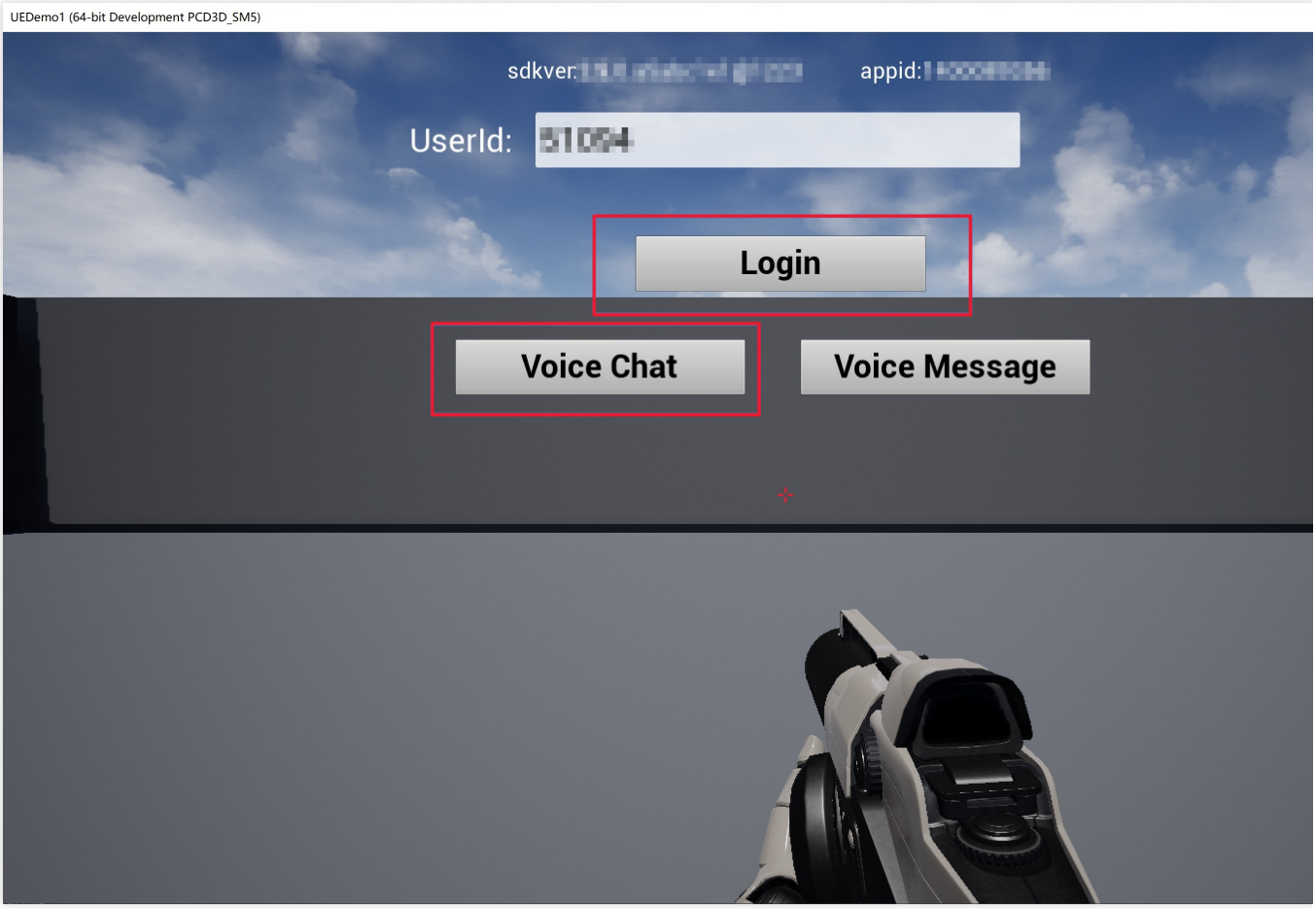
2）初期化

UserID：openIDに相当します。openIDはアプリケーション内でユーザを識別する唯一の識別子です。各端末で一意のopenidを確保しなければなりません。

Voice Chat：リアルタイム音声機能インターフェース。

Voice Message：音声メッセージ機能のインターフェース。

Loginをクリックして初期化を行い、**Voice Chat**をクリックしてリアルタイムボイスルームの設定画面を開きます。



3）リアルタイム音声ルームへ参加

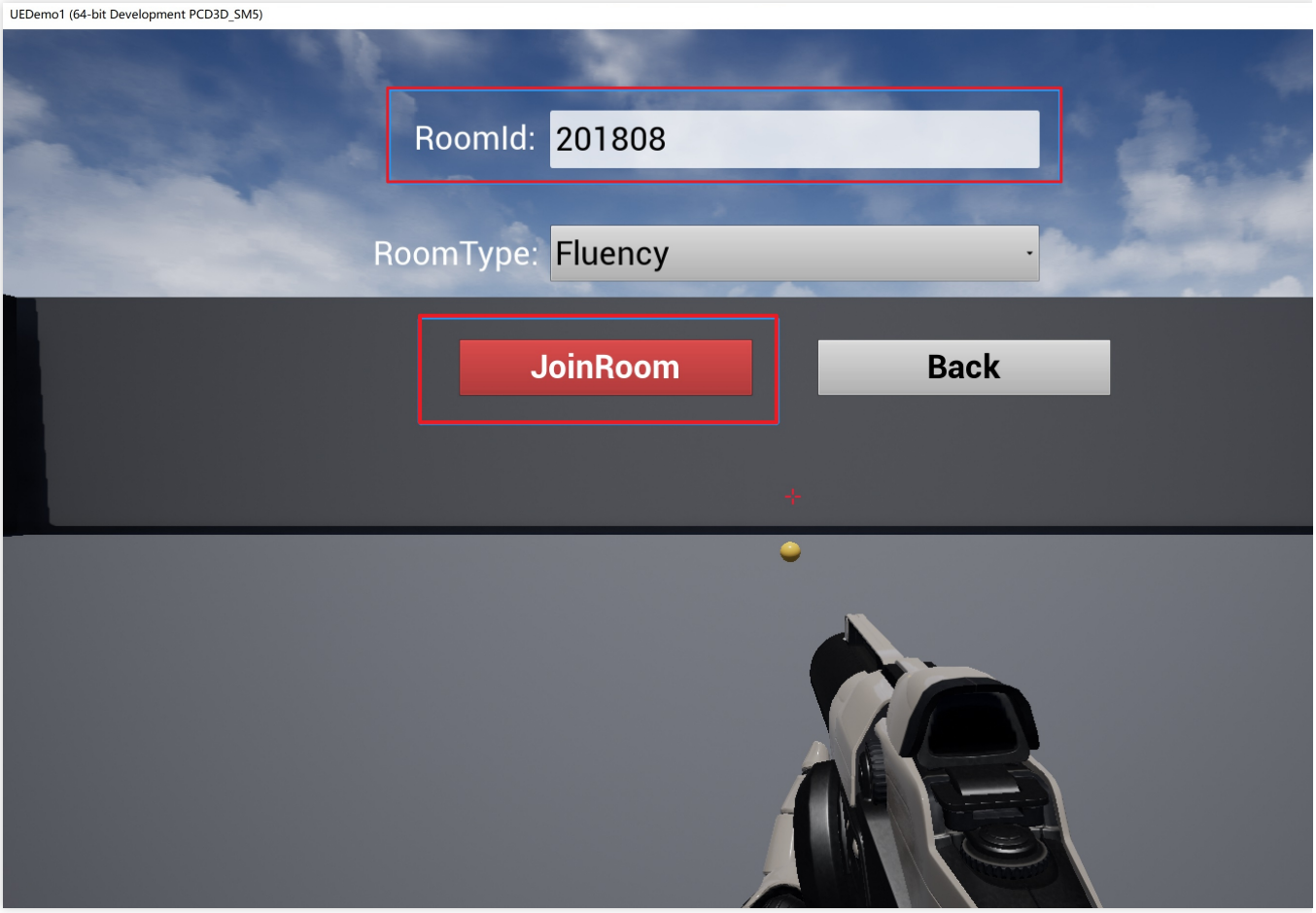
RoomId：部屋番号です。同じルームのメンバーが音声で話すことができます。

RoomType：Fluencyを使用してルームに参加してください。

JoinRoom：音声ルームに参加します。

Back：前の画面に戻ります。

リアルタイムボイスルーム番号を設定したら、**JoinRoom**をクリックして、リアルタイムボイスルームに入ります。



4）リアルタイム音声機能

ルームに参加したRoomidとローカル端末のopenidが画面に表示されます。

Mic：マイクをオンにします。

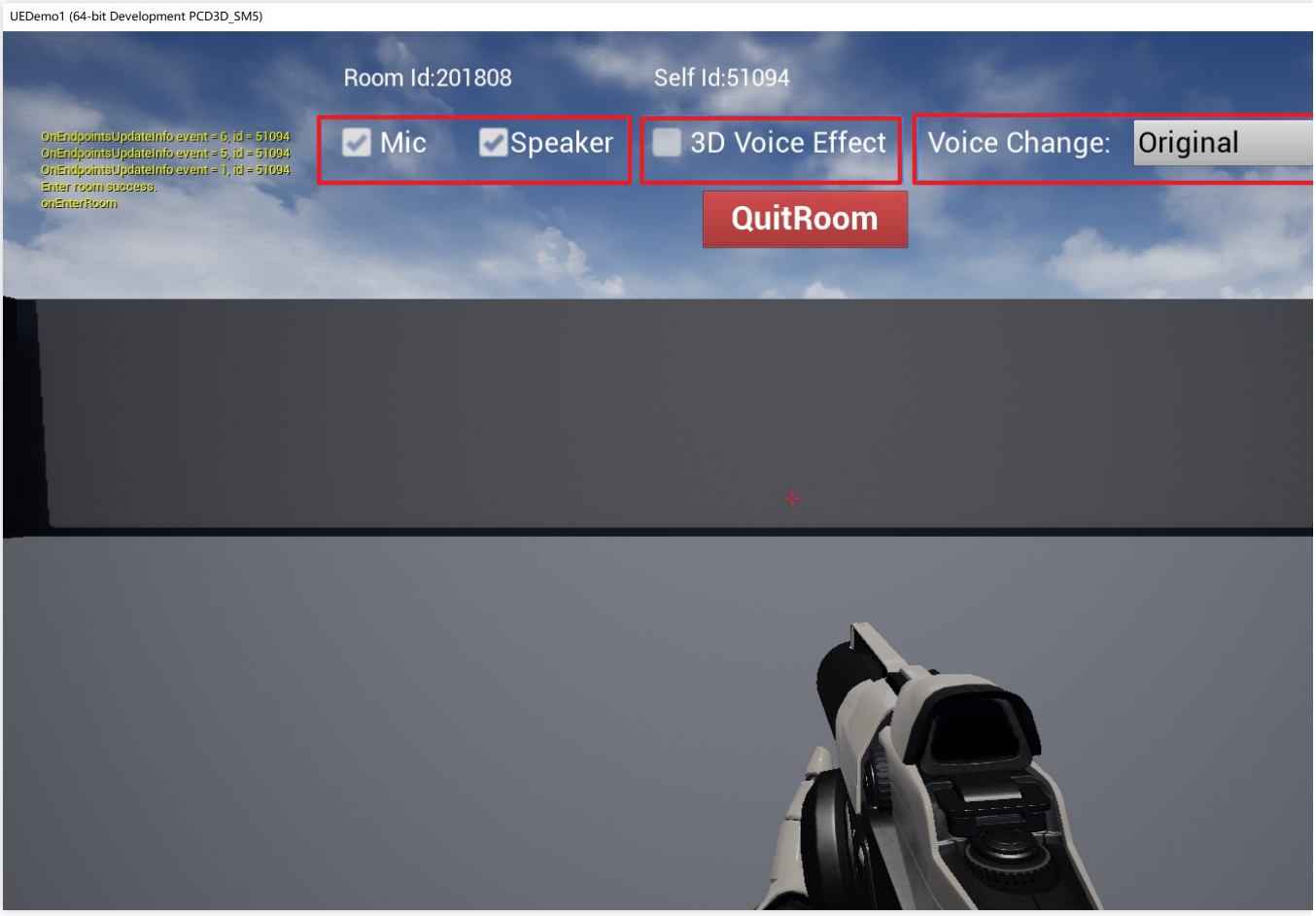
Speaker：スピーカーをオンにします。

3D Voice Effect：3Dサウンドをオンにします。

Voice Change：ボイス・チェンジエフェクトを選択します。

ローカル端末でMicおよびSpeakerにチェックを入れ、もう一方の端末の機器も以上の手順を繰り返し、同じルームに参加し、MicおよびSpeakerにチェックを入れ、互いにコミュニケーションを行うことができます。

両端末とも3D Voice Effectにチェックを入れ、キーボード【A】【S】【D】【W】で方位を変え、3D音声方位感を体験できます。



5）音声メッセージ機能

Language：テキスト変換対象音声を選択し、例えば中国語を話す場合は北京語を選択します。

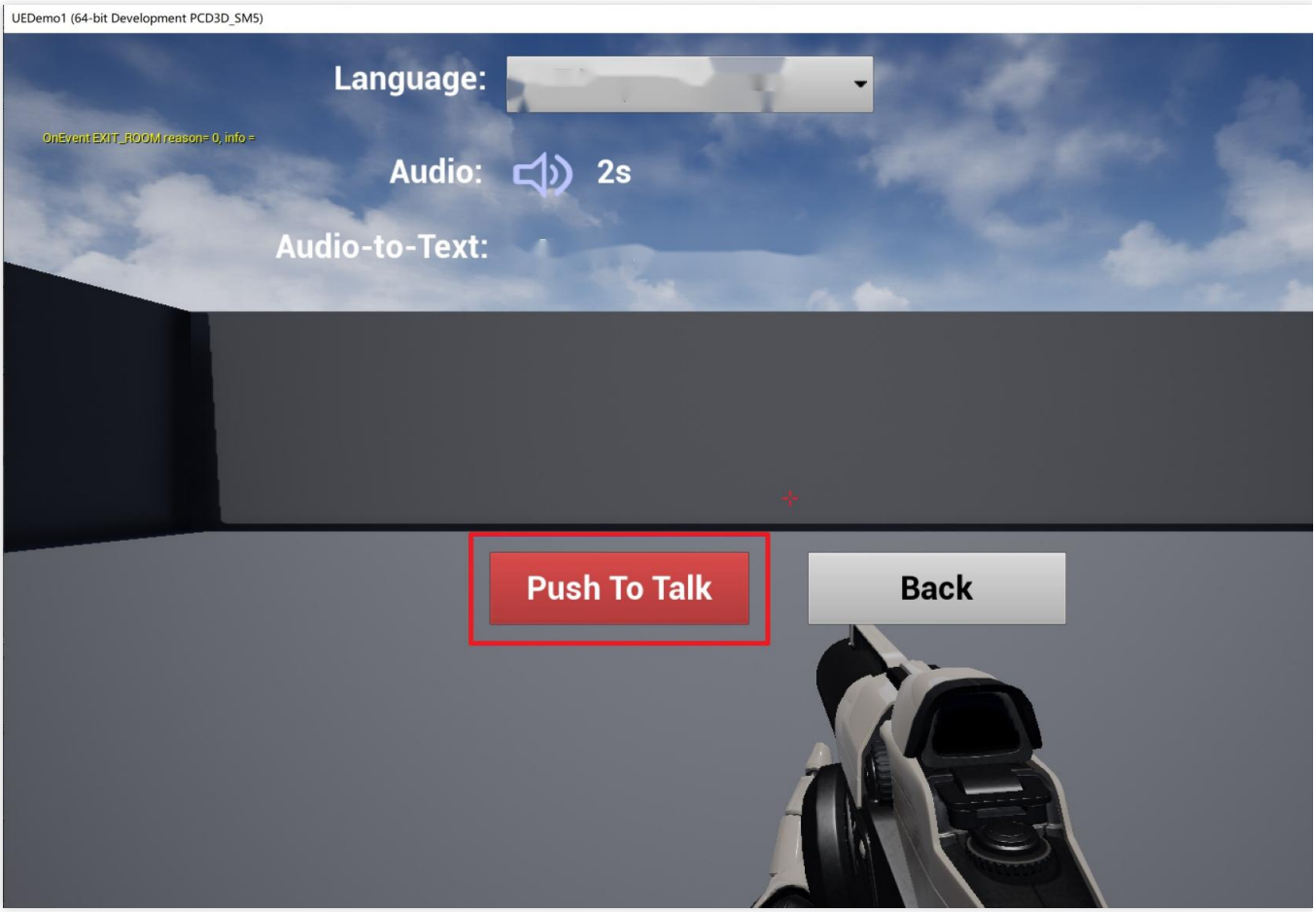
Audio：録音後にクリックすると聴くことができます。

Audio-to-Text：音声から変換されるテキスト内容。

Push To Talk：長押しすると録音できます。

Back：前の画面に戻ります。

Push to Talkボタンを長押ししたままマイクに向かって話しかけ、ボタンを離すと音声がテキストに変換されて画面に表示されます。

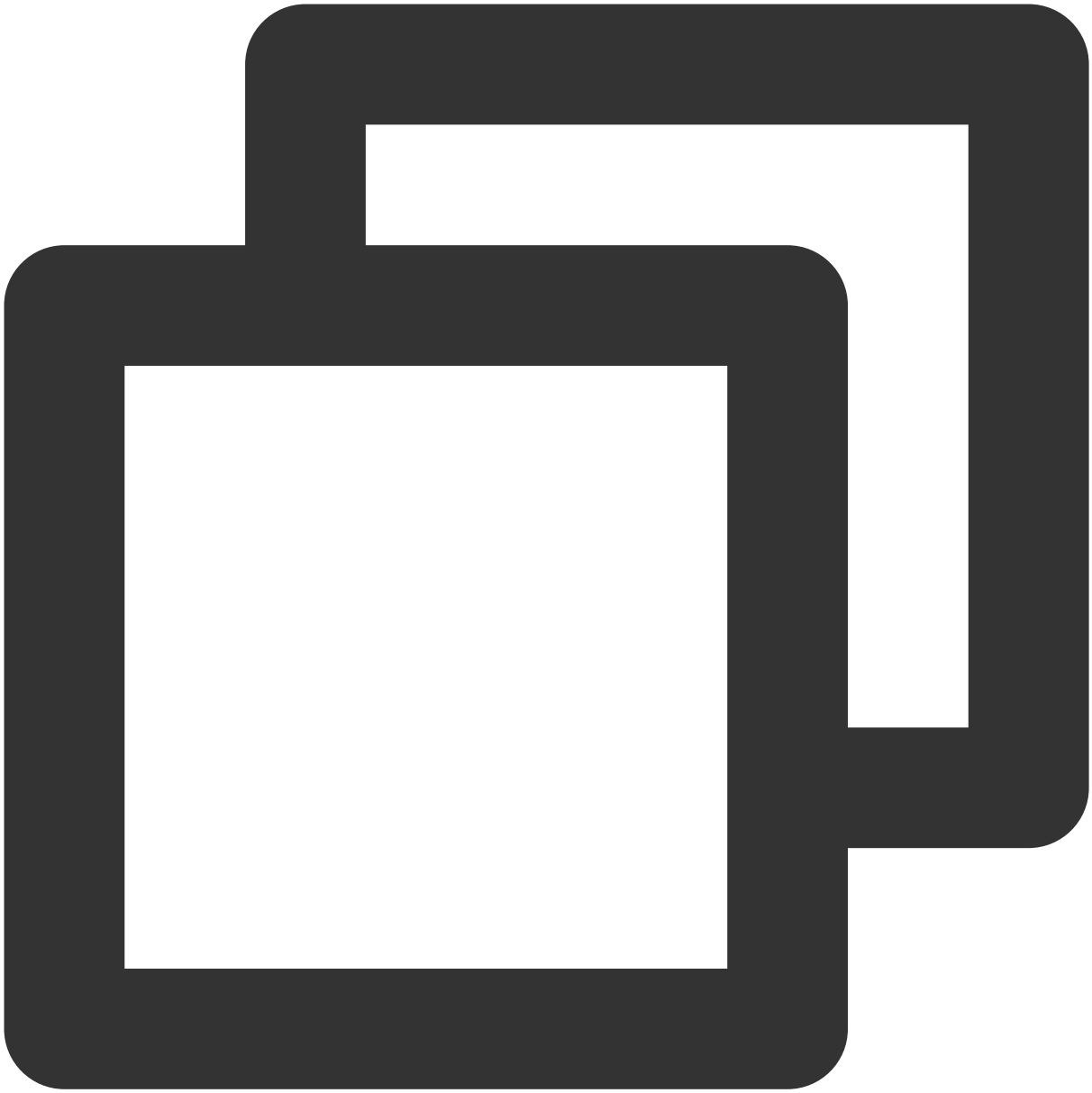


Sample Projectコードの説明

GMEリアルタイム音声を利用する主な手順は、Init>EnterRoom>EnableMic>EnableSpeakerです。Sample Projectの主なコードは、BaseViewController.cppおよびExperientialDemoViewController.cppにあります。

初期化関連

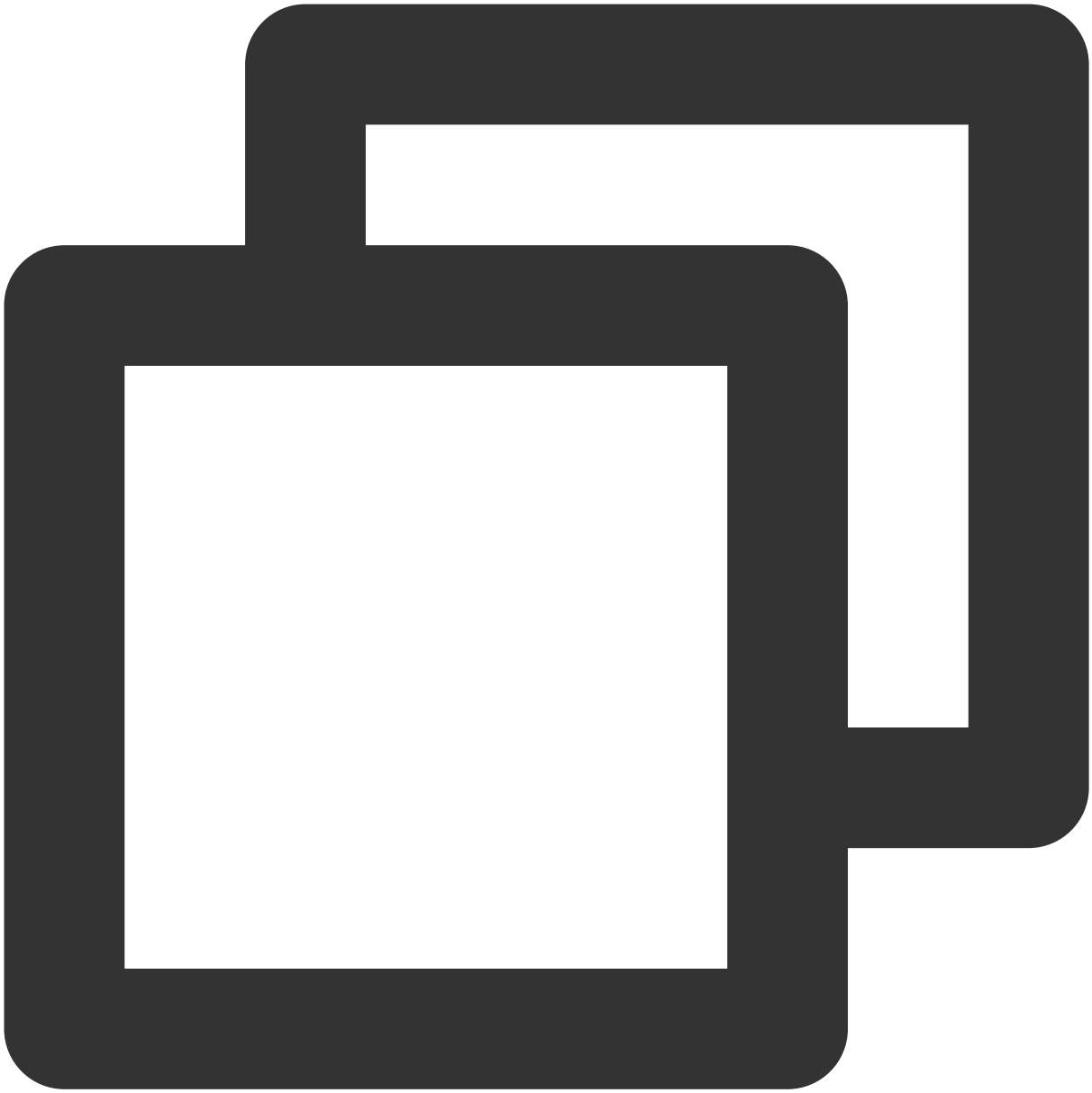
初期化に関連するコードは、BaseViewController.cppファイルのInitGME関数にあります。これには、初期化、音声メッセージの認証初期化、およびコールバックTMGDelegateの設定が含まれます。



```
int UBaseViewController::InitGME(std::string sdkAppId, std::string sdkAppKey, std::

    int nAppid = atoi(sdkAppId.c_str());
    int ret = ITMGContextGetInstance()->Init(sdkAppId.c_str(), userId.c_str());
    ITMGContextGetInstance()->SetTMGDelegate(this);

    int RetCode = (int) ITMGContextGetInstance()->CheckMicPermission();
    FString msg = FString::Printf(TEXT("check Permission retcode =%d"), RetCode);
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, *msg);

    char strSig[128] = {0};
```

```
    unsigned int nLength = 128;
    nLength = QAVSDK_AuthBuffer_GenAuthBuffer(nAppid, "0", userId.c_str(), sdkAppKe
    ITMGContextGetInstance()->GetPTT()->ApplyPTTAuthbuffer(strSig, nLength);

    m_appId = sdkAppId;
    m_appKey = sdkAppKey;
    m_userId = userId;
    m_isEnableTips = false;
    m_tipsMark = 0;
    return ret;
}
```

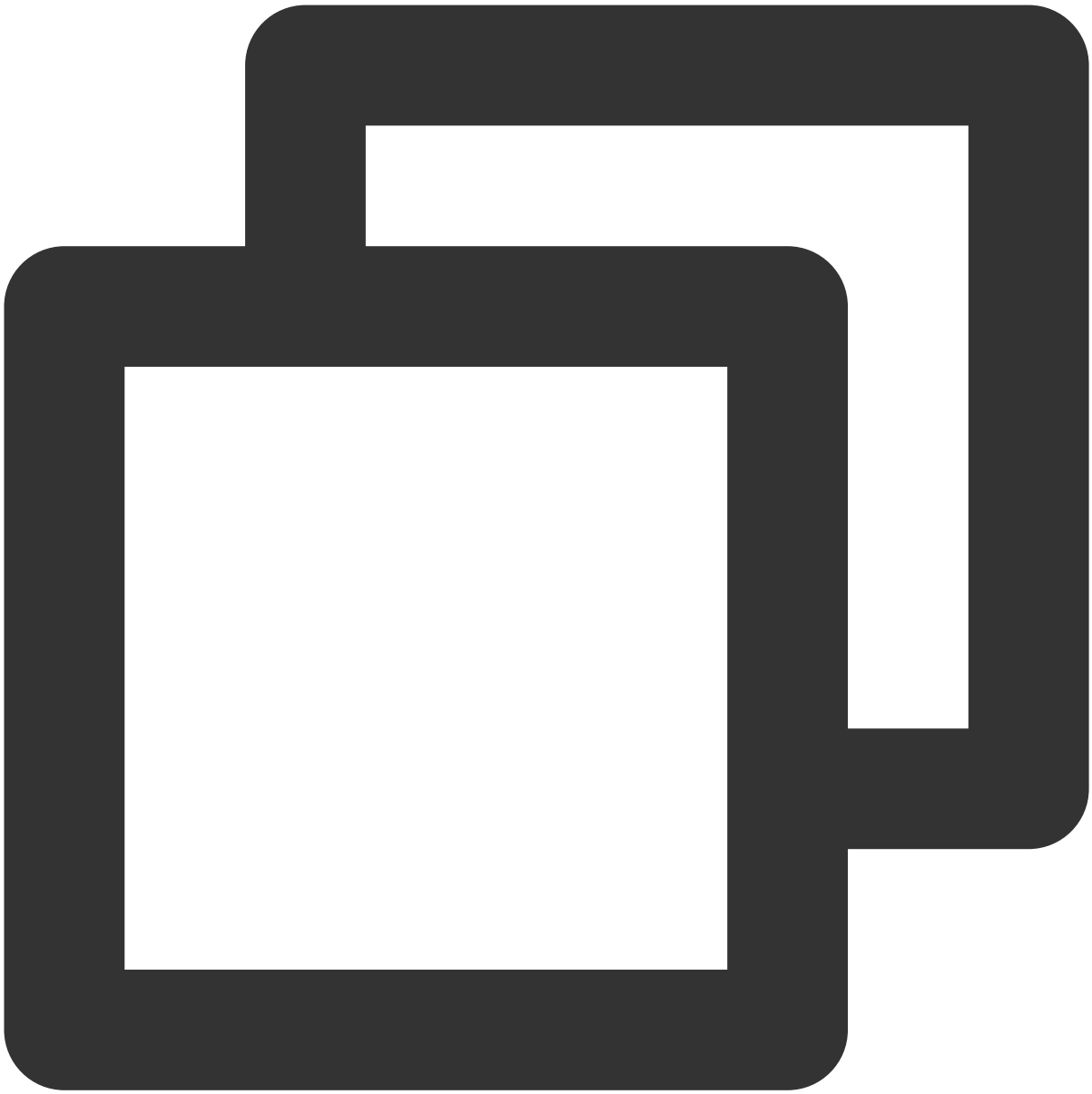
GMEを使用するにはPoll関数を定期的に呼び出してください。UEDemoLevelScriptActor.cppスクリプトのTickで呼び出されます。



```
void AUEDemoLevelScriptActor::Tick(float DeltaSeconds) {
    Super::Tick(DeltaSeconds);

    m_pTestDemoViewController->UpdateTips();
    m_pCurrentViewController->UpdatePosition();
    ITMGContextGetInstance()->Poll();
}
```

入室関連

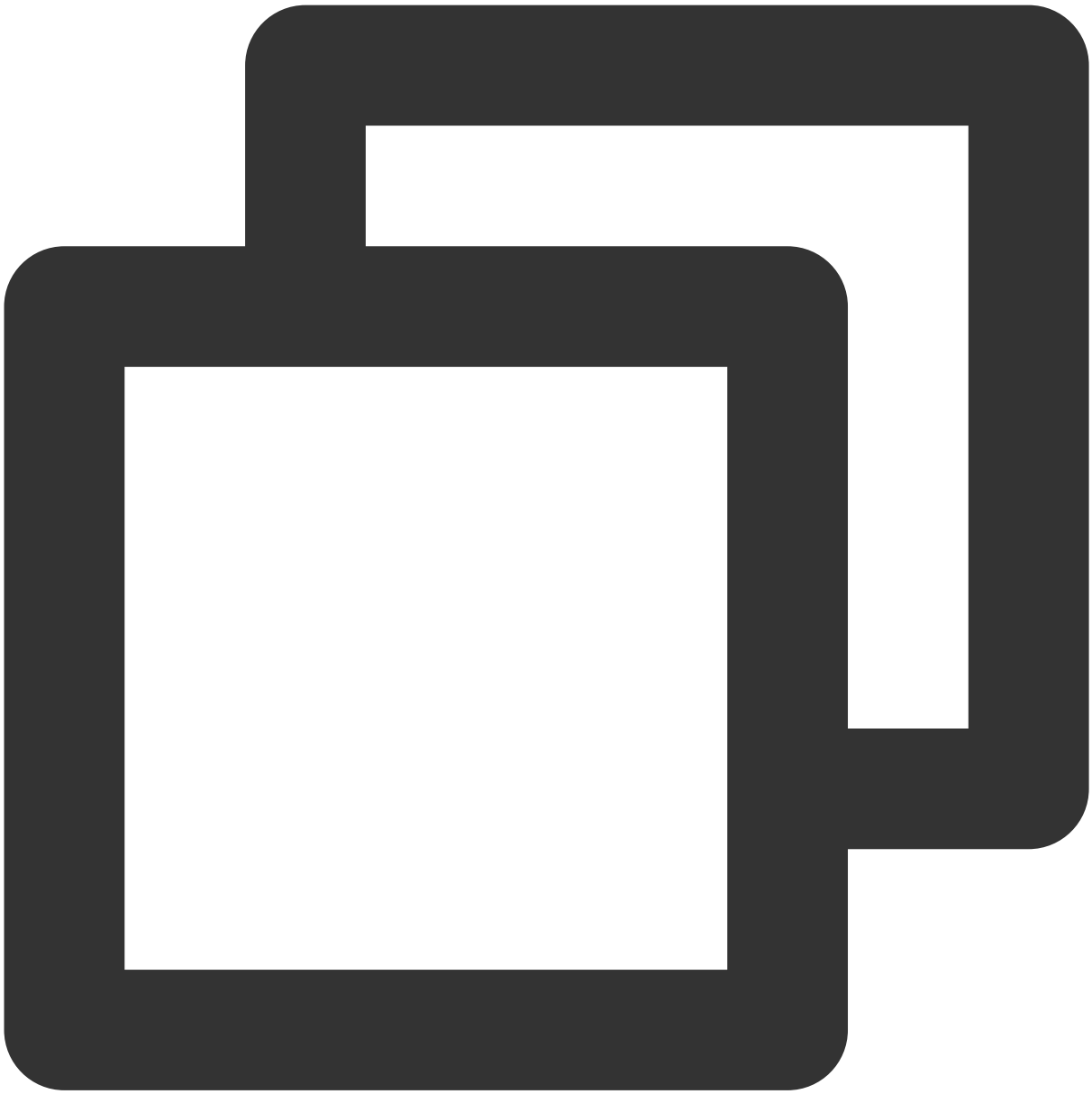
入室関連のコードは、BaseViewController.cppファイルのEnterRoom関数にあります。



```
void UBaseViewController::EnterRoom(std::string roomID, ITMG_ROOM_TYPE roomType) {
    int nAppid = atoi(m_appId.c_str());
    UserConfig::SetRoomID(roomID);

    char strSig[128] = {0};
    unsigned int nLength = 128;
    nLength = QAVSDK_AuthBuffer_GenAuthBuffer(nAppid, roomID.c_str(), m_userId.c_st
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, TEXT("onEnt
    ITMGContextGetInstance()->EnterRoom(roomID.c_str(), roomType, strSig, nLength);
```

```
}
```

入室のコールバックは同じスクリプト内のOnEvent関数中にあります。

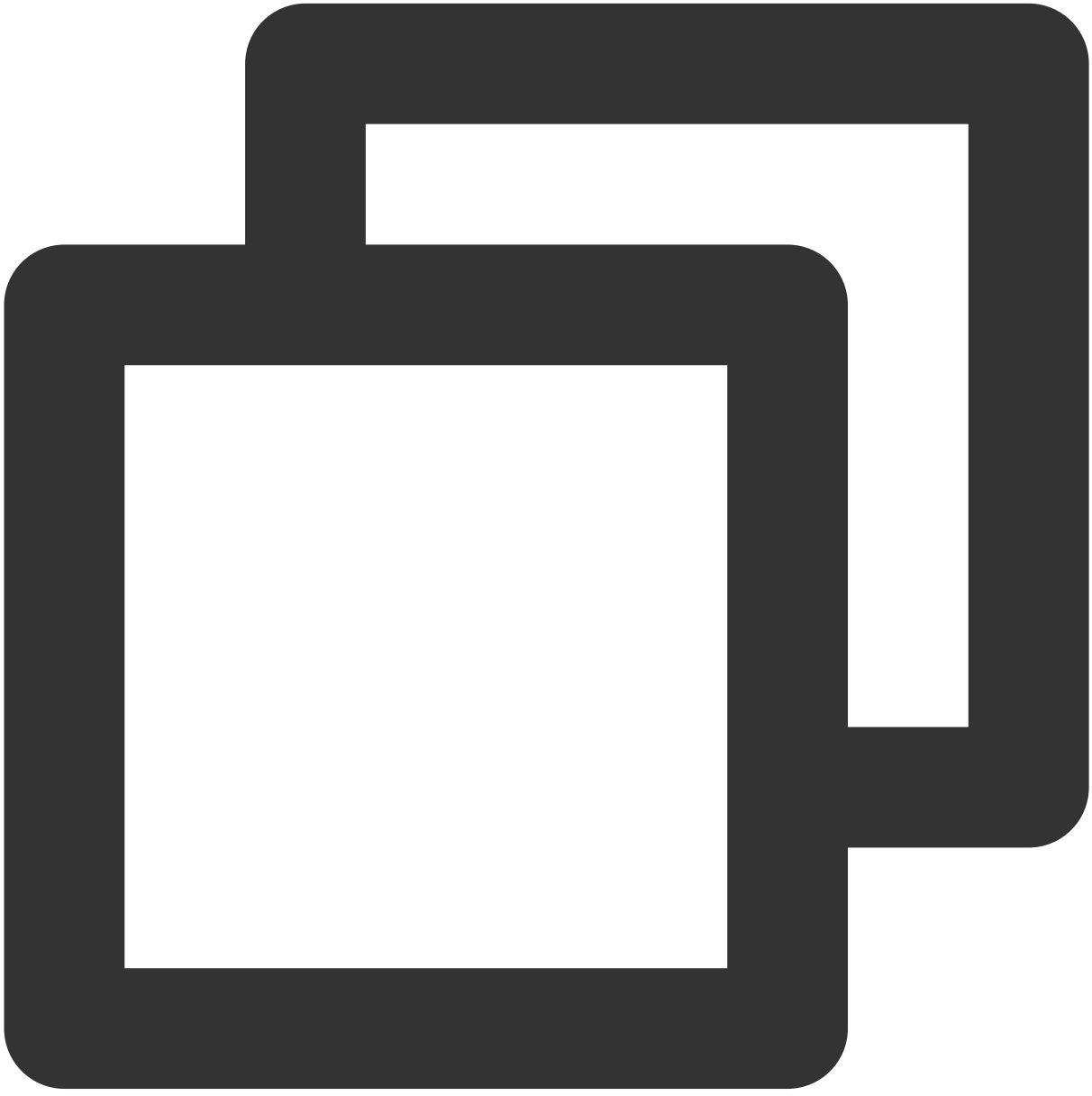


```
if (eventType == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM) {
        int32 result = JsonObject->GetIntegerField(TEXT("result"));
        FString error_info = JsonObject->GetStringField(TEXT("error_info"));
        if (result == 0) {
            GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, TEX
        }
        else{
            FString msg = FString::Printf(TEXT("Enter room failed. result=%d, info
```

```
            GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, *ms
        }
        onEnterRoomCompleted(result, error_info);
```

デバイスをオンにする

正常に入室すると、デバイスをオンにします。関連コードはExperientialDemoViewController.cppにあります。

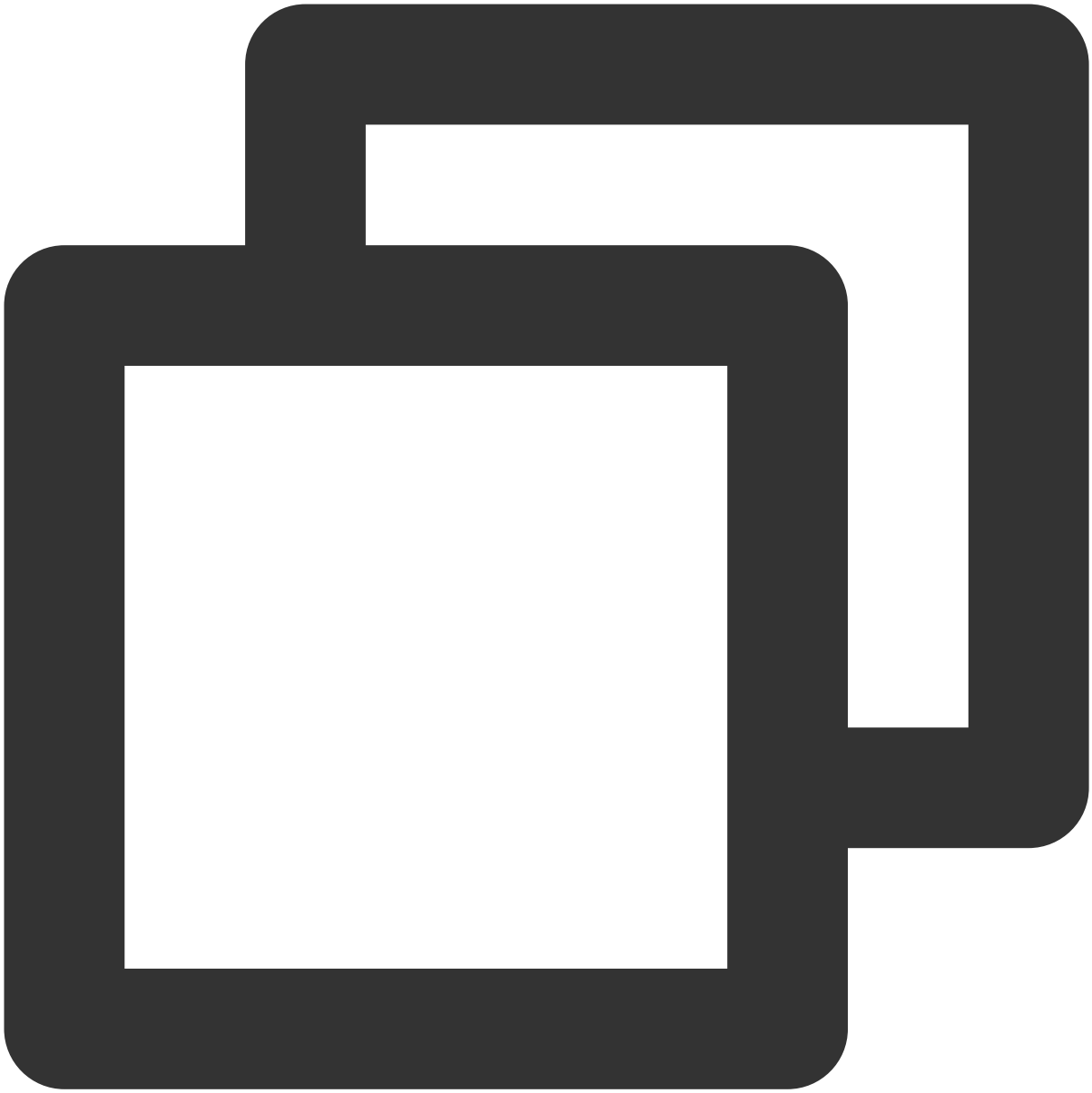


```
void UExperientialDemoViewController::onCheckMic(bool isChecked) {
    //GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, L"onCheck
    ITMGContext *pContext = ITMGContextGetInstance();
```

```
    if (pContext) {
        ITMGAudioCtrl *pTmgCtrl = pContext->GetAudioCtrl();
        if (pTmgCtrl) {
            pTmgCtrl->EnableMic(isChecked);
        }
    }
}

void UExperientialDemoViewController::onCheckSpeaker(bool isChecked) {
    //GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, L"onCheck
    ITMGContext *pContext = ITMGContextGetInstance();
    if (pContext) {
        ITMGAudioCtrl *pTmgCtrl = pContext->GetAudioCtrl();
        if (pTmgCtrl) {
            pTmgCtrl->EnableSpeaker(isChecked);
        }
    }
}
```

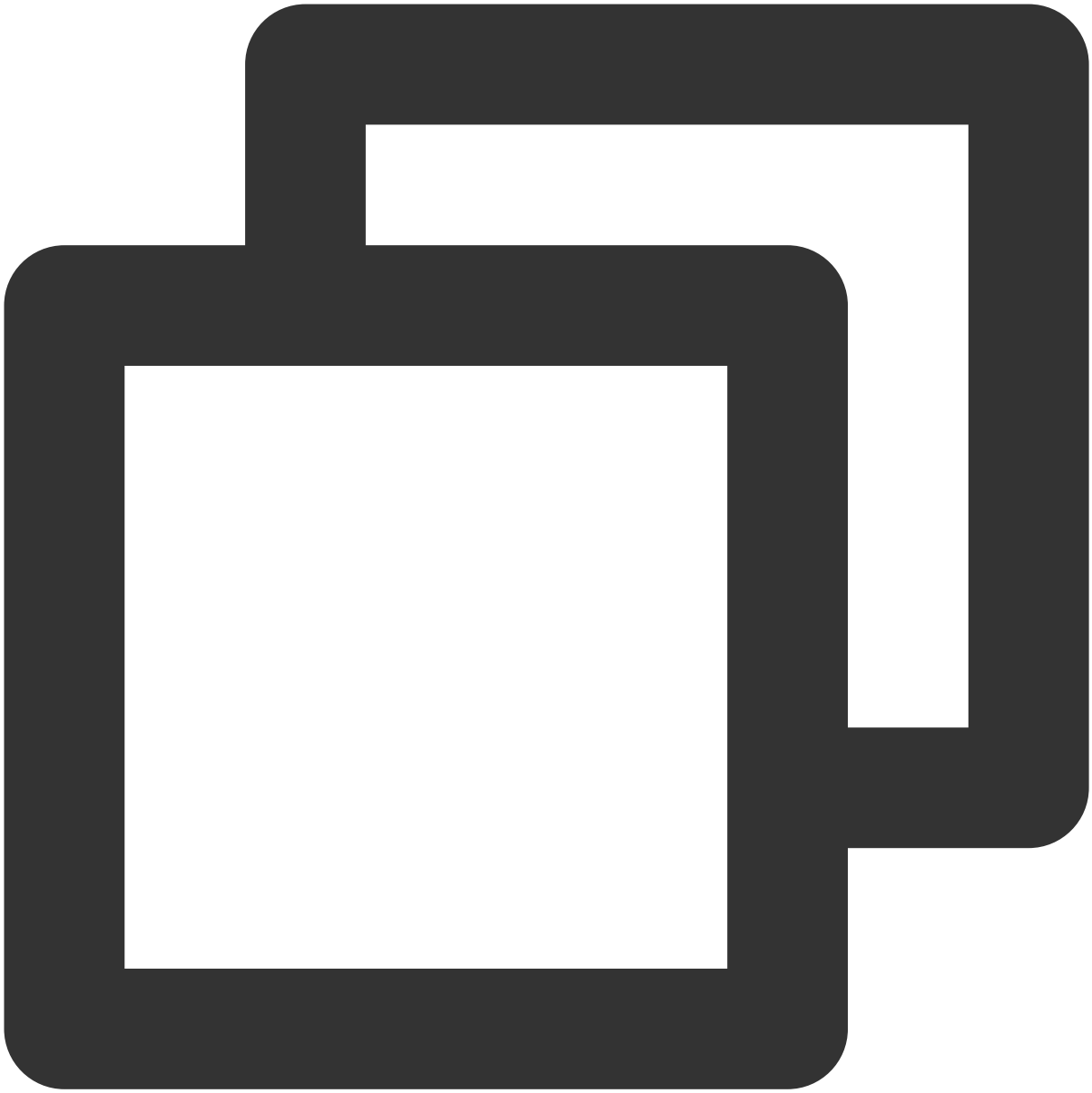
3Dサウンド関連

3Dサウンドの導入について、3Dサウンドドキュメントをご参照ください。プロジェクトでは、まず3Dサウンド機能が初期化され、関連コードはExperientialDemoViewController.cppにあります。



```
void UExperientialDemoViewController::onCheckSpatializer(bool isChecked) {
    char buffer[256]={0};
//    snprintf(buffer, sizeof(buffer), "%s3d_model", getFilePath().c_str());
    snprintf(buffer, sizeof(buffer), "%sgme_2.8_3d_model.dat", getFilePath().c_str(
    int ret1 = ITMGContextGetInstance()->GetAudioCtrl()->InitSpatializer(buffer);
    int ret2 = ITMGContextGetInstance()->GetAudioCtrl()->EnableSpatializer(isChecke
    FString msg = FString::Printf(TEXT("InitSpatializer=%d, EnableSpatializer ret=%
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, msg);
}
```

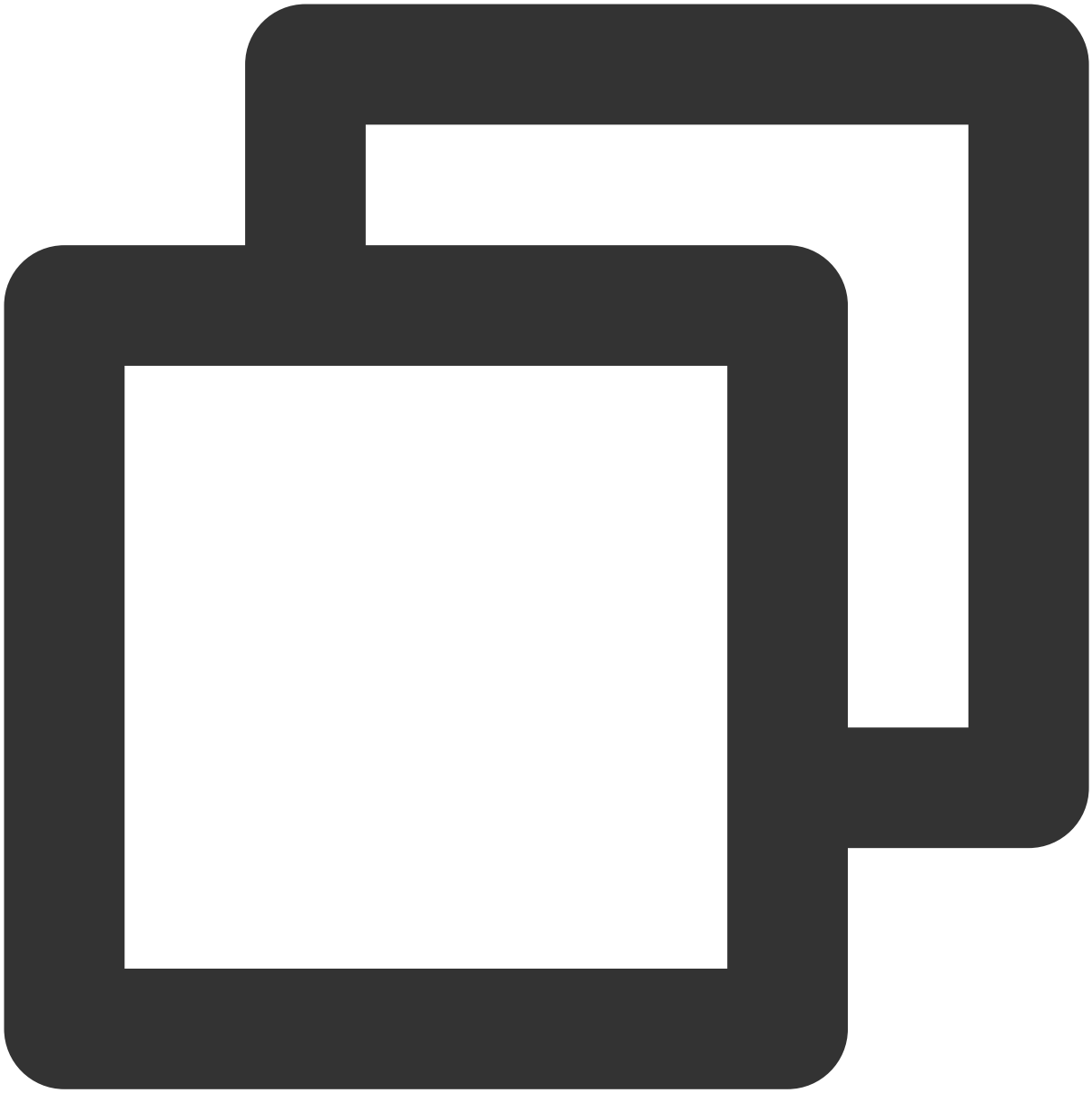
UEDemoLevelScriptActor.cppスクリプトのTickでUpdatePosition関数が呼び出されます。



```
void AUEDemoLevelScriptActor::Tick(float DeltaSeconds) {
    Super::Tick(DeltaSeconds);

    m_pTestDemoViewController->UpdateTips();
    m_pCurrentViewController->UpdatePosition();
    ITMGContextGetInstance()->Poll();
}


void UBaseViewController::UpdatePosition() {
    if (!m_isCreated)
```

```
        return;

    ITMGRoom *pTmgRoom = ITMGContextGetInstance()->GetRoom();
    if (!pTmgRoom)
    {
        return;
    }

    int nRange = GetRange();
    pTmgRoom->UpdateAudioRecvRange(nRange);

    FVector cameraLocation = UGameplayStatics::GetPlayerCameraManager(m_pActor->Get
    FRotator cameraRotation = UGameplayStatics::GetPlayerCameraManager(m_pActor->Ge

    FString msg = FString::Printf(TEXT("location(x=%.2f,y=%.2f,z=%.2f),  rotation(p
        cameraLocation.X, cameraLocation.Y, cameraLocation.Z, cameraRotation.Pitch,

    int position[] = { (int)cameraLocation.X,(int)cameraLocation.Y, (int)cameraLoca
    FMatrix matrix = ((FRotationMatrix)cameraRotation);
    float forward[] = { matrix.GetColumn(0).X,matrix.GetColumn(1).X,matrix.GetColum
    float right[] = { matrix.GetColumn(0).Y,matrix.GetColumn(1).Y,matrix.GetColumn(
    float up[] = { matrix.GetColumn(0).Z,matrix.GetColumn(1).Z,matrix.GetColumn(2).


    pTmgRoom->UpdateSelfPosition(position, forward, right, up);
    SetPositionInfo(msg);
}
```

ExperientialDemoViewController.cppで3Dサウンドをオンにします。



```
void UExperientialDemoViewController::onCheckSpatializer(bool isChecked) {
    char buffer[256]={0};
//    snprintf(buffer, sizeof(buffer), "%s3d_model", getFilePath().c_str());
    snprintf(buffer, sizeof(buffer), "%sgme_2.8_3d_model.dat", getFilePath().c_str(
    int ret1 = ITMGContextGetInstance()->GetAudioCtrl()->InitSpatializer(buffer);
    int ret2 = ITMGContextGetInstance()->GetAudioCtrl()->EnableSpatializer(isChecke
    FString msg = FString::Printf(TEXT("InitSpatializer=%d, EnableSpatializer ret=%
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, msg);
}
```