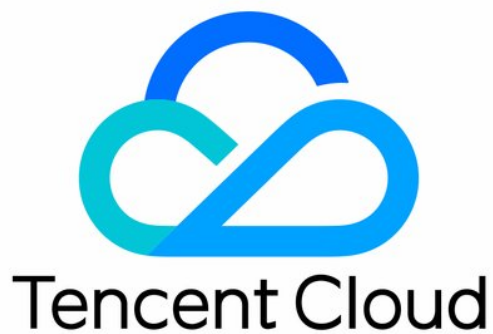


Game Multimedia Engine

Getting Started

Product Documentation



Copyright Notice

©2013-2025 Tencent Cloud. All rights reserved.

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

Trademark Notice



All trademarks associated with Tencent Cloud and its services are owned by the Tencent corporate group, including its parent, subsidiaries and affiliated companies, as the case may be. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

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

Contents

Getting Started

Quick Integration of SDK

- Quick Integration of Native SDK

- Quick Integration of SDK for Unity

- Quick Integration of SDK for Unreal Engine

Quick Integration of Sample Project

- Quick Run of Unreal Engine Sample Project

Getting Started

Quick Integration of SDK

Quick Integration of Native SDK

Last updated : 2024-01-18 11:53:35

This document provides a detailed description that makes it easy for Native project developers to debug and integrate the APIs for Game Multimedia Engine (GME).

This document only provides the main APIs to help you get started with GME to debug and integrate the APIs.

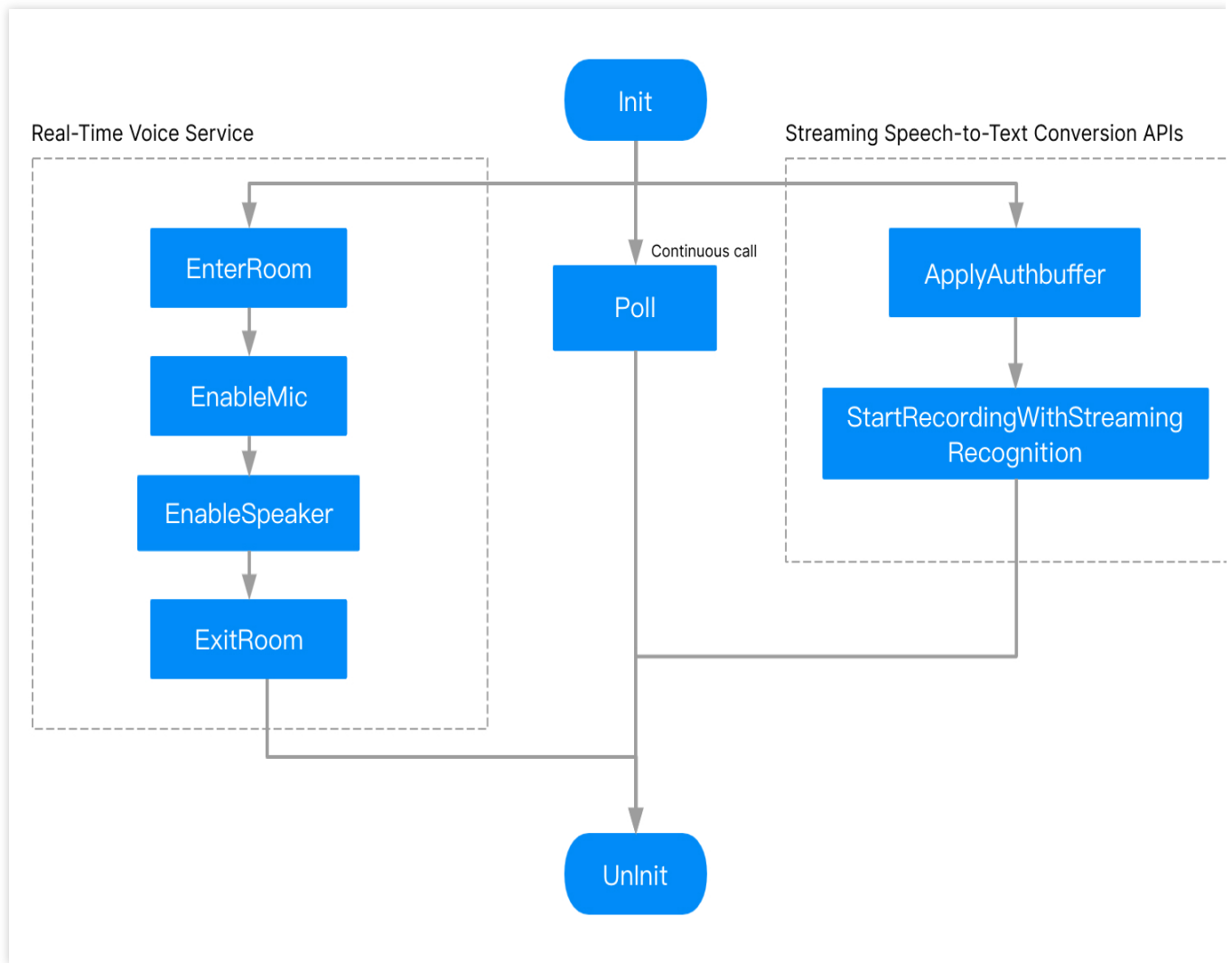
Key Considerations for Using GME

GME provides two services: Voice chat service and voice messaging and speech-to-text service, both of which rely on key APIs such as Init and Poll.

Note on Init API:

If you need to use voice chat and voice messaging services at the same time, **you only need to call `Init` API once.**

API call flowchart



Directions

Core APIs

Initializing GMEAPI: [Init](#)

Calling Poll periodically to trigger event callbacksAPI: [Poll](#)

Voice Chat

1. [Entering a voice chat room](#)API: [EnterRoom](#)
2. [Turning on or off the microphone](#)API: [EnableMic](#)
3. [Turning on or off the speaker](#)API: [EnableSpeaker](#)
4. [Exiting a voice room](#)API: [ExitRoom](#)

Voice Message

1. [Initializing authentication](#)API: [ApplyPTTAuthbuffer](#)
2. [Starting streaming speech recognition](#)API: [StartRecordingWithStreamingRecognition](#)

3. Stop recordingAPI: StopRecording

[Uninitializing GMEAPI: UnInit](#)

Core API Integration

1. Download the SDK

On the SDK download guide page, download the appropriate [client SDKDownload](#).

2. Importing the header file

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. Getting singleton

To use the voice feature, get the `ITMGContext` object first.

Function prototype

Java

Object-C

C++

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

```
+ (ITMGContext*) GetInstance;

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

Sample code

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. Setting callback

The API class uses the `Delegate` method to send callback notifications to the application. Register the callback function to the SDK for receiving callback messages before room entry.

Function prototype and sample code

Register the callback function to the SDK for receiving callback messages before room entry.

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];

// When initializing the SDK
m_pTmgContext = ITMGContextGetInstance();
m_pTmgContext->SetTMGDelegate(this);
// In the destructor
CTMGSDK_For_AudioDlg::~CTMGSDK_For_AudioDlg()
{
    ITMGContextGetInstance()->SetTMGDelegate(NULL);
}
```

Callback examples

Override this callback function in the constructor to process the parameters of the callback.

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)
    {
        // Processing callbacks
    }
}

// Refer to TMGCallbackDispatcher.java, TMGCallbackHelper.java, and TMGDispatcherBa

//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;
    }
}

// Refer to DispatchCenter.h and DispatchCenter.m

// Declaration in the header file
virtual void OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data);
// Sample code
void CTMGSDK_For_AudioDlg::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char* data
{
    switch(eventType)
    {
        case ITMG_MAIN_EVENT_TYPE_XXXX_XXXX:
        {
            // Process the callback
        }
        break;
    }
}
```

Parameter	Type	Description
type	ITMGContext.ITMG_MAIN_EVENT_TYPE	Event type in the callback response
data	Intent message type	Callback message, i.e., event data

5. Initializing SDK

You need to initialize the SDK through the `Init` API before you can use the real-time voice, voice message, and speech-to-text services. The `Init` API must be called in the same thread as other APIs. We recommend you call all APIs in the main thread.

API prototype

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)
```

Parameter	Type	Description
sdkAppId	string	<code>AppID</code> provided in the GME console , which can be obtained as instructed in Activating Services .
openID	string	<code>openID</code> can only be in <code>Int64</code> type, which is passed in after being converted to a string. You can customize its rules, and it must be unique in the application. To pass in <code>openID</code> as a string, submit a ticket for application.

Sample code

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) // SDK has been initialized. This operat
{
    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"
const char* openId="10001";
ITMGContext* context = ITMGContextGetInstance();
context->Init(SDKAPPID3RD, openId);

```

6. Triggering event callback

Event callbacks can be triggered by periodically calling the `Poll` API in `update`. The `Poll` API is GME's message pump and should be called periodically for GME to trigger event callbacks; otherwise, the entire SDK service will run abnormally. For more information, see the `EnginePollHelper` file in [SDK Download Guide](#).

Sample code

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);
    }
};

// For the code of calling Poll periodically, see EnginePollHelper.java.

//TMGSampleViewController.m
[EnginePollHelper createEnginePollHelper];
// Refer to EnginePollHelper.m and EnginePollHelper.h

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

7. Calculating the local authentication key

Generate `AuthBuffer` for encryption and authentication of relevant features. For release in the production environment, please use the backend deployment key as detailed in [Authentication Key](#).

API prototype

Java

Object-C

C++

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

//TMGSampleViewController.m
[EnginePollHelper createEnginePollHelper];
// Refer to EnginePollHelper.m and EnginePollHelper.h

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

Parameter	Type	Description
-----------	------	-------------

appld	int	<code>AppId</code> from the Tencent Cloud console.
roomId	string	Room ID, which can contain up to 127 characters (For voice message, enter "null".)
openId	string	User ID, which is the same as <code>openId</code> during initialization.
key	string	Permission key from the Tencent Cloud console .

Sample code

Java

Object-C

C++

```
//GMEAuthBufferHelper.java
import com.tencent.av.sig.AuthBuffer;// Header file
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;
}

// Voice chat authentication
NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:SDKAPPID3RD.intValue roomId:self.
// Voice message authentication
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
```

Voice Chat Access

1. Entering a room

This API is used to enter a room with the generated authentication information. The mic and speaker are not turned on by default after room entry. The returned value of `AV_OK` indicates successful API call but not successful room entry.

API prototype

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
```

Parameter	Type	Description
roomId	String	Room ID, which can contain up to 127 characters
roomType	int	Use <code>FLUENCY</code> sound quality to enter the room
authBuffer	byte[]	Authentication code

Sample code

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);
```

Callback for room entry

After the user enters the room, the message `ITMG_MAIN_EVENT_TYPE_ENTER_ROOM` will be sent and identified in the `OnEvent` function for callback and processing. A successful callback means that the room entry is successful, and the billing starts.

Billing references

[Purchase Guide](#)

[Billing FAQs](#)

[Will Voice Chat still be charged when client is offlined?](#)

Sample code Sample code for processing the callback, including room entry and network disconnection events.

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());
        }
    }
}
```

Error code

Error Code Value	Cause and Suggested Solution
7006	Authentication failed. Possible causes: The `AppID` does not exist or is incorrect. An error occurred while authenticating the `authbuff`. Authentication expired. The `openId` does not meet the specification.
7007	Already in another room.
1001	The user was already in the process of entering a room but repeated this operation. It is recommended not to call the room entering API until the room entry callback is returned.

1003	The user was already in the room and called the room entering API again.
1101	Make sure that the SDK is initialized, `openId` complies with the rules, the APIs are called in the same thread, and the `Poll` API is called normally.

2. Turning on or off the microphone

This API is used to turn on or off the mic. Mic and speaker are not enabled by default after room entry.

Sample code

Java

Object-C

C++

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

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

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

3. Turning on or off the speaker

This API is used to turn on/off the speaker.

Sample code

Java

Object-C

C++

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

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

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

4. Exiting the room

This API is called to exit the current room. It needs to wait for and process the callback for exit.

Sample code

Java

Object-C

C++

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

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

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

Callback for room exit

After the user exits a room, a callback will be returned with the message being

`ITMG_MAIN_EVENT_TYPE_EXIT_ROOM`. The sample code is shown below:

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)
    {
        // Receive the event of successful room exit
    }
}

//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:
        {
            // Receive the event of successful room exit

        }
        break;
    }
}

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

Voice Message Access

1. Initializing authentication

Call authentication initialization after initializing the SDK. For more information on how to get the `authBuffer` , please see `genAuthBuffer` (the voice chat authentication information API).

API prototype

Java

Object-C

C++

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

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

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

Parameter	Type	Description
authBuffer	String	Authentication

Sample code

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.integer
[[[ITMGContext GetInstance] GetPTT] ApplyPTTAuthbuffer:authBuffer];

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

2. Starting streaming speech recognition

This API is used to start streaming speech recognition. Text obtained from speech-to-text conversion will be returned in real time in its callback. **To stop recording, call `StopRecording`**. The callback will be returned after the recording is stopped.

API prototype

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()
```

Parameter	Type	Description
filePath	String	Path of stored audio file

Sample code

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 = @"Start streaming recording";
} else {
    self.currentStatus = @"Failed to start streaming recording";
}

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

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen for callback messages in the callback function

`onEvent`. The event message is `ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE`, namely returns text after the recording is stopped and the recognition is completed, which is equivalent to returning the recognized text after a paragraph of speech.

The event message will be identified in the `OnEvent` function based on the actual needs. The passed parameters include the following four messages.

--	--

Message Name	Description
result	A return code for judging whether the streaming speech recognition is successful.
text	Text converted from speech
file_path	Local path of stored recording file
file_id	Backend URL address of recording file, which will be retained for 90 days

Sample code

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 = @"Streaming conversion completed";
        }
    }
    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  
}
```

```
}  
}
```

Error code

Error Code	Description	Suggested Solution
32775	Streaming speech-to-text conversion failed, but recording succeeded.	Call the `UploadRecordedFile` API to upload the recording file and then call the `SpeechToText` API to perform speech-to-text conversion.
32777	Streaming speech-to-text converting failed, but recording and upload succeeded	The message returned contains a backend URL after successful upload. Call the `SpeechToText` API to perform speech-to-text conversion.
32786	Streaming speech-to-text conversion failed.	During streaming recording, wait for the execution result of the streaming recording API to return.

3. Stopping recording

This API is used to stop recording. It is async, and a callback for recording completion will be returned after recording stops. A recording file will be available only after recording succeeds.

API prototype

Java

Object-C

C++

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

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

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

Sample code

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_COMPLETE
    // https://intl.cloud.tencent.com/document/product/607/15221
    QAVResult ret = [[[ITMGContext GetInstance] GetPTT] StopRecording];
    if (ret == 0) {
        self.currentStatus = @"Stop recording";
    } else {
        self.currentStatus = @"Failed to stop recording";
    }
}

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

Quick Integration of SDK for Unity

Last updated : 2024-12-18 14:51:59

This document provides a detailed description that makes it easy for Unity project developers to debug and integrate the APIs for Game Multimedia Engine (GME).

This document only provides the main APIs to help you get started with GME to debug and integrate the APIs.

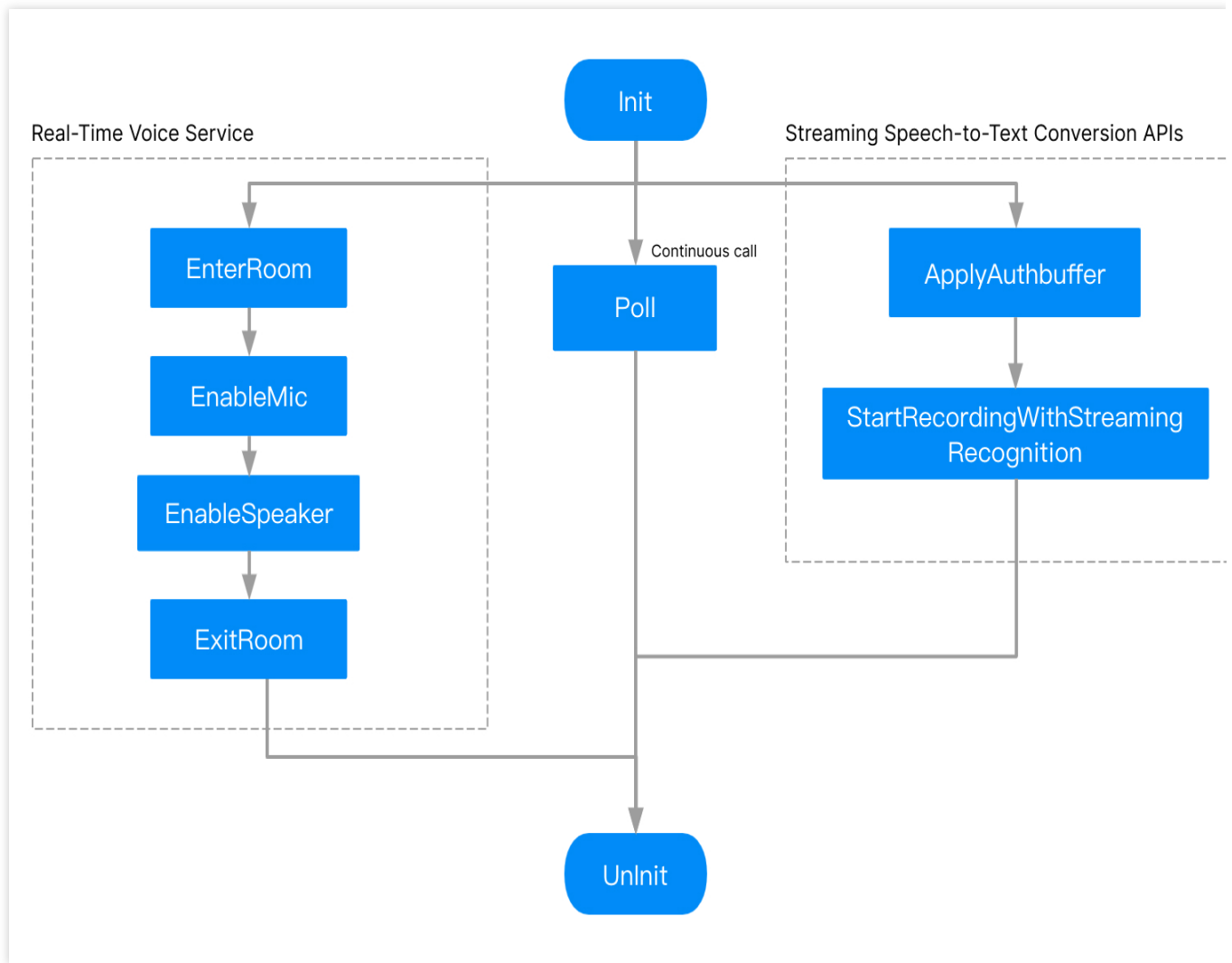
Key Considerations for Using GME

GME provides two services: Voice chat service and voice messaging and speech-to-text service, both of which rely on key APIs such as Init and Poll.

Note on Init API:

If you need to use voice chat and voice messaging services at the same time, **you only need to call `Init` API once.**

API call flowchart



Directions

Integrating SDK

To integrate the SDK into the project, see [Integrating SDK](#).

Core APIs

Initializing GMEAPI: [Init](#)

Calling Poll periodically to trigger event callbacksAPI: [Poll](#)

Listening on room entry/exit notificationListener: [QAVEnterRoomComplete](#)

Voice Chat

1. [Entering a voice chat room](#)API: [EnterRoom](#)
2. [Turning on or off the microphone](#)API: [EnableMic](#)
3. [Turning on or off the speaker](#)API: [EnableSpeaker](#)
4. [Exiting a voice room](#)API: [ExitRoom](#)

Voice Message

1. [Initializing authenticationAPI: ApplyPTTAuthbuffer](#)
 2. [Starting streaming speech recognitionAPI: StartRecordingWithStreamingRecognition](#)
 3. [Stop recordingAPI: StopRecording](#)
- [Uninitializing GMEAPI: UnInit](#)

Key API Access

1. Download the SDK

On the SDK download guide page, download the appropriate [client SDKDownload](#).

2. Importing the header file

```
using GME;
```

3. Getting the Context instance

Get the `Context` instance by using the `ITMGContext` method instead of `QAVContext.GetInstance()`.

Sample code

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

4. Initializing SDK

You need to initialize the SDK through the `Init` API before you can use the real-time voice, voice message, and speech-to-text services. The `Init` API must be called in the same thread as other APIs. We recommend you call all APIs in the main thread.

API prototype

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

Parameter	Type	Description
sdAppId	string	<code>AppID</code> provided in the GME console , which can be obtained as instructed in Activating Services .
openID	string	<code>openID</code> can only be in <code>Int64</code> type, which is passed in after being converted to a string. You can customize its rules, and it must be unique in the application. To

pass in `openID` as a string, [submit a ticket](#) for application.

Sample code

```
int ret = ITMGContext.GetInstance().Init(sdkAppId, openID);
// Determine whether the initialization is successful by the returned value
if (ret != QAVError.OK)
{
    Debug.Log("SDK initialization failed:"+ret);
    return;
}
```

5. Triggering event callback

Event callbacks can be triggered by periodically calling the `Poll` API in `update`. The `Poll` API is GME's message pump and should be called periodically for GME to trigger event callbacks; otherwise, the entire SDK service will run abnormally. For more information, see the `EnginePollHelper` file in [SDK Download Guide](#).

Sample code

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

6. Listening on room entry/exit notification

Room entry notification

```
// Delegate function:
public delegate void QAVEnterRoomComplete(int result, string error_info);
// Event-triggered function:
public abstract event QAVEnterRoomComplete OnEnterRoomCompleteEvent;
```

Room exit notification

```
Delegate function:
public delegate void QAVExitRoomComplete();
Event-triggered function:
public abstract event QAVExitRoomComplete OnExitRoomCompleteEvent;
```

7. Calculating the local authentication key

Generate `AuthBuffer` for encryption and authentication of relevant features. For release in the production environment, please use the backend deployment key as detailed in [Authentication Key](#).

API prototype

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

Parameter	Type	Description
appId	int	<code>AppId</code> from the Tencent Cloud console.
roomId	string	Room ID, which can contain up to 127 characters (For voice message, enter "null".)
openId	string	User ID, which is the same as <code>openId</code> during initialization.
key	string	Permission key from the Tencent Cloud console .

Sample code

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

Voice Chat Access

1. Entering a room

This API is used to enter a room with the generated authentication information. The mic and speaker are not enabled by default after room entry. The returned value of `AV_OK` indicates successful API call but not successful room entry.

API prototype

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

Parameter	Type	Description
roomId	String	Room ID, which can contain up to 127 characters
roomType	ITMGRoomType	Just enter <code>ITMGRoomType.ITMG_ROOM_TYPE_FLUENCY</code>

authBuffer	byte[]	Authentication code
------------	--------	---------------------

Sample code

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

Callback for room entry

After the user enters the room, the room entry result will be called back, which can be listened on for processing. A successful callback means that the room entry is successful, and the billing **starts**.

Billing references

[Purchase Guide](#)

[Billing FAQs](#)

[Will Voice Chat still be charged when client is offlined?](#)

Sample codeSample code for processing the callback:

```
// Listen on an event:  
ITMGContext.GetInstance().OnEnterRoomCompleteEvent += new  
QAVEnterRoomComplete(OnEnterRoomComplete);  
// Process the event listened on:  
void OnEnterRoomComplete(int err, string errInfo)  
{  
    if (err != 0) {  
        ShowLoginPanel("error code:" + err + " error message:" + errInfo);  
        return;  
    }  
    else{  
        // Entered room successfully  
    }  
}
```

Error code

Error Code Value	Cause and Suggested Solution
7006	Authentication failed. Possible causes: The `AppID` does not exist or is incorrect. An error occurred while authenticating the `authbuff`. Authentication expired. The `openId` does not meet the specification.
7007	Already in another room.

1001	The user was already in the process of entering a room but repeated this operation. It is recommended not to call the room entering API until the room entry callback is returned.
1003	The user was already in the room and called the room entering API again.
1101	Make sure that the SDK is initialized, `openId` complies with the rules, the APIs are called in the same thread, and the `Poll` API is called normally.

2. Turning on or off the microphone

This API is used to turn on or off the mic. Mic and speaker are not turned on by default after room entry.

Sample code

```
// Listen on an event:
ITMGContext.GetInstance().OnEnterRoomCompleteEvent += new
QAVEnterRoomComplete(OnEnterRoomComplete);
// Process the event listened on:
void OnEnterRoomComplete(int err, string errInfo)
{
    if (err != 0) {
        ShowLoginPanel("error code:" + err + " error message:" + errInfo);
        return;
    }
    else{
        // Entered room successfully
        // Turn on mic
        ITMGContext.GetInstance().GetAudioCtrl().EnableMic(true);
    }
}
```

3. Turning on or off the speaker

This API is used to turn on/off the speaker.

Sample code

```
// Listen on an event:
ITMGContext.GetInstance().OnEnterRoomCompleteEvent += new
QAVEnterRoomComplete(OnEnterRoomComplete);
// Process the event listened on:
void OnEnterRoomComplete(int err, string errInfo)
{
    if (err != 0) {
        ShowLoginPanel("error code:" + err + " error message:" + errInfo);
        return;
    }
}
```



```
}
else{
    // Entered room successfully
    // Turn on the speaker
    ITMGContext.GetInstance().GetAudioCtrl().EnableSpeaker(true);
}
}
```

4. Exiting the room

This API is called to exit the current room. It needs to wait for and process the callback for exit.

Sample code

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

Callback for room exit

After the user exits a room, a callback will be returned. The sample code is as shown below:

```
Listen on an event:
ITMGContext.GetInstance().OnExitRoomCompleteEvent += new
QAVExitRoomComplete(OnExitRoomComplete);
Process the event listened on:
void OnExitRoomComplete(){
    // Send a callback after room exit
}
```

Voice Message Access

1. Initializing authentication

Call authentication initialization after initializing the SDK. For more information on how to get the `authBuffer`, please see `genAuthBuffer` (the voice chat authentication information API).

API prototype

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

Parameter	Type	Description
authBuffer	String	Authentication

Sample code

```
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.GetUserID(), null, UserConfig.GetAuthKey());
ITMGContext.GetInstance ().GetPttCtrl ().ApplyPTTAuthbuffer(authBuffer);
```

2. Starting streaming speech recognition

This API is used to start streaming speech recognition. Text obtained from speech-to-text conversion will be returned in real time in its callback. **To stop recording, call `StopRecording`** . The callback will be returned after the recording is stopped.

API prototype

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

Parameter	Type	Description
filePath	String	Path of stored audio file

Sample code

```
string recordPath = Application.persistentDataPath + string.Format("/{0}.silk",
sUid++);
int ret =
ITMGContext.GetInstance ().GetPttCtrl ().StartRecordingWithStreamingRecognition(r
ecordPath);
```

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen on callback messages in the

`OnStreamingSpeechComplete` or `OnStreamingSpeechisRunning` notification, which is as detailed below:

`OnStreamingSpeechComplete` returns text after the recording is stopped and the recognition is completed, which is equivalent to returning the recognized text after a paragraph of speech.

`OnStreamingSpeechisRunning` returns the recognized text in real time during the recording, which is equivalent to returning the recognized text while speaking.

The event message will be identified in the `OnEvent` function based on the actual needs. The passed parameters include the following four messages.

Message Name	Description
result	A return code for judging whether the streaming speech recognition is successful.
text	Text converted from speech
file_path	Local path of stored recording file
file_id	Backend URL address of recording file, which will be retained for 90 days

Sample code

```
// Listen on an event:
ITMGContext.GetInstance().GetPttCtrl().OnStreamingSpeechComplete +=new
QAVStreamingRecognitionCallback (OnStreamingSpeechComplete);
ITMGContext.GetInstance().GetPttCtrl().OnStreamingSpeechisRunning += new
QAVStreamingRecognitionCallback (OnStreamingRecisRunning);
// Process the event listened on:
void OnStreamingSpeechComplete(int code, string fileid, string filepath, string
result){
    // Callback for streaming speech recognition
}
void OnStreamingRecisRunning(int code, string fileid, string filePath, string
result){
    if (code == 0)
    {
        setBtnText(mStreamBtn, "Streaming");
        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("Recording");
    }
}
```

Error code

Error Code	Description	Suggested Solution
32775	Streaming speech-to-text	Call the `UploadRecordedFile` API to upload the

	conversion failed, but recording succeeded.	recording file and then call the `SpeechToText` API to perform speech-to-text conversion.
32777	Streaming speech-to-text converting failed, but recording and upload succeeded	The message returned contains a backend URL after successful upload. Call the `SpeechToText` API to perform speech-to-text conversion.
32786	Streaming speech-to-text conversion failed.	During streaming recording, wait for the execution result of the streaming recording API to return.

3. Stopping recording

This API is used to stop recording. It is async, and a callback for recording completion will be returned after recording stops. A recording file will be available only after recording succeeds.

API prototype

```
ITMGPTT int StopRecording()
```

Sample code

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

Quick Integration of SDK for Unreal Engine

Last updated : 2024-12-18 14:52:35

This document provides a detailed description that makes it easy for Unreal Engine project developers to debug and integrate the APIs for Game Multimedia Engine (GME).

This document only provides the main APIs to help you get started with GME to debug and integrate the APIs.

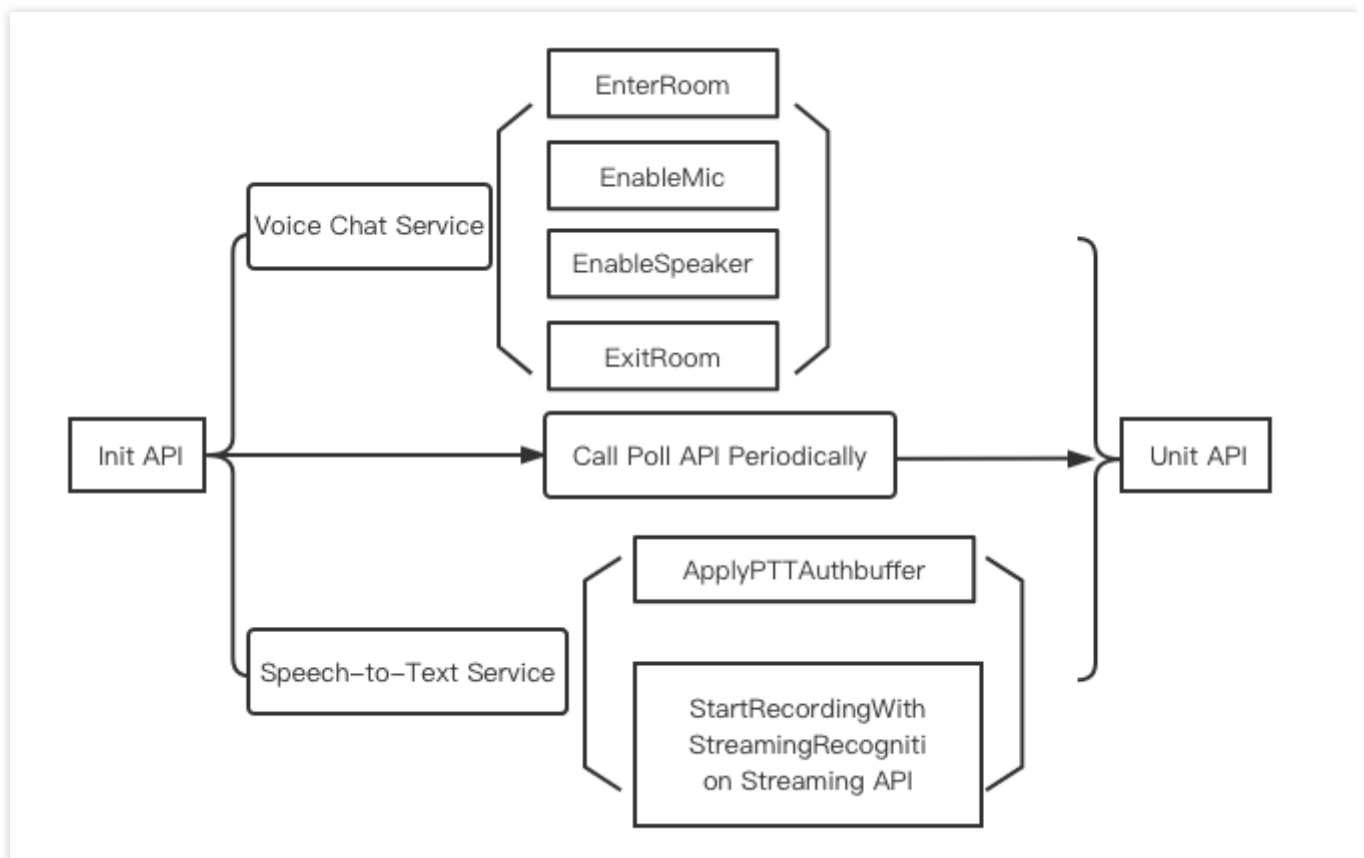
Key Considerations for Using GME

GME provides two services: voice chat service and voice message and speech-to-text service, both of which rely on key APIs such as Init and Poll.

Note on Init API:

If you need to use voice chat and voice message services at the same time, **you only need to call `Init` API once.**

API call flowchart



Directions

Integrating SDK

Refer to [Integrating SDK](#) to integrate the SDK into the project.

Key APIs

[Initializing GMEAPI: Init](#)

[Calling Poll periodically to trigger event callbacksAPI: Poll](#)

[Listening on room entry/exit notificationListener: QAVEnterRoomComplete](#)

Voice Chat

1. [Entering a voice chat roomAPI: EnterRoom](#)
2. [Enabling or disabling the microphoneAPI: EnableMic](#)
3. [Enabling or disabling the speakerAPI: EnableSpeaker](#)
4. [Exiting a voice roomAPI: ExitRoom](#)

Voice Message

1. [Initializing authenticationAPI: ApplyPTTAuthbuffer](#)
2. [Starting streaming speech recognitionAPI: StartRecordingWithStreamingRecognition](#)
3. [Stop recordingAPI: StopRecording](#)

[Uninitializing GMEAPI: UnInit](#)

Key API Access

1. Downloading the SDK

On the SDK download guide page, download the appropriate [client SDKDownload](#).

2. Importing the header file

```
#include "tmg_sdk.h"

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

3. Setting the singleton

You need to get `ITMGContext` first before you can call the `EnterRoom` function, because all calls begin with `ITMGContext` and callbacks are passed to the application through `ITMGDelegate`.

Sample code

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

4. Initializing SDK

This API is used to initialize the GME service. It is recommended to call it when initializing the application.

The openID uniquely identifies a user with the rules stipulated by the application developer and unique in the application (currently, only INT64 is supported).

If the user switches the login account, they need to call `Uninit` and then call `Init` again with the new `OpenId`.

Function prototype

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

Parameter	Type	Description
sdkAppId	const char*	<code>AppId</code> provided by the GME service from the Tencent Cloud console
OpenId	const char*	<code>OpenId</code> can only be in Int64 type, which is passed after being converted to a string.

Sample code

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

5. Triggering event callback

Event callbacks can be triggered by periodically calling the `Poll` API in `update`. The `Poll` API should be called periodically for GME to trigger event callbacks; otherwise, the entire SDK service will run exceptionally.

Refer to the `UEDemoLevelScriptActor.cpp` file in the demo.

Sample code

```
// Declaration in the header file
virtual void Tick(float DeltaSeconds);

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

6. Setting the callback

The API class uses the `Delegate` method to send callback notifications to the application.

`ITMG_MAIN_EVENT_TYPE` indicates the message type. The data on Windows is in json string format. For the key-value pairs, please see the relevant documentation.

Sample code

```
// Function implementation:
//UEDemoLevelScriptActor.h:
class UEDEMO1_API AUEDemoLevelScriptActor : public ALevelScriptActor, public
SetTMGDelegate
{
public:
    void OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char* data);
}

//UEDemoLevelScriptActor.cpp:
void AUEDemoLevelScriptActor::OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const
char* data){
    // Identify and manipulate `eventType` here
}
```

7. Authentication

Generate `AuthBuffer` for encryption and authentication of relevant features. To get authentication for voice message and speech-to-text, the room ID parameter must be set to `null`.

Function prototype

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


Parameter	Type	Description
dwSdkAppID	int	<code>AppId</code> from the Tencent Cloud console.
strRoomID	char*	Room ID, which can contain up to 127 characters.
strOpenID	char*	User ID, which is the same as <code>openID</code> during initialization.
strKey	char*	Permission key from the Tencent Cloud console .
strAuthBuffer	char*	Returned <code>authbuff</code>
bufferLength	int	Length of the <code>authbuff</code> passed in. 500 is recommended.

Sample code

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

Voice Chat Access

1. Entering a room

This API is used to enter a room with the generated authentication information. The mic and speaker are not enabled by default after room entry. The returned value of 0 indicates successful API call but not successful room entry. For more information on how to choose a room audio type, please see [Sound Quality Selection](#).

Function prototype

```
ITMGContext virtual int EnterRoom(const char* roomId, ITMG_ROOM_TYPE roomType,
const char* authBuff, int buffLen)
```

Parameter	Type	Description
roomId	char*	Room ID, which can contain up to 127 characters
roomType	ITMG_ROOM_TYPE	Room audio type
authBuffer	char*	Authentication key

buffLen	int	Authentication key length
---------	-----	---------------------------

Sample code

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

Callback for room entry

After the user enters the room, a room entry notification will be received and identified in the listener function for processing. A successful callback (err = 0) means that the room entry is successful, and the **billing** starts. If the total call duration on the day is below 700 minutes, no fees will be incurred.

Billing references

[Purchase Guide](#)

[Billing FAQs](#)

[Will the billing continue if the client is disconnected when using the voice chat?](#)

Sample codeSample code for processing the callback:

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

FString jsonData = FString(UTF8_TO_TCHAR(data));
TSharedPtr<FJsonObject> JsonObject;
TSharedPtr<TJsonReader<>> Reader =
TJsonReaderFactory<>::Create(FString(UTF8_TO_TCHAR(data)));
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 room success.));
    }
    else {
        FString msg = FString::Printf(TEXT("Enter room failed. result=%d, info =
%ls"), result, *error_info);
        GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, *msg);
    }
    onEnterRoomCompleted(result, error_info);
}
}
```

Error code

Error Code Value	Cause and Suggested Solution
7006	Authentication failed. Possible causes: The `AppID` does not exist or is incorrect. An error occurred while authenticating the `authbuff`. Authentication expired. The `openId` does not meet the specification.
7007	Already in another room.
1001	The user was already in the process of entering a room but repeated this operation. It is recommended not to call the room entering API until the room entry callback is returned.
1003	The user was already in the room and called the room entering API again.
1101	Make sure that the SDK is initialized, `openId` complies with the rules, the APIs are called in the same thread, and the `Poll` API is called normally.

2. Enabling or disabling the microphone

This API is used to enable/disable the mic. Mic and speaker are not enabled by default after room entry.

Sample code

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

    FString jsonData = FString(UTF8_TO_TCHAR(data));
    TSharedPtr<FJsonObject> JsonObject;
    TSharedPtr<TJsonReader<>> Reader =
TJsonReaderFactory<>::Create(FString(UTF8_TO_TCHAR(data)));
    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 room success.));
            // Enable mic
            ITMGContextGetInstance()->GetAudioCtrl()->EnableMic(true);
        }
    }
    else {
```

```
FString msg = FString::Printf(TEXT("Enter room failed. result=%d, info = %ls"), result, *error_info);
GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, *msg);
}
onEnterRoomCompleted(result, error_info);
}
}
```

3. Enabling or disabling the speaker

This API is used to enable/disable the speaker.

Sample code

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

    FString jsonData = FString(UTF8_TO_TCHAR(data));
    TSharedPtr<FJsonObject> JsonObject;
    TSharedPtr<TJsonReader<>> Reader =
TJsonReaderFactory<>::Create(FString(UTF8_TO_TCHAR(data)));
    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 room success.));
            // Enable the speaker
            ITMGContextGetInstance()->GetAudioCtrl()->EnableSpeaker(true);
        }
        else {
            FString msg = FString::Printf(TEXT("Enter room failed. result=%d, info = %ls"), result, *error_info);
            GEngine->AddOnScreenDebugMessage(INDEX_NONE, 20.0f, FColor::Yellow, *msg);
        }
        onEnterRoomCompleted(result, error_info);
    }
}
```

4. Exiting the room

This API is called to exit the current room. It needs to wait for and process the callback for exit.

Sample code

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

Callback for room exit

After the user exits a room, a callback will be returned. The sample code is as shown below:

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

Voice Message Access

1. Initializing authentication

Call authentication initialization after initializing the SDK. For more information on how to get the `authBuffer`, please see `genAuthBuffer` (the voice chat authentication information API).

Function prototype

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

Parameter	Type	Description
authBuffer	char*	Authentication
authBufferLen	int	Authentication length

Sample code

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

2. Starting streaming speech recognition

This API is used to start streaming speech recognition. Text obtained from speech-to-text conversion will be returned in real time in its callback. It can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation. **To stop recording, call `StopRecording`**. The callback will be returned after the recording is stopped.

Function prototype

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

Parameter	Type	Description
filePath	char*	Path of stored audio file
speechLanguage	char*	The language in which the audio file is to be converted to text. For parameters, please see Language Parameter Reference List
translateLanguage	char*	The language into which the audio file will be translated. For parameters, please see Language Parameter Reference List (This parameter is currently unavailable. Enter the same value as that of <code>speechLanguage</code>)

Sample code

```
ITMGContextGetInstance()->GetPTT()-  
>StartRecordingWithStreamingRecognition(filePath,"cmn-Hans-CN","cmn-Hans-CN");
```

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen for callback messages in the callback function `onEvent`. The event message is `ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE`, namely returns text after the recording is stopped and the recognition is completed, which is equivalent to returning the recognized text after a paragraph of speech.

The event message will be identified in the `OnEvent` function based on the actual needs. The passed parameters include the following four messages.

Message Name	Description
result	A return code for judging whether the streaming speech recognition is successful.
text	Text converted from speech
file_path	Local path of stored recording file
file_id	Backend URL address of recording file, which will be retained for 90 days

Sample code

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

FString jsonData = FString(UTF8_TO_TCHAR(data));
TSharedPtr<FJsonObject> JsonObject;
TSharedPtr<TJsonReader<>> Reader =
TJsonReaderFactory<>::Create(FString(UTF8_TO_TCHAR(data)));
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);
}
}
```

Error code

Error Code	Description	Suggested Solution

32775	Streaming speech-to-text conversion failed, but recording succeeded.	Call the `UploadRecordedFile` API to upload the recording file and then call the `SpeechToText` API to perform speech-to-text conversion.
32777	Streaming speech-to-text converting failed, but recording and upload succeeded	The message returned contains a backend URL after successful upload. Call the `SpeechToText` API to perform speech-to-text conversion.
32786	Streaming speech-to-text conversion failed.	During streaming recording, wait for the execution result of the streaming recording API to return.

3. Stopping recording

This API is used to stop recording. It is async, and a callback for recording completion will be returned after recording stops. A recording file will be available only after recording succeeds.

Function prototype

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

Sample code

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


Quick Integration of Sample Project

Quick Run of Unreal Engine Sample Project

Last updated : 2024-01-18 11:53:35

This document describes how to quickly run GME Unreal Engine sample project and integrate the sample code to a project.

Running the Unreal Engine Sample Project

Environment requirements

Unreal Engine 4.22 or later

Microsoft Visual Studio

A configuration environment that can run Unreal Engine projects

Prerequisites

You need to activate the voice chat and voice messaging services of GME and get the `AppId` and `Key` in advance. For more information on how to apply for GME services, see [Activating Services](#). `appId` is the `AppID` and `authKey` is the permission key in the console.

Directions

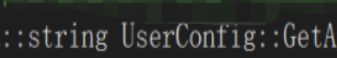
Step 1. Download the project

Download the Unreal Engine sample project as instructed in [SDK Download Guide](#). As the demo configurations for UE5 and UE4 are different, you need to download the sample project for the corresponding engine version.

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

Step 2. Configure the project

After downloading, open the project directory, find `UserConfig.cpp` in the `Source\\UEDemo1` path, and change the `appID` and `appKey` in the red box as shown below to the `AppID` and permission key applied for in [Service Management > Application Settings](#) in the GME console.

```
//   
//  https://console.cloud.tencent.com/gamegame  
//   
std::string UserConfig::GetAppID() {  
    FString appID;  
    GConfig->GetString(*UserInfoSection, TEXT("AppID"), appID, GGameIni);  
    if (appID.IsEmpty())  
    {  
        appID = "xxxxxxx";  
    }  
    return TCHAR_TO_UTF8(*appID);  
}  
  
void UserConfig::SetAppKey(const std::string& appKey) {  
    GConfig->SetString(*UserInfoSection, TEXT("AppKey"), UTF8_TO_TCHAR(appKey.c_str()), GGameIni);  
    GConfig->Flush(false, GGameIni);  
}  
  
std::string UserConfig::GetAppKey() {  
    FString appKey;  
    GConfig->GetString(*UserInfoSection, TEXT("AppKey"), appKey, GGameIni);  
    if (appKey.IsEmpty())  
    {  
        appKey = "xxxxxxxxx";  
    }  
    return TCHAR_TO_UTF8(*appKey);  
}
```

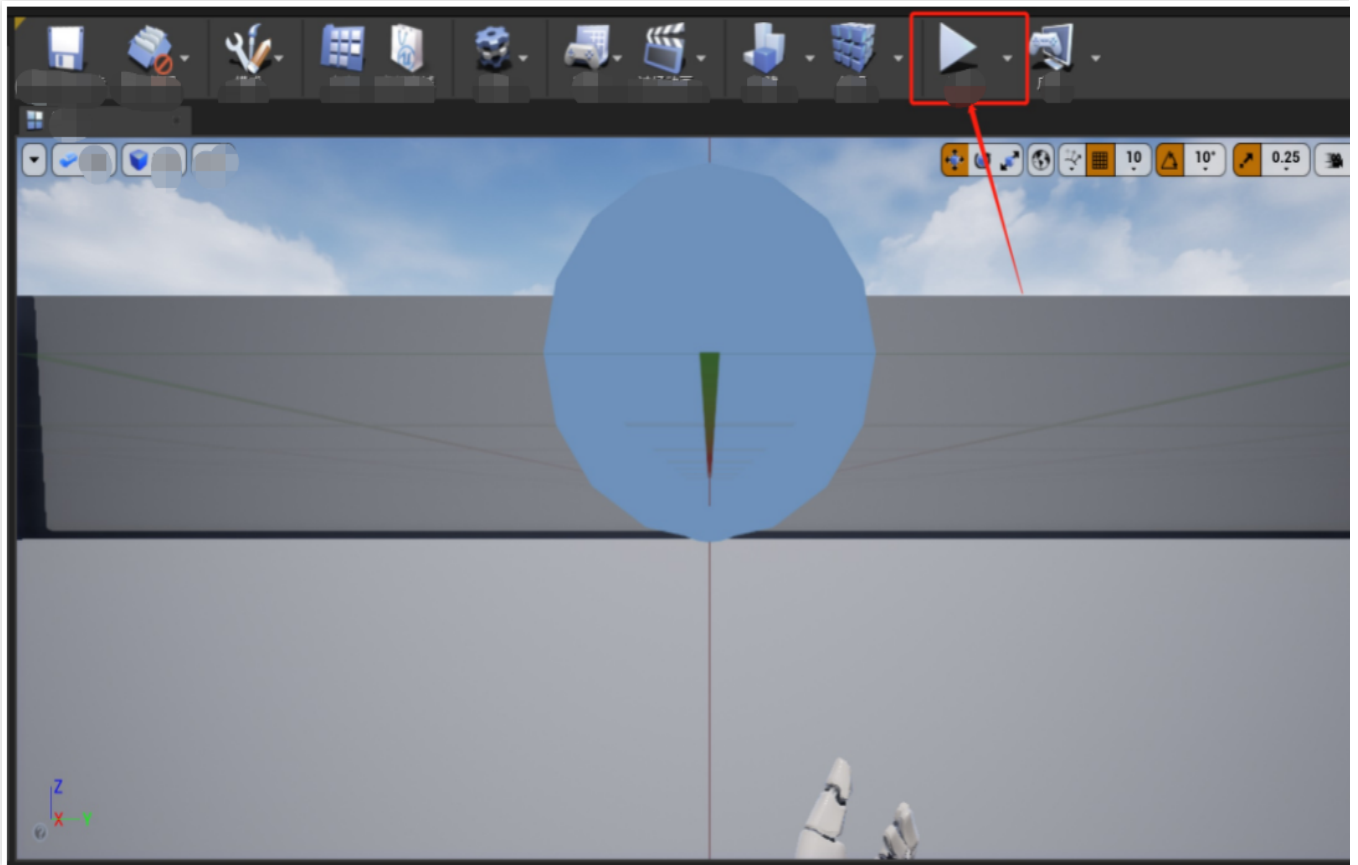
Step 3. Compile and run the demo

1. Run the program

Click



in the Editor to run the program.



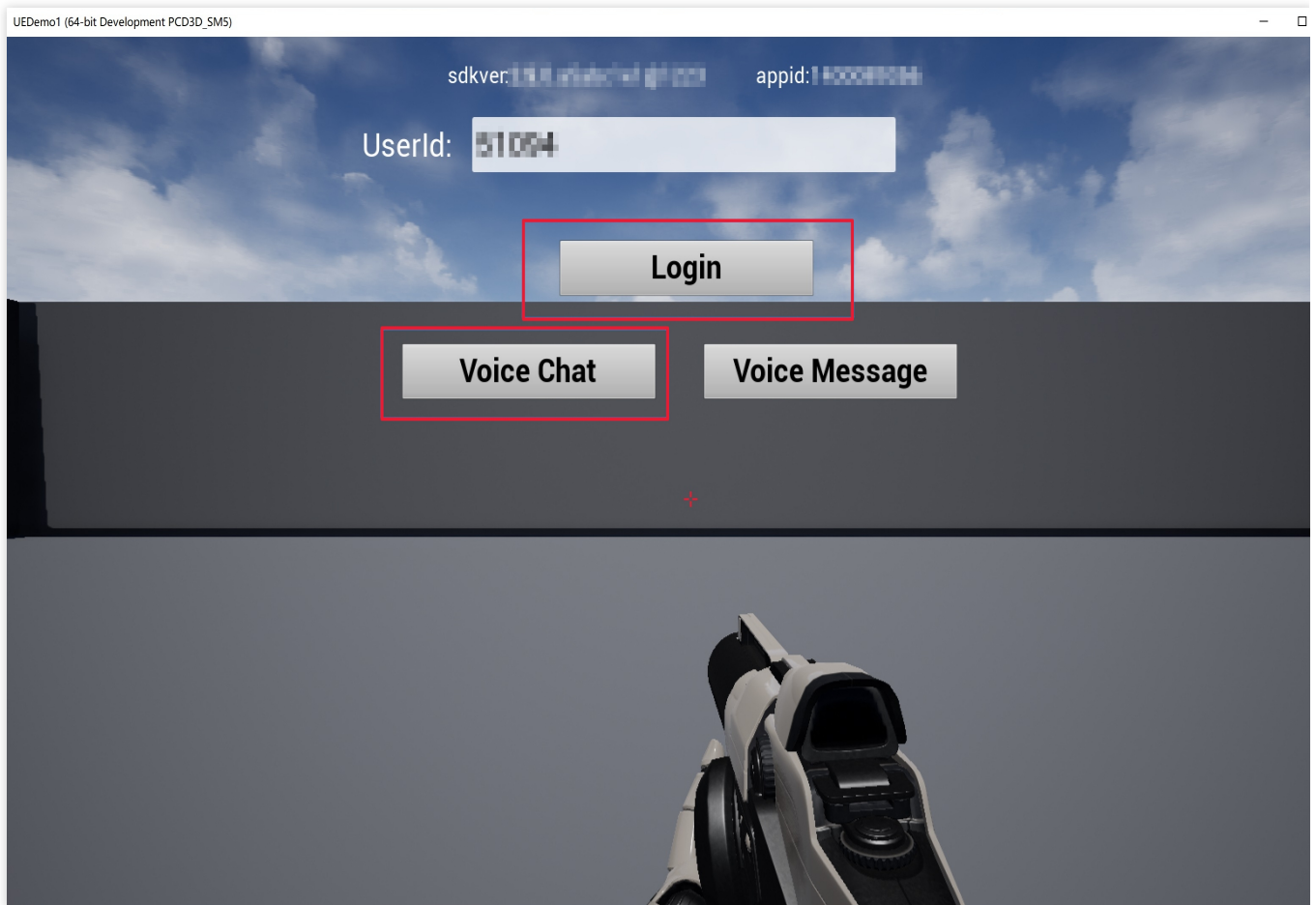
2. Initialize

UserID: It is equivalent to `openID` , which is the unique identifier of a user in the application. The `openID` value must be unique on each terminal.

Voice Chat: voice chat feature UI.

Voice Messaging: voice messaging feature UI.

Click **Login** to initialize, and then click **Voice Chat** to enter the voice chat room configuration page.



3. Enter a voice chat room

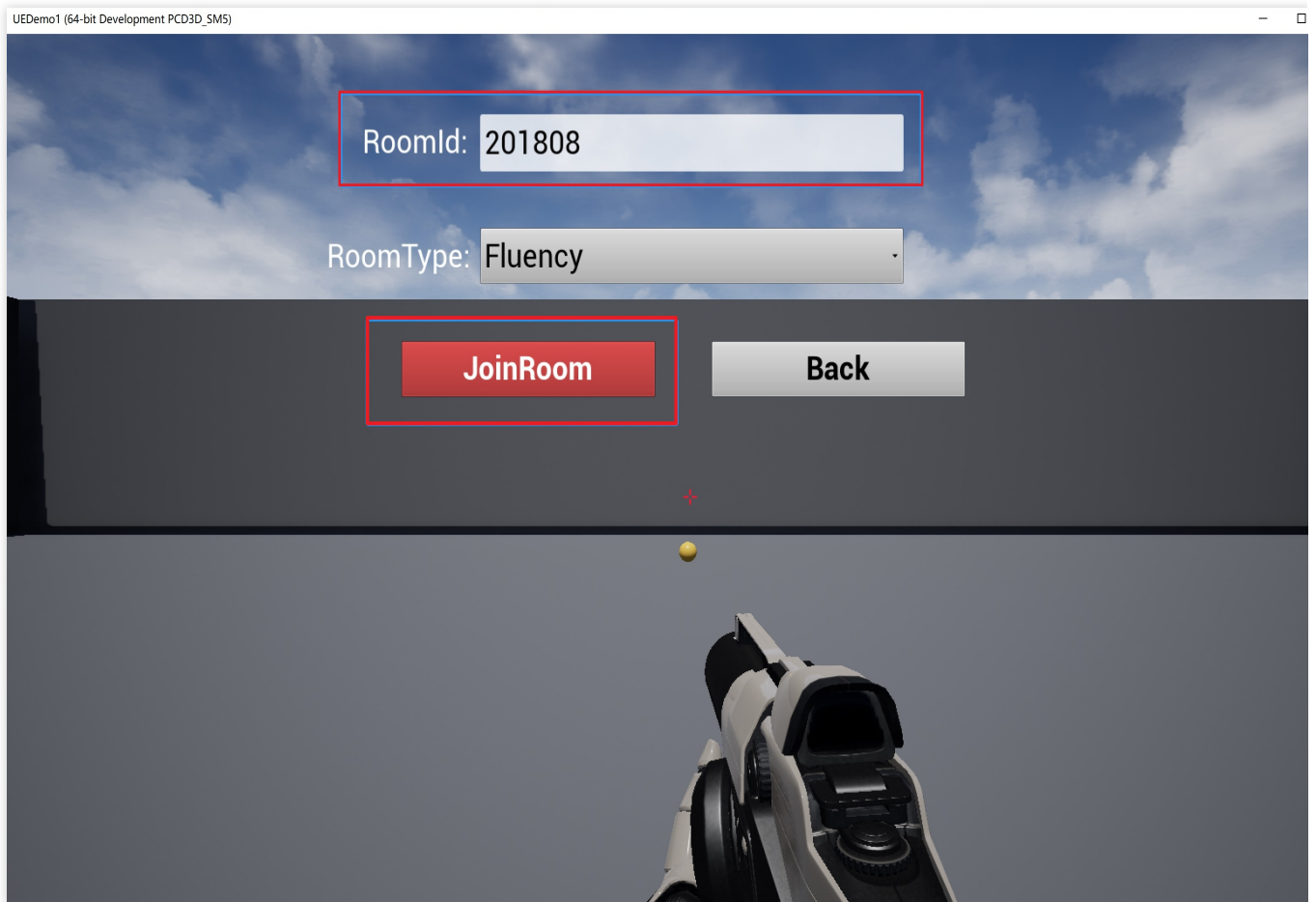
RoomId: Room ID. Users in the same room can communicate with each other by voice.

RoomType: Use Fluency to enter the room.

JoinRoom: Enter the voice room.

Back: Go back to the previous page.

After configuring the voice chat room ID, click **JoinRoom** to enter the room.



4. Use voice chat

The page will display the `RoomID` for room entry and the local `openID` .

Mic: Select to turn on the mic.

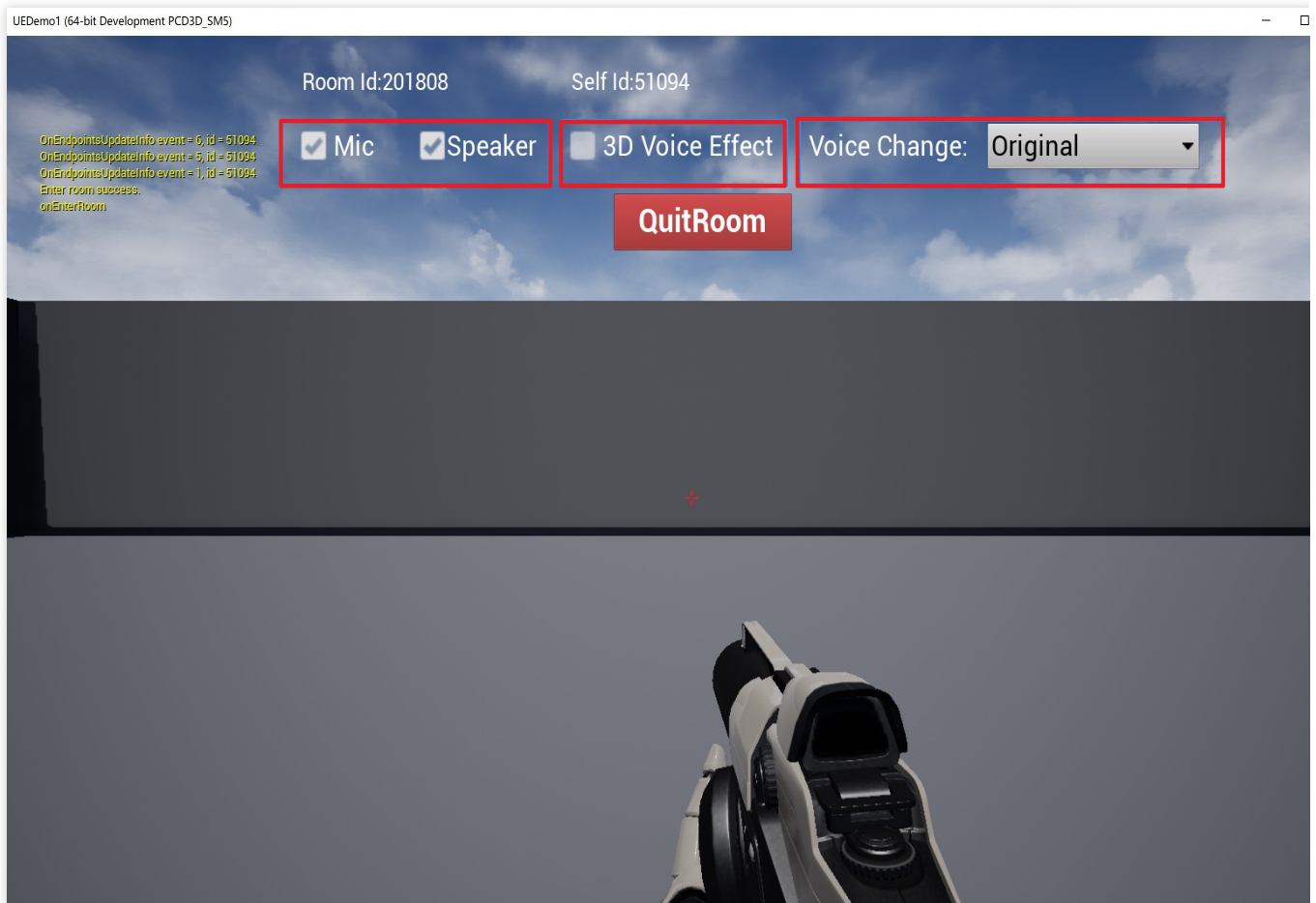
Speaker: Select to turn on the speaker.

3D Voice Effect: Select to enable 3D sound effects.

Voice Chang: Select to enable voice changing effects.

After the mic and speaker are selected locally, repeat the above steps on another device to enter the same room and turn on the mic and speaker, so that communication can be implemented.

If `3D Voice Effect` is selected on both terminals, use the A, S, D, and W keys to move around and experience the directional 3D stereo effect.



5. Use voice messaging

Language: Select the target language for text conversion. For example, if you speak Chinese, choose Mandarin.

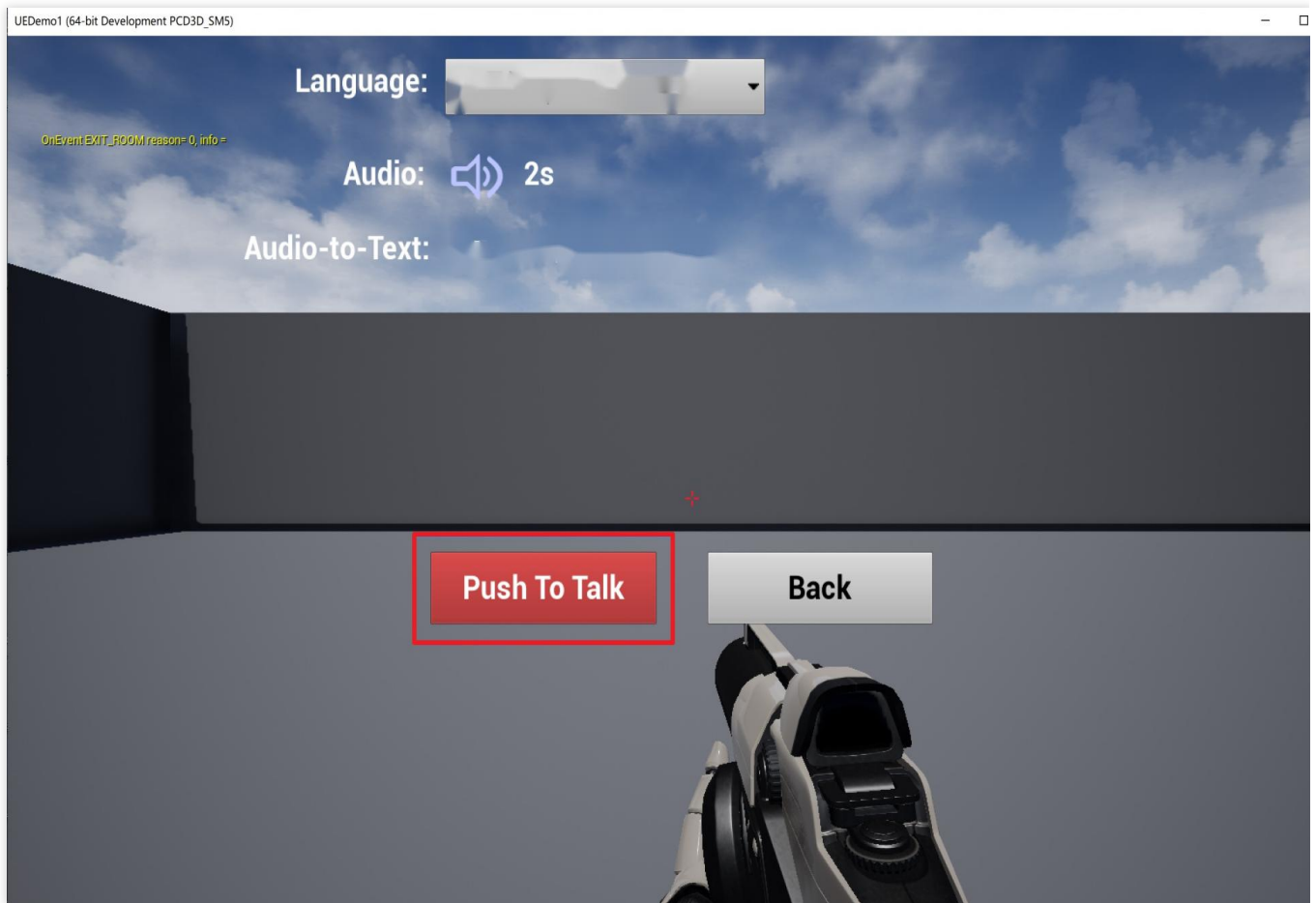
Audio: Click to listen after recording.

Audio-to-Text: Text content of the voice message.

Push To Talk: Press and hold to record.

Back: Go back to the previous page.

Press and hold **Push to Talk** and speak into the mic. After you release the button, your voice message will be converted into text and displayed in the UI.



Sample Project Code Overview

The main process to use GME voice chat is `Init > EnterRoom > EnableMic > EnableSpeaker`. The main code of the sample project is in `BaseViewController.cpp` and `ExperientialDemoViewController.cpp`.

Initialization

The initialization code is in the `InitGME` function in the `BaseViewController.cpp` file. It includes initialization, authentication initialization for voice message, and `TMGDelegate` callback settings.

```
int UBaseViewController::InitGME(std::string sdkAppId, std::string sdkAppKey,
std::string userId) {

    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(),
sdkAppKey.c_str(), (unsigned char *)strSig, nLength);
    ITMGContextGetInstance()->GetPTT()->ApplyPTTAauthbuffer(strSig, nLength);

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

```

Using GME requires periodic calls to the `Poll` function in `Tick` in the `UEDemoLevelScriptActor.cpp` script.

```

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

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

```

Room entry

The room entry code is in the `EnterRoom` function in the `BaseViewController.cpp` file.

```

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_str(), m_appKey.c_str(), (unsigned char *)strSig, nLength);
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow,
TEXT("onEnterRoom"));
    ITMGContextGetInstance()->EnterRoom(roomID.c_str(), roomType, strSig,
nLength);
}

```

The room entry callback is in the `OnEvent` function in the same script.

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

Device enablement

Device enablement code after successful room entry is in `ExperientialDemoViewController.cpp`.

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

3D sound effect

For the connection of 3D sound effect, see [3D Sound Effect](#). In the project, initialize the 3D sound effect feature first with the code in `ExperientialDemoViewController.cpp`.

```
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(isChecked, false);
    FString msg = FString::Printf(TEXT("InitSpatializer=%d, EnableSpatializer
ret=%d"), ret1, ret2);
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, msg);
}
```

Call the `UpdatePosition` function in `Tick` in the `UEDemoLevelScriptActor.cpp` script.

```
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-
>GetWorld(), 0)->GetCameraLocation();
    FRotator cameraRotation =
UGameplayStatics::GetPlayerCameraManager(m_pActor->GetWorld(), 0)-
```

```

>GetCameraRotation();

FString msg = FString::Printf(TEXT("location(x=%.2f,y=%.2f,z=%.2f),
rotation(pitch=%.2f,yaw=%.2f,roll=%.2f)"),
    cameraLocation.X, cameraLocation.Y, cameraLocation.Z,
    cameraRotation.Pitch, cameraRotation.Yaw, cameraRotation.Roll);

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

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

```

Enable 3D effects in `ExperientialDemoViewController.cpp` .

```

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(isChecked, false);
    FString msg = FString::Printf(TEXT("InitSpatializer=%d, EnableSpatializer
ret=%d"), ret1, ret2);
    GEngine->AddOnScreenDebugMessage(INDEX_NONE, 10.0f, FColor::Yellow, msg);
}

```