Game Multimedia Engine

Client API

Product Documentation

Contents

Client API
SDK for Unity
Integrating SDK

Last updated：2023-01-16 16:08:28

This document describes how to configure a Unity project for the GME APIs for Unity.

SDK Download

1. Download the applicable demo and SDK. For more information, see SDK Download Guide.
2. Locate the SDK resources for Unity on the page.
3. Click **Download**. After decompression, the downloaded SDK resources include the following files:

File Name	Description	Usage
Plugins	SDK library files	Stores library files for each platform
GMESDK	SDK code files	Provides APIs

4. To use HD sound quality, see Using HD Sound Quality.

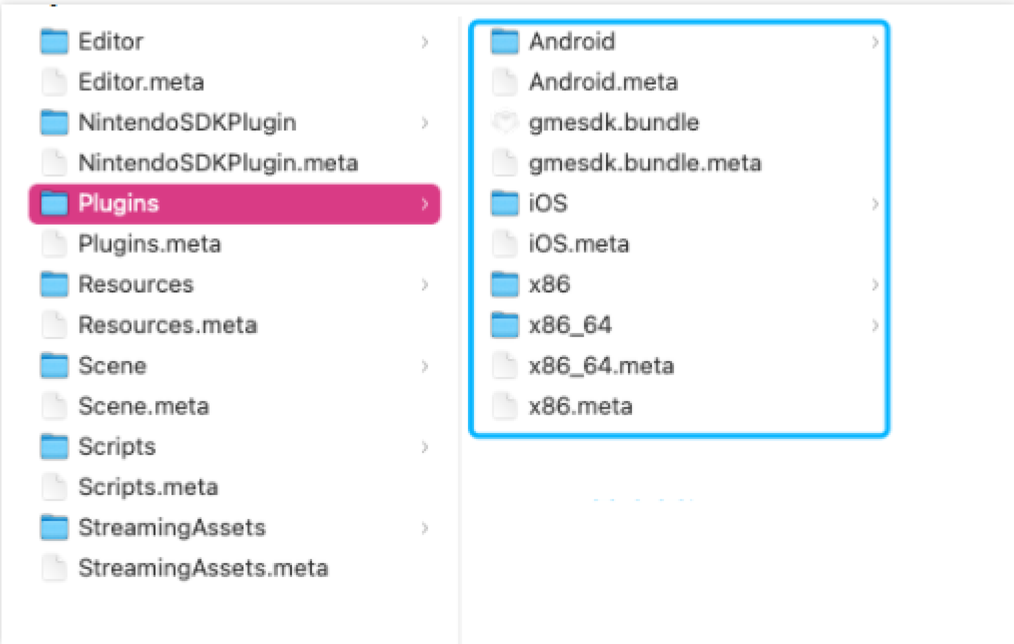
Supported Platforms

The SDK for Unity has integrated Windows, macOS, Android, iOS, PlayStation, Xbox, Switch, and WebGL platform architectures at the same time.

Project Configuration

Step 1: import Plugins files

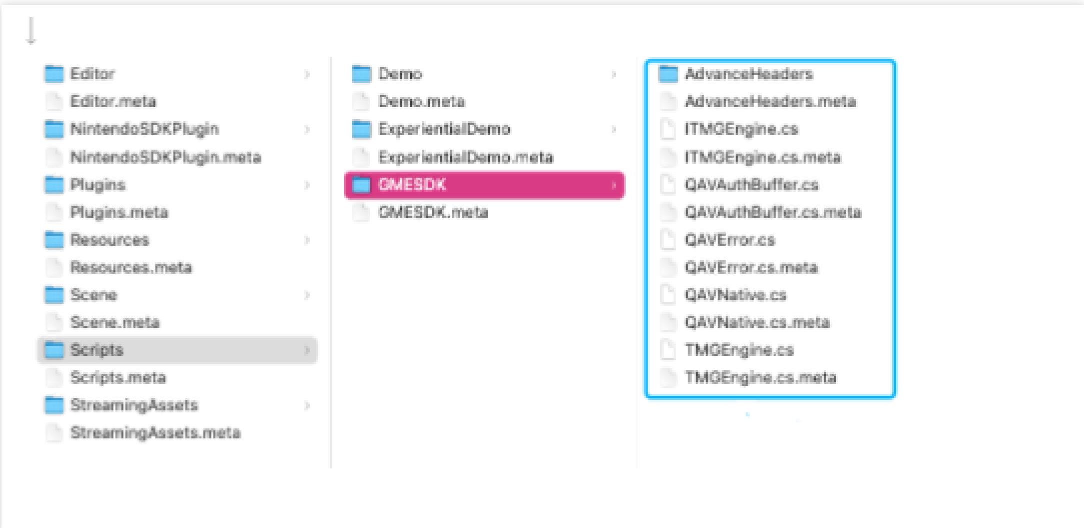
Copy the files from the `Plugins` folder in the SDK to the folder under **Unity project** > **Assets** > **Plugins** as shown below:



Note:
If you don't need to export executables in the Win32 architecture, delete the `x86` folder under the `Plugins` folder.

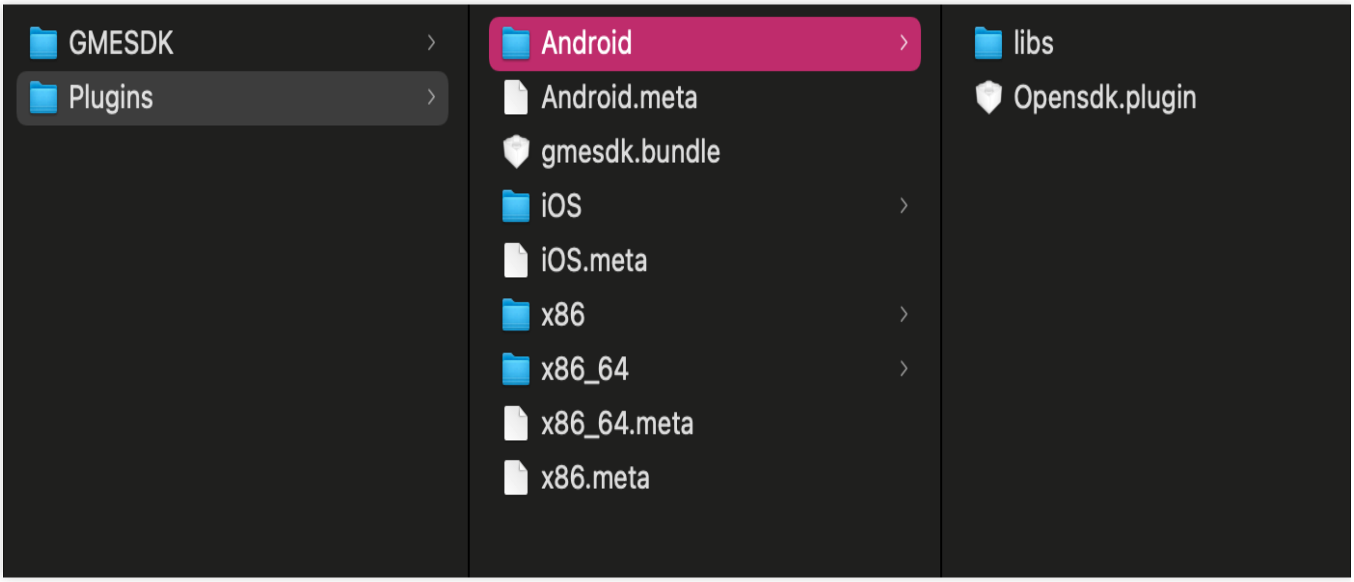
Step 2: import code files

Copy the files in the `Scripts` folder in the SDK to the folder used to store code in your Unity project as shown below:



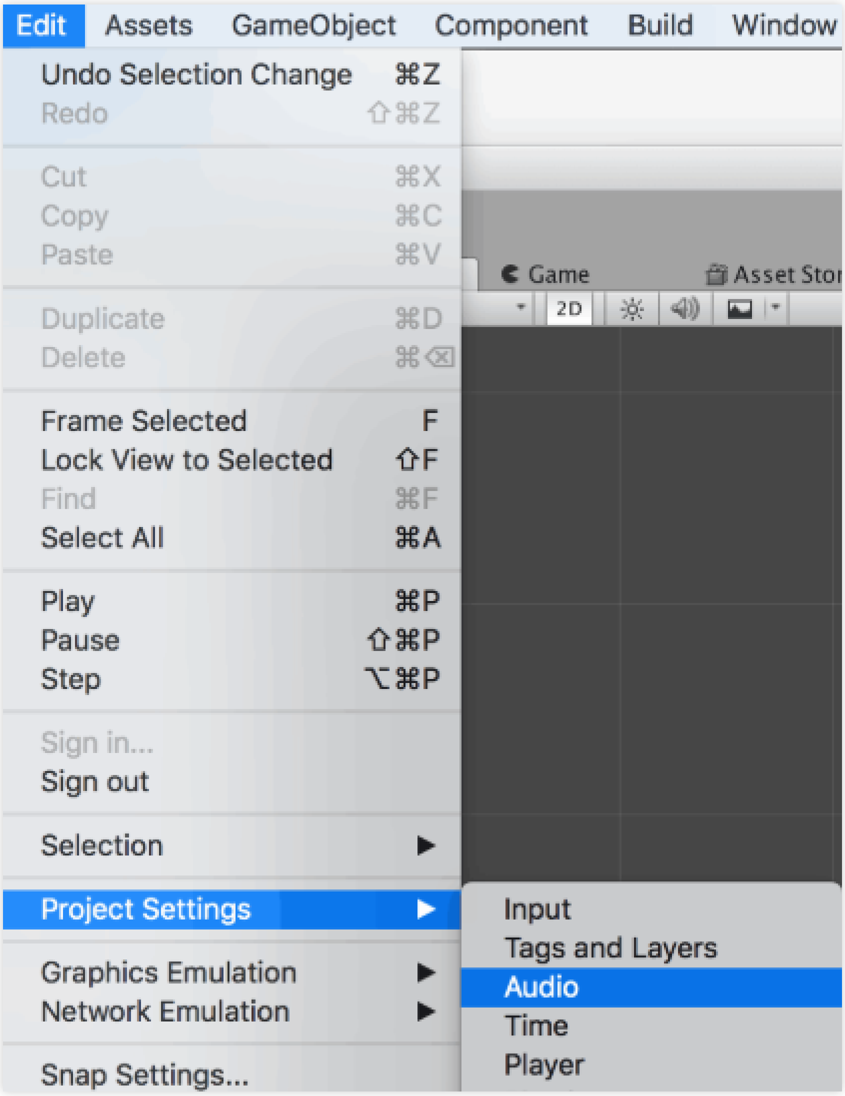
Unity 2021 Configuration

If you use Unity Editor 2021 or higher, you need to cut the `lib` folder under `Plugins` > `Android` > `Opensdk.plugin` and paste it in the `Android` directory of the `Plugins` file in the project, at the same level as `Opensdk.plugin` .



Audio Settings

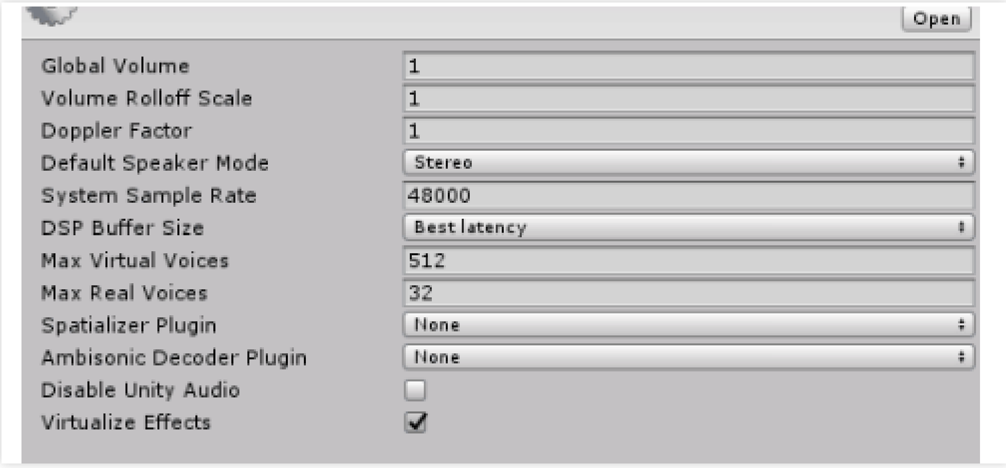
In the Unity editor, go to **Edit** > **Project Settings** > **Audio** and use the default system settings. If you make a change to the settings, Unity playback sound effect will be affected due to the hardware buffer set on the iOS device, as shown below:



Unity Audio Setting

Please do not set the **Audio** module in **Project Settings**.

If the settings are as follows, Unity playback sound effect will be interrupted due to the hardware buffer set on the iOS device:



Operations on macOS

If you use Unity to integrate the GME SDK on macOS 10.15.x, an error shows that the file is corrupted during the execution due to the `com.apple.quarantine` attribute.

The most direct solution is to delete the `com.apple.quarantine` attribute, as shown below:

1. Run the `cd` command in terminal to go to the `Unity_OpenSDK_Audio/Assets/Plugins/` folder in the project.
2. Run the following command.

```
$ xattr -d com.apple.quarantine gmesdk.bundle
```

Note:

This operation is risky. We recommend you use an earlier version of macOS for access.

Voice Chat

Last updated：2023-04-27 17:06:33

This document describes how to integrate with and debug GME client APIs for the voice chat feature for Unity.

Key Considerations for Using GME

GME provides the real-time voice chat service, voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

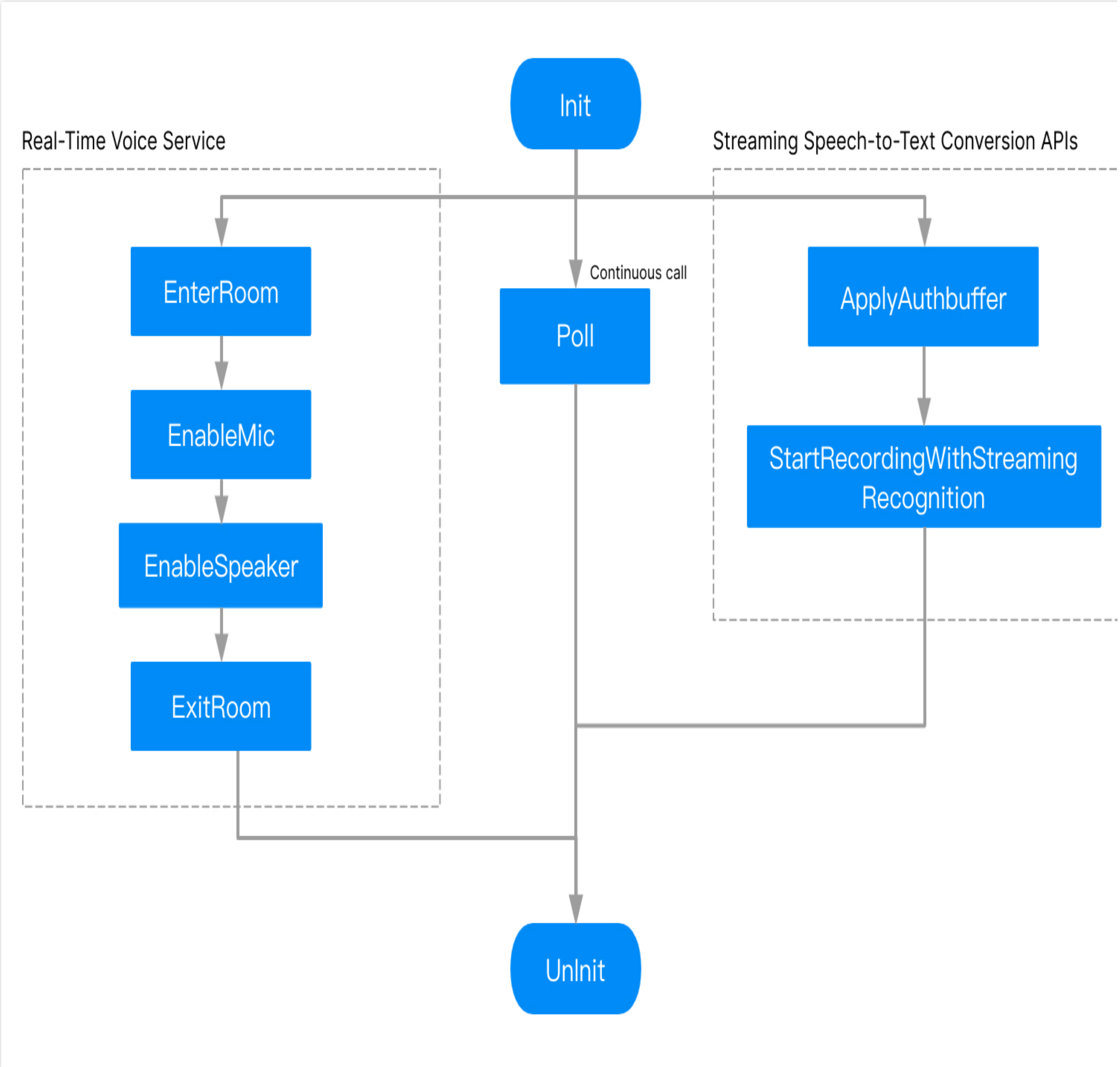
Notes

You have created a GME application and obtained the SDK `AppID` and key. For more information, see Activating Services.
You have activated **GME voice chat, voice messaging, and speech-to-text services**. For more information, see Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `QAVError.OK` will be returned with the value being `0` .
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error codes, see Error Codes.
GME only supports simple voice chat features on Unity WebGL. For more information, see Project Configuration.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initialize GME
2. Call `Poll` periodically to trigger callbacks
3. Enter a voice chat room
4. Turn on the mic
5. Turn on the speaker
6. Exit the voice chat room
7. Uninitialize GME

C# classes

Class	Description
ITMGContext	Core APIs

ITMGRoom	Room APIs
ITMGRoomManager	Room management APIs
ITMGAudioCtrl	Audio APIs
ITMGAudioEffectCtrl	Sound effect and accompaniment APIs

Core APIs

API	Description
Init	Initializes GME.
Poll	Triggers an event callback.
Pause	Pauses the system.
Resume	Resumes the system.
Uninit	Uninitializes GME.

Importing header files

```
using GME;
```

Getting an instance

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

Initializing the SDK

You need to initialize the SDK through the `Init` API before you can use the real-time voice chat, 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

sdkAppId	string	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	string	`openID` can only be in Int64 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.

Returned values

Return Value	Handling
QAVError.OK= 0	The SDK was initialized successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. We recommend that you delete it and then import it again.

Notes on 7015 error code

The error code 7015 is identified based on the MD5 value. If this error is reported during integration, check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party enhancement, the Unity packaging mechanism, and other factors, the library file MD5 will be affected, resulting in misjudgment. Therefore, **ignore this error in the logic for official releases**, and avoid displaying it on the UI.

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;
    }
```

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.

Call the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
ITMGContext public abstract int Poll();
```

Sample code

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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. For example, when the application switches to the background (OnApplicationPause, isPause=True), and you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext public abstract int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext  public abstract int Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK. **If the game account is bound to** `openid` , switching game account requires uninitializing GME and then using the new `openid` to initialize again.

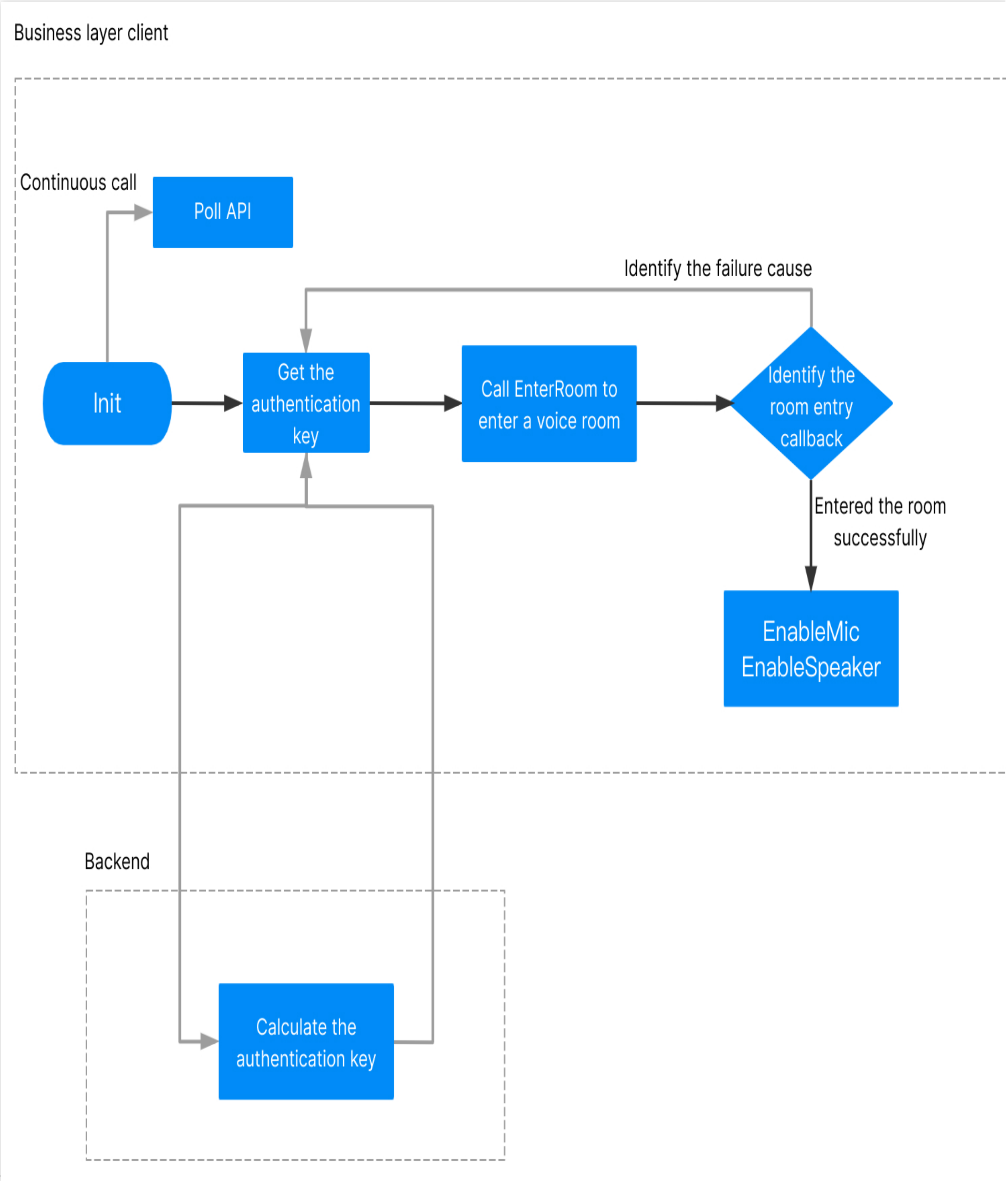
API prototype

```
ITMGContext public abstract int Uninit()
```

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, see Sound and Audio.



API	Description
GenAuthBuffer	Calculates the local authentication key.

EnterRoom	Enter a room
ExitRoom	Leave room
IsRoomEntered	Determines whether room entry is successful.
SwitchRoom	Switch Room

Local authentication key calculation

Generate `AuthBuffer` for encryption and authentication of relevant features. For release in the production environment, 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	`AppID` from the Tencent Cloud console
roomId	string	Room ID, which can contain up to 127 characters.
openId	string	User ID, which is the same as `openID` 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);
}
```

WebGL adaptation

On WebGL, after the local authentication function is called, the authentication key is saved in the JS code, and `authBuffer` is not returned to the C# layer. After a user calls the `GetAuthBuffer` API for local authentication, the user can enter `""` or a random value in the called room entry API.
If the authentication key is calculated on the backend, `GetAuthBuffer` doesn't need to be called.

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.

Notes

If the room entry callback result is `0`, the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.

The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will become smooth sound quality. Only when a member in the room calls the `ChangeRoomType` API, the audio type of the room will be changed.

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	Room type. We recommend that you select `ITMG_ROOM_TYPE_FLUENCY` for games. For more information on room audio types, see Sound Quality.
authBuffer	Byte[]	Authentication key

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

Will Voice Chat still be charged when client gets offline?

API prototype

```
public delegate void QAVEnterRoomComplete(int result, string error_info);
```



```
public abstract event QAVEnterRoomComplete OnEnterRoomCompleteEvent;
```

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
    }
}
```

Data details

Messages	Data	Example
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnection callback notification `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code	Cause and Suggested Solution
7006	Authentication failed. Causes: `AppID` doesn't exist or is incorrect. An error occurred while authenticating `authbuff` . Authentication expired.

	`OpenId` is invalid.
7007	The user was already in another room.
1001	The user was already in the process of entering a room but repeated this operation. We recommend that you not call the room entry API until the room entry callback is returned.
1003	The user was already in the room and called the room entry 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.

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API; instead, you can directly call the `EnterRoom` API.

API prototype

```
ITMGContext ExitRoom()
```

Sample code

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

Callback for room exit

A callback will be executed through a delegate function to pass a message after room exit.

API prototype

```
public delegate void QAVExitRoomComplete();
public abstract event QAVExitRoomComplete OnExitRoomCompleteEvent;
```

Sample code

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

```
}
```

Determining whether user has entered room

This API is used to determine whether the user has entered a room. A value in bool type will be returned. Do not call this API during room entry.

API prototype

```
ITMGContext abstract bool IsRoomEntered()
```

Sample code

```
ITMGContext.GetInstance().IsRoomEntered();
```

Switching room

User can call this API to quickly switch the voice chat room after entering the room. After the room is switched, the device is not reset, that is, if the microphone is already enabled in this room, the microphone will keep enabled after the room is switched.
The callback for quickly switching rooms is `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM` , and the fields are `error_info` and `result` .

API prototype

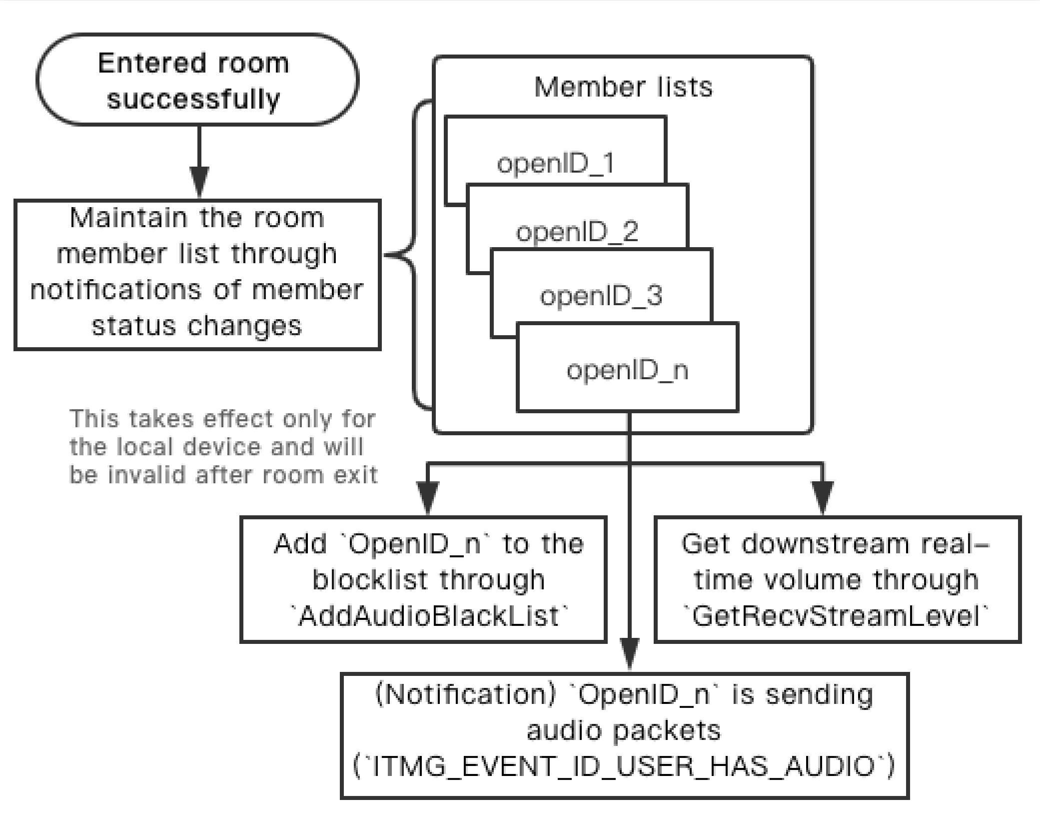
```
public abstract int SwitchRoom(string targetRoomID, byte[] authBuffer);
```

Type description

Parameter	Type	Description
targetRoomID	String	ID of the room to enter
authBuffer	byte[]	Generates a new authentication key with the ID of the room to enter

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed.
AddAudioBlackList	Mutes a member in the room.
RemoveAudioBlackList	Unmutes a member.

Notification events of the member room entry and speaking status

This event is used to get the user speaking in the room and display the user on the UI, and to send a notification when someone enters or exits the room.
A notification for this event will be sent only when the status changes. To get the member status in real time, cache the notification when it is received at the business layer. The event message `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` containing `event_id` , `count` , and `openIdList` will be returned, which will be identified in the `OnEvent` notification.
Notifications of the `EVENT_ID_ENDPOINT_NO_AUDIO` audio event will be sent only when the threshold is exceeded; that is, other members in the room can receive the notification that the local user stops speaking only after the local client captures no voice for two seconds.
The audio event returns only the member speaking status but not the specific volume level. If you need the specific volume levels of members in the room, you can use the `GetRecvStreamLevel` API.

event_id	Description	Maintenance

EVENT_ID_ENDPOINT_ENTER	Return the `openid` of the member entering the room.	Member list
EVENT_ID_ENDPOINT_EXIT	Return the `openid` of the member exiting the room.	Member list
EVENT_ID_ENDPOINT_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
EVENT_ID_ENDPOINT_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
public delegate void QAVEndpointsUpdateInfo(int eventID, int count,
[MarshalAs(UnmanagedType.LPArray, SizeParamIndex = 1)]string[] openIdList);
public abstract event QAVEndpointsUpdateInfo OnEndpointsUpdateInfoEvent;

// Listen on an event:
ITMGContext.GetInstance().OnEndpointsUpdateInfoEvent += new
QAVEndpointsUpdateInfo(OnEndpointsUpdateInfo);
// Process the event listened on:
void OnEndpointsUpdateInfo(int eventID, int count, string[] openIdList)
{
                // Process
            switch (eventID)
             {
             case EVENT_ID_ENDPOINT_ENTER:
                  // A member enters the room
                  break;
             case EVENT_ID_ENDPOINT_EXIT:
                  // A member exits the room
                break;
            case EVENT_ID_ENDPOINT_HAS_AUDIO:
                // A member sends audio packets
                break;
            case EVENT_ID_ENDPOINT_NO_AUDIO:
                // A member stops sending audio packets
                break;

            default:
                break;
             }
        break;
```

```
}
```

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B.

If B blocks neither A nor C, B can hear both of them.

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
ITMGContext ITMGAudioCtrl AddAudioBlackList(String openId)
```

Parameter	Type	Description
openId	String	`openid` of the user to be blocked

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl ().AddAudioBlackList (openId);
```

Unmuting

This API is used to remove an ID from the audio data blacklist. A returned value of 0 indicates the call is successful.

API prototype

```
ITMGContext ITMGAudioCtrl RemoveAudioBlackList(string openId)
```

Parameter	Type	Description
openId	String	User `openid` to be unblocked

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl ().RemoveAudioBlackList (openId);
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.
When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend that you call the `EnableMic` or `EnableSpeaker` API.
To enable the user to press the mic button on the UI to speak and release it to stop speaking, we recommend that you call `EnableAudioCaptureDevice` once during room entry and call `EnableAudioSend` to enable the user to speak while pressing the button.

API	Description
EnableMic	Turns on/off the mic.
GetMicState	The mic status was obtained.
EnableAudioCaptureDevice	Enables/Disables the capturing device.
IsAudioCaptureDeviceEnabled	Gets the capturing device status.
EnableAudioSend	Enables/Disables audio upstreaming.
IsAudioSendEnabled	Gets the audio upstreaming status.
GetMicLevel	Gets the real-time mic volume level.
GetSendStreamLevel	Gets the real-time audio upstreaming volume level.
SetMicVolume	Sets the mic volume level.
GetMicVolume	Gets the mic volume level.

Enabling or disabling mic

This API is used to enable/disable the mic. The mic and speaker are not enabled by default after room entry.
EnableMic = EnableAudioCaptureDevice + EnableAudioSend

API prototype

```
ITMGAudioCtrl EnableMic(bool isEnabled)
```

Parameter	Type	Description
isEnabled	boolean	To enable the mic, set this parameter to `true` ; otherwise, set it to `false` .

Sample code


```
// Turn on mic
ITMGContext.GetInstance().GetAudioCtrl().EnableMic(true);
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
ITMGAudioCtrl GetMicState()
```

Sample code

```
micToggle.isOn = ITMGContext.GetInstance().GetAudioCtrl().GetMicState();
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.

This API can only be called after room entry. The device will be disabled automatically after room exit.

Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
ITMGAudioCtrl int EnableAudioCaptureDevice(bool isEnabled)
```

Parameter	Type	Description
isEnabled	bool	To enable the capturing device, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable capturing device
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioCaptureDevice(true);
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
ITMGAudioCtrl bool IsAudioCaptureDeviceEnabled()
```

Sample code

```
bool IsAudioCaptureDevice =
ITMGContext.GetInstance().GetAudioCtrl().IsAudioCaptureDeviceEnabled();
```

Enabling or disabling audio stream sending

This API is used to enable/disable audio stream sending. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain mute. For more information on how to enable/disable the capturing device, see the `EnableAudioCaptureDevice` API.

API prototype

```
ITMGAudioCtrl int EnableAudioSend(bool isEnabled)
```

Parameter	Type	Description
isEnabled	bool	To enable audio upstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioSend(true);
```

Getting the audio stream sending status

This API is used to get the status of audio stream sending.

API prototype

```
ITMGAudioCtrl bool IsAudioSendEnabled()
```

Sample code

```
bool IsAudioSend =
ITMGContext.GetInstance().GetAudioCtrl().IsAudioSendEnabled();
```

Getting the real-time mic volume

This API is used to get the real-time mic volume. An int-type value in the range of 0-100 will be returned. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl int GetMicLevel
```

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().GetMicLevel();
```

Getting the real-time audio stream sending volume

This API is used to get the local real-time audio stream sending volume. An int-type value in the range of 0–100 will be returned.

API prototype

```
ITMGAudioCtrl int GetSendStreamLevel()
```

Sample code

```
int Level = ITMGContext.GetInstance().GetAudioCtrl().GetSendStreamLevel();
```

Setting the mic volume

This API is used to set the mic volume level. The corresponding parameter is `volume`, which is equivalent to attenuating or gaining the captured sound.

API prototype

```
ITMGAudioCtrl SetMicVolume(int volume)
```

Parameter	Type	Description
volume	int	Value range: 0-200. Default value: `100`. `0` indicates that the audio is muted, while `100` indicates that the volume level remains unchanged.

Sample code

```
int micVol = (int)(value * 100);
ITMGContext.GetInstance().GetAudioCtrl().SetMicVolume (micVol);
```

Getting the mic volume

This API is used to obtain the microphone volume. An "int" value is returned. Value 101 represents API SetMicVolume has not been called.

API prototype

```
ITMGAudioCtrl GetMicVolume()
```

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().GetMicVolume();
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker.
GetSpeakerState	The speaker status was obtained.
EnableAudioPlayDevice	Enables/Disables the playback device.
IsAudioPlayDeviceEnabled	Gets the playback device status.
EnableAudioRecv	Enables/Disables audio downstreaming.
IsAudioRecvEnabled	Gets the audio downstreaming status.
GetSpeakerLevel	Gets the real-time speaker volume level.
GetRecvStreamLevel	Gets the real-time downstreaming audio levels of other members in the room.
SetSpeakerVolume	Sets the speaker volume level.
GetSpeakerVolume	Gets the speaker volume level.

Enabling or disabling speaker

This API is used to enable/disable the speaker. **EnableSpeaker = EnableAudioPlayDevice + EnableAudioRecv**

API prototype

```
ITMGAudioCtrl EnableSpeaker(bool isEnabled)
```

Parameter	Type	Description
isEnabled	bool	To disable the speaker, set this parameter to `false`; otherwise, set it to `true`.

Sample code

```
// Turn on the speaker
ITMGContext.GetInstance().GetAudioCtrl().EnableSpeaker(true);
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

API prototype

```
ITMGAudioCtrl GetSpeakerState()
```

Sample code

```
speakerToggle.isOn =
ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerState();
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

API prototype

```
ITMGAudioCtrl EnableAudioPlayDevice(bool isEnabled)
```

Parameter	Type	Description
isEnabled	bool	To disable the playback device, set this parameter to `false`; otherwise, set it to `true`.

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioPlayDevice(true);
```

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
ITMGAudioCtrl bool IsAudioPlayDeviceEnabled()
```

Sample code

```
bool IsAudioPlayDevice =
ITMGContext.GetInstance().GetAudioCtrl().IsAudioPlayDeviceEnabled();
```

Enabling or disabling audio stream receiving

This API is used to enable/disable audio stream receiving. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain muted. For more information on how to enable/disable the playback device, see the `EnableAudioPlayDevice` API.

API prototype

```
ITMGAudioCtrl int EnableAudioRecv(bool isEnabled)
```

Parameter	Type	Description
isEnabled	bool	To enable audio downstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioRecv(true);
```

Getting the audio stream receiving status

This API is used to get the status of audio stream receiving.

API prototype

```
ITMGAudioCtrl bool IsAudioRecvEnabled()
```

Sample code

```
bool IsAudioRecv =
ITMGContext.GetInstance().GetAudioCtrl().IsAudioRecvEnabled();
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume. An int-type value will be returned to indicate the volume. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl GetSpeakerLevel()
```

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerLevel();
```

Getting the audio stream volume levels of other members in the room

This API is used to get the real-time audio stream volume levels of other members in the room. An int-type value will be returned. Value range: 0–200.

API prototype

```
ITMGAudioCtrl int GetRecvStreamLevel(string openId)
```

Parameter	Type	Description
openId	string	`openId` of another member in the room

Sample code

```
int Level =
ITMGContext.GetInstance().GetAudioCtrl().GetRecvStreamLevel(openId);
```

Dynamically setting the volume of a member of the room

This API is used to set the volume of a member in the room. It takes effect only on the local side.

API prototype

```
public abstract int SetSpeakerVolumeByOpenID(string openid, int volume);
```

Parameter	Type	Description
openId	String	`OpenID` of the target user
volume	int	Percentage. Recommended value range: 0-200. Default value: `100` .

Setting the speaker volume

This API is used to set the speaker volume.

API prototype

```
ITMGAudioCtrl SetSpeakerVolume(int volume)
```

Parameter	Type	Description
volume	int	Volume level. Value range: 0-200. Default value: `100` . `0` indicates that the audio is muted, while `100` indicates that the volume level remains unchanged.

Sample code

```
int speVol = (int)(value * 100);
ITMGContext.GetInstance().GetAudioCtrl().SetSpeakerVolume(speVol);
```

Getting the speaker volume

This API is used to get the speaker volume. An int-type value will be returned to indicate the volume. 101 indicates that the `SetSpeakerVolume` API has not been called.

"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

API prototype

```
ITMGAudioCtrl GetSpeakerVolume()
```

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerVolume();
```

Device Selection APIs

Device selection APIs can be used only on PC.

API	Description
GetMicListCount	Gets the number of mics.
GetMicList	Enumerates mics.
GetSpeakerListCount	Gets the number of speakers.
GetSpeakerList	Enumerates speakers.
SelectMic	Selects a mic.
SelectSpeaker	Selects a speaker.

Getting the number of mics

This API is used to get the number of mics.

Function prototype

```
public abstract int GetMicListCount()
```

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().GetMicListCount();
```

Enumerating mics

This API is used together with the `GetMicListCount` API to enumerate mics.

Function prototype

```
public abstract int GetMicList(out List<TMGAudioDeviceInfo> devicesInfo, int
count)
```

Parameter	Type	Description

ppDeviceInfoList	TMGAudioDeviceInfo	Device list
count	int	Number of mics

Parameter of `TMGAudioDeviceInfo`	Type	Description
m_strDeviceID	string	Device name
m_strDeviceID	string	Device ID

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().GetMicList(devicesInfo,count);
```

Selecting mic

This API is used to select a mic. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default mic will be selected.
The 0th device id returned in the GetMicList API is the default device of the call device. If there is a selected call device, it will be maintained by service. If it is unplugged, the call device will be changed back into the default device.

Function prototype

```
public abstract int SelectMic(string micID);
```

Parameter	Type	Description
pMicID	string	Mic ID, which is from the list returned by `GetMicList` .

Sample code

```
string deviceID = DEVICE_ID_DEFAULT;
                if (index != 0)
                {
                    deviceID = listMicInfo[index - 1].m_strDeviceID;
                }
                ITMGContext.GetInstance().GetAudioCtrl().SelectMic(deviceID);
                selectedMicID = deviceID;
```

This API is used to get the number of speakers.

Function prototype

```
public abstract int GetSpeakerListCount();
```

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerListCount();
```

Enumerating speakers

This API is used together with the `GetSpeakerListCount` API to enumerate speakers.

Function prototype

```
public abstract int GetSpeakerList(out List<TMGAudioDeviceInfo> devicesInfo,
int count)
```

Parameter	Type	Description
ppDeviceInfoList	TMGAudioDeviceInfo	Device list
count	int	Number of speakers

Parameter of `TMGAudioDeviceInfo`	Type	Description
m_strDeviceID	string	Device name
m_strDeviceID	string	Device ID

Sample code

```
int speakerCount =
ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerListCount();
Debug.LogFormat("speakerCount = {0}", speakerCount);
if (speakerCount > 0)
    {
        int ret = ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerList(out
listSpeakerInfo, speakerCount);
        Debug.LogFormat("GetSpeakerList ret = {0}", ret);
        if (ret != 0)
        {
            listSpeakerInfo = null;
        }
```

```
    }
}
```

Selecting speaker

This API is used to select a playback device. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default playback device will be selected.

Function prototype

```
public abstract int SelectSpeaker(string speaker);
```

Parameter	Type	Description
speaker	string	Speaker ID, which is from the list returned by `GetSpeakerList` .

Sample code

```
speakerDropdown =
transform.Find("DevicePanel/SpeakerSelect").GetComponent<Dropdown>();
        if (speakerDropdown != null)
        {
            speakerDropdown.onValueChanged.AddListener(delegate (int index)
            {
                string deviceID = DEVICE_ID_DEFAULT;
                if (index != 0)
                {
                    deviceID = listSpeakerInfo[index - 1].m_strDeviceID;
                }

ITMGContext.GetInstance().GetAudioCtrl().SelectSpeaker(deviceID);
                selectedSpeakerID = deviceID;
            });
        }
```

Special APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

API prototype

```
ITMGContext GetAudioCtrl EnableLoopBack(bool enable)
```

Parameter	Type	Description
enable	bool	Specifies whether to enable the in-ear monitoring.

Sample code

```
ITMGContext.GetInstance().GetAudioCtrl().EnableLoopBack(true);
```

Callback for device use and release

After a device is used or released in a room, a callback will be executed through a delegate function to pass a message of the event.

```
public delegate void QAVOnDeviceStateChangedEvent(int deviceType, string
deviceId, bool openOrClose);
public abstract event QAVOnDeviceStateChangedEvent OnDeviceStateChangedEvent;
```

Parameter	Type	Description
deviceType	int	`1` indicates capturing device. `2` indicates playback device.
deviceId	string	Device GUID, which identifies a device and only applies to Windows and macOS.
openOrClose	bool	Whether the capturing/playback device is occupied or released

Sample code

```
Listen on an event:
ITMGContext.GetInstance().GetAudioCtrl().OnDeviceStateChangedEvent += new
QAVAudioDeviceStateCallback(OnAudioDeviceStateChange);
Process the event listened on:
void QAVAudioDeviceStateCallback(int deviceType, string deviceId, bool
openOrClose){
    // Callback for device occupancy and release
}
```

Getting user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, see the `EnterRoom` API.

API prototype

```
ITMGContext ITMGRoom public int GetRoomType()
```

Sample code

```
ITMGContext.GetInstance().GetRoom().GetRoomType();
```

Changing the room type

This API is used to modify a user's room audio type. For the result, see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` . The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

API prototype

```
ITMGContext ITMGRoom public int ChangeRoomType(ITMGRoomType roomtype)
```

Parameter	Type	Description
roomtype	ITMGRoomType	Room type to be switched to. For room audio types, see the `EnterRoom` API.

Sample code

```
ITMGContext.GetInstance().GetRoom().ChangeRoomType(ITMG_ROOM_TYPE_FLUENCY);
```

Callback event

Set the room type. After the room type is set, a callback will be executed through a delegate function to pass a message indicating that the modification has been completed.

Returned Parameter	Description
roomtype	Updated room type

```
public abstract event QAVCallback OnChangeRoomtypeCallback;
public abstract event QAVOnRoomTypeChangedEvent OnRoomTypeChangedEvent;
```

Sample code

```
// Listen on an event:
ITMGContext.GetInstance ().OnRoomTypeChangedEvent += new
QAVOnRoomTypeChangedEvent (OnRoomTypeChangedEvent);
// Process the event listened on:
void OnRoomTypeChangedEvent(int roomtype)
{
        ShowWarnning (string.Format ("RoomTypeChanged current:{0}",roomtype));
}
```

Notification of room type change

Once the room type is changed by you or another user in the room, this notification event will be used to notify the business layer of the room type change. The returned value will be the room type. For more information, see the `EnterRoom` API.

```
public delegate void QAVOnRoomTypeChangedEvent(int roomtype);
public abstract event QAVOnRoomTypeChangedEvent OnRoomTypeChangedEvent;
```

Sample code

```
// Listen on an event:
ITMGContext.GetInstance().OnRoomTypeChangedEvent += new
QAVOnRoomTypeChangedEvent(OnRoomTypeChangedEvent);
// Process the event listened on:
void OnRoomTypeChangedEvent(int roomtype){
    // Send a callback after the room type is changed
}
```

The monitoring event of room call quality

This is the quality monitoring event used to listen on the network quality. If your network conditions are poor, the business layer will ask you to switch the network through the UI. This event is triggered once every two seconds after room entry, and its message is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which are as detailed below:

Parameter	Type	Description
weight	int	Value range: 1–50. `50` indicates excellent sound quality, `1` indicates very poor (barely usable) sound quality, and `0` represents an initial meaningless value. Generally, if the value is below 30, you can remind users that the network is poor and recommend them to switch the network.
Loss	double	Upstream packet loss rate

Delay	int	Voice chat delay in ms

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
ITMGContext  abstract string GetSDKVersion()
```

Sample code

```
ITMGContext.GetInstance().GetSDKVersion();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext  SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: `TMG_LOG_LEVEL_INFO` .
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: `TMG_LOG_LEVEL_ERROR` .

`ITMG_LOG_LEVEL` description:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs

TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContext.GetInstance().SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before initialization.

Platform	Path
Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents
Android	/sdcard/Android/data/xxx.xxx.xxx/files
macOS	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext  SetLogPath(string logDir)
```

Parameter	Type	Description
logDir	String	Path

Sample code

```
ITMGContext.GetInstance().SetLogPath(path);
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

API prototype

```
ITMGRoom GetQualityTips()
```

Sample code

```
string tips = ITMGContext.GetInstance().GetRoom().GetQualityTips();
```

Speech-to-Text Service

Last updated：2023-05-19 15:46:00

This document describes how to integrate with and debug GME client APIs for the voice message and speech-to-text services for Unity.

Key Considerations for Using GME

GME provides the real-time voice chat service, voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll`.

Notes

You have created a GME application and obtained the SDK `AppID` and key. For more information, see Activating Services.
You have activated **GME voice chat, voice messaging, and speech-to-text services**. For more information, see Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `QAVError.OK` will be returned with the value being `0`.
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error codes, see Error Codes.

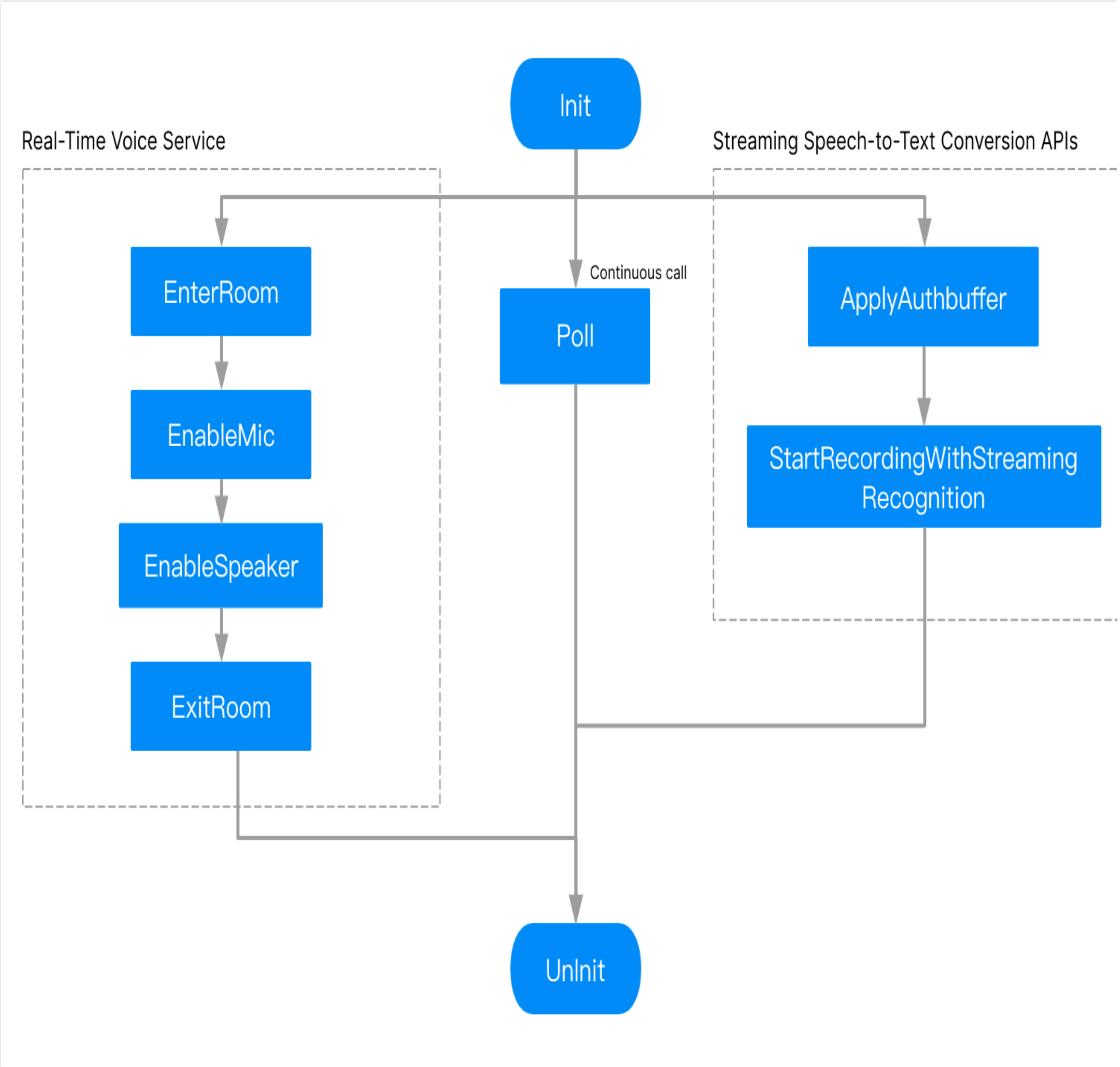
Notes

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.
Non-streaming speech-to-text API **SpeechToText()**: There can be up to 10 concurrent requests per account.
Streaming speech-to-text API **StartRecordingWithStreamingRecognition()**: There can be up to 50 concurrent requests per account.
Voice chat streaming speech-to-text API **StartRealTimeASR()**: There can be up to 50 concurrent requests per account.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initialize GME
2. Call `Poll` periodically to trigger callbacks
3. Initialize authentication
4. Starting streaming speech-to-text conversion
5. Stop recording
6. Uninitialize GME

C# classes

Class	Description
ITMGContext	Core APIs

ITMGPTT	Voice messaging and speech-to-text APIs

Core APIs

API	Description
Init	Initializes GME.
Poll	Triggers an event callback.
Pause	Pauses the system.
Resume	Resumes the system.
Uninit	Uninitializes GME.

Importing header files

```
using GME;
```

Getting an instance

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

Initializing the SDK

You need to initialize the SDK through the `Init` API before you can use the real-time voice chat, 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
sdkAppId	string	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	string	`openID` can only be in Int64 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.

Returned values

Return Value	Handling
QAVError.OK= 0	The SDK was initialized successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. We recommend that you delete it and then import it again.

Notes on 7015 error code

The error code 7015 is identified based on the MD5 value. If this error is reported during integration, check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party enhancement, the Unity packaging mechanism, and other factors, the library file MD5 will be affected, resulting in misjudgment. Therefore, **ignore this error in the logic for official releases**, and avoid displaying it on the UI.

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;
    }
```

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.

Notes

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
ITMGContext public abstract int Poll();
```

Sample code


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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. For example, when the application switches to the background (OnApplicationPause, isPause=True), and you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext public abstract int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext  public abstract int Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK. **If the game account is bound to** `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.

API prototype

```
ITMGContext public abstract int Uninit()
```

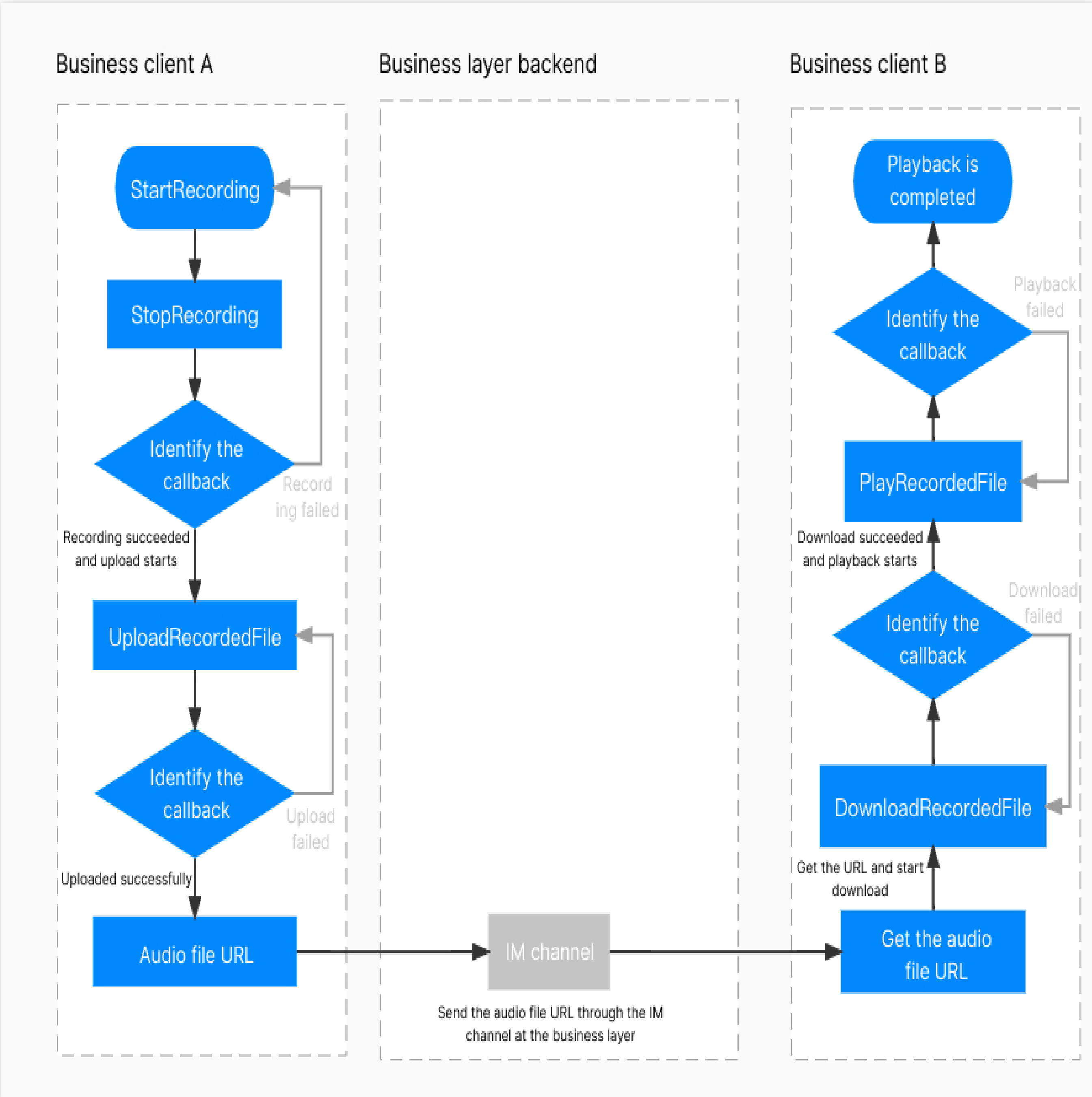
Voice Messaging and Speech-to-Text Services

Note

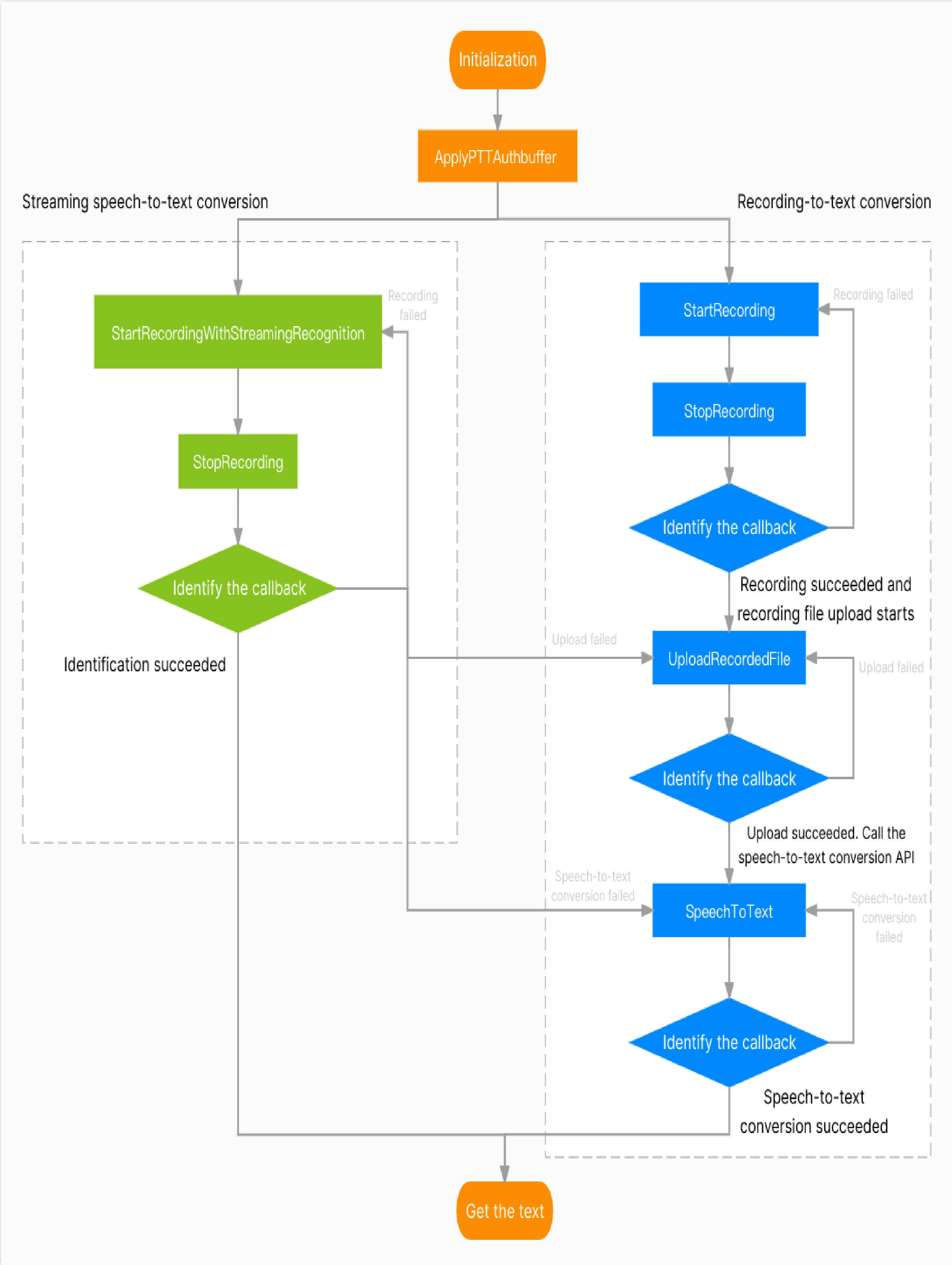
The speech-to-text service consists of fast recording-to-text conversion and streaming speech-to-text conversion.
You do not need to enter a voice chat room when using the voice messaging service.
The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum

recording duration to 10 seconds, please call the `SetMaxMessageLength` API to set it after initialization.

Flowchart for using the voice message service



Flowchart for using the speech-to-text service



API	Description

GenAuthBuffer	Generates the local authentication key.
ApplyPTTAuthbuffer	Initializes authentication.
SetMaxMessageLength	Specifies the maximum duration of a voice message.

Generating the local authentication key

Generate `AuthBuffer` for encryption and authentication of relevant features. For release in the production environment, 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	`AppId` from the Tencent Cloud console
roomId	string	Enter `null` or an empty string.
openId	string	User ID, which is the same as `OpenId` during initialization.
key	string	Permission key from the Tencent Cloud console.

Application authentication

After the authentication information is generated, the authentication is assigned to the SDK.

API prototype

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

Parameter	Type	Description
authBuffer	byte[]	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);
```

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

API prototype

```
ITMGPTT int SetMaxMessageLength(int msTime)
```

Parameter	Type	Description
msTime	int	Audio duration in ms. Value range: 1000 < `msTime` < = 58000.

Sample code

```
ITMGContext.GetInstance().GetPttCtrl().SetMaxMessageLength(58000);
```

Streaming Speech Recognition

Voice messaging and speech-to-text APIs

API	Description
StartRecordingWithStreamingRecognition	Starts streaming recording.
StopRecording	This API is used to stop audio recording.

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 text recognized in speech into a specified language and return the translation. **To stop recording, call StopRecording.**

API prototype

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

```
ITMGPTT int StartRecordingWithStreamingRecognition(string filePath, string
speechLanguage,string translateLanguage)
```

Parameter	Type	Description
filePath	String	Path of the stored audio file
speechLanguage	String	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translateLanguage	String	Enter the value of `speechLanguage` .

Sample code

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

Notes

Translation incurs additional fees. For more information, see Purchase Guide.

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` notification based on the actual needs. The passed parameters include the following four messages.

Parameter	Description
result	Return code indicating whether streaming speech-to-text conversion is successful
text	Text converted from speech
file_path	Local path of the stored recording file
file_id	Backend URL address of recording file, which will be retained for 90 days

Notes

The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

Error Code	Description	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 conversion 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.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

If the error code 4098 is reported, see Speech-to-text Conversion for solutions.

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");
                    }
}
```

Voice Message Recording

The recording process is as follows: start recording > stop recording > return recording callback > start the next recording.

Voice messaging and speech-to-text APIs

API	Description
StartRecording	Starts recording.
PauseRecording	Pauses recording.
ResumeRecording	Resumes recording.
StopRecording	This API is used to stop audio recording.
CancelRecording	Cancels recording.

Starting recording

This API is used to start recording.

API prototype

```
ITMGPTT int StartRecording(string fileDir)
```

Parameter	Type	Description
fileDir	string	Path of the stored audio file

Sample code

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

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();
```

Callback for recording start

A callback will be executed through a delegate function to pass a message when recording is completed.
**To stop recording, call `StopRecording` **. The callback for recording start will be returned after the recording is stopped.

API prototype

```
public delegate void QAVRecordFileCompleteCallback(int code, string filepath);
public abstract event QAVRecordFileCompleteCallback OnRecordFileComplete;
```

Parameter	Type	Description
code	string	`0` : Recording is completed.
filepath	string	Path of the stored recording file, which must be accessible and cannot be the `fileid` .

Error codes

Error Code	Caused By	Suggestion
4097	Empty parameters.	Check whether the API parameters in the code are correct.
4098	An initialization error occurred.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Make sure that the SDK recording feature is used at the right time.
4100	No audio data is captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	The mic permission is required for using the SDK. To add the permission, see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

```
// Listen on an event
ITMGContext.GetInstance().GetPttCtrl().OnRecordFileComplete +=  new
QAVRecordFileCompleteCallback (OnRecordFileComplete);
// Process the event listened on
void OnRecordFileComplete(int code, string filepath){
    // Callback for recording start
}
```

Pausing recording

This API is used to pause recording. If you want to resume recording, call the `ResumeRecording` API.

API prototype

```
ITMGPTT int PauseRecording()
```

Sample code

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

Resuming recording

This API is used to resume recording.

API prototype

```
ITMGPTT int ResumeRecording()
```

Sample code

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

Canceling recording

This API is used to cancel recording. **There is no callback after cancellation**.

API prototype

```
ITMGPTT int CancelRecording()
```

Sample code

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

Voice Message Upload, Download, and Playback

API	Description
UploadRecordedFile	Uploads an audio file.
DownloadRecordedFile	Downloads an audio file.
PlayRecordedFile	Plays back an audio file.
StopPlayFile	Stops playing back an audio file.
GetFileSize	Gets the audio file size.

GetVoiceFileDuration	Gets the audio file duration.

Uploading an audio file

This API is used to upload an audio file.

API prototype

```
ITMGPTT int UploadRecordedFile (string filePath)
```

Parameter	Type	Description
filePath	String	Path of the uploaded audio file, which is a local path.

Sample code

```
ITMGContext.GetInstance().GetPttCtrl().UploadRecordedFile(filePath);
```

Callback for audio file upload completion

A callback will be executed through a delegate function to pass a message when the upload of audio file is completed.

API prototype

```
public delegate void QAVUploadFileCompleteCallback(int code, string filepath,
string fileid);
public abstract event QAVUploadFileCompleteCallback OnUploadFileComplete;
```

Parameter	Type	Description
code	int	`0` : Recording is completed.
filepath	string	Path of the stored recording file
fileid	string	File URL

Error codes

Error Code	Cause	Suggestion
8193	An error occurred while accessing the file during upload.	Ensure the existence of the file and the validity of the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and

		whether the voice messaging and speech-to-text feature is initialized.
8195	Network error.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device network can normally access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device network can normally access the internet.
8200	`appinfo` is not set.	Check whether the `apply` API is called or whether the input parameter is not specified or null.

Sample code

```
// Listen on an event
ITMGContext.GetInstance().GetPttCtrl().OnUploadFileComplete +=new
QAVUploadFileCompleteCallback (OnUploadFileComplete);
// Process the event listened on
void OnUploadFileComplete(int code, string filepath, string fileid){
    // Callback for audio file upload completion
}
```

Downloading the audio file

This API is used to download an audio file.

API prototype

```
ITMGPTT DownloadRecordedFile (string fileID, string downloadFilePath)
```

Parameter	Type	Description
fileID	String	File URL
downloadFilePath	String	Local path of the saved file, which must be accessible and cannot be the `fileid` .

Sample code


```
ITMGContext.GetInstance().GetPttCtrl().DownloadRecordedFile(fileId, filePath);
```

Callback for audio file download completion

A callback will be executed through a delegate function to pass a message when the download of audio file is completed.

API prototype

```
public delegate void QAVDownloadFileCompleteCallback(int code, string filepath,
string fileid);
public abstract event QAVDownloadFileCompleteCallback OnDownloadFileComplete;
```

Parameter	Type	Description
code	int	`0` : Recording is completed.
filepath	string	Path of the stored recording file
fileid	string	URL of the recording file, which will be retained on the server for 90 days.

Error codes

Error Code	Cause	Suggestion
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
12291	A network storage system exception occurred.	The server failed to get the audio file. Check whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	A server file system error occurred.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed while getting the download parameters.	Check whether the device can access the internet.
12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.

12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	`appinfo` is not set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.

Sample code

```
// Listen on an event
ITMGContext.GetInstance().GetPttCtrl().OnDownloadFileComplete +=new
QAVDownloadFileCompleteCallback(OnDownloadFileComplete);
// Process the event listened on
void OnDownloadFileComplete(int code, string filepath, string fileid){
    // Callback for audio file download completion
}
```

Playing back audio

This API is used to play back audio.

API prototype

```
ITMGPTT PlayRecordedFile(string filePath)
ITMGPTT PlayRecordedFile(string filePath,int voiceType);
```

Parameter	Type	Description
filePath	string	Local audio file path
voicetype	int	Voice changing type. For more information, see Voice Changing.

Error codes

Error Code	Cause	Suggestion
20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
ITMGContext.GetInstance().GetPttCtrl().PlayRecordedFile(filePath);
```

Callback for audio playback

A callback will be executed through a delegate function to pass a message when an audio file is played back.

API prototype

```
public delegate void QAVPlayFileCompleteCallback(int code, string filepath);
public abstract event QAVPlayFileCompleteCallback OnPlayFileComplete;
```

Parameter	Type	Description
code	int	`0` : Playback is completed.
filepath	string	Path of the stored recording file

Error codes

Error Code	Cause	Suggestion
20481	An initialization error occurred.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	Empty parameters.	Check whether the API parameters in the code are correct.
20484	Internal error	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
// Listen on an event:
ITMGContext.GetInstance().GetPttCtrl().OnPlayFileComplete +=new
QAVPlayFileCompleteCallback(OnPlayFileComplete);
// Process the event listened on:
void OnPlayFileComplete(int code, string filepath){
    // Callback for audio playback
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

API prototype

```
ITMGPTT int StopPlayFile()
```

Sample code

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

Getting audio file size

This API is used to get the size of an audio file.

API prototype

```
ITMGPTT GetFileSize(string filePath)
```

Parameter	Type	Description
filePath	String	Path of the audio file, which is a local path.

Sample code

```
int fileSize = ITMGContext.GetInstance().GetPttCtrl().GetFileSize(filepath);
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

API prototype

```
ITMGPTT int GetVoiceFileDuration(string filePath)
```

Parameter	Type	Description
filePath	String	Path of the audio file, which is a local path.

Sample code

```
int fileDuration =
ITMGContext.GetInstance().GetPttCtrl().GetVoiceFileDuration(filepath);
```

Fast Recording-to-Text Conversion

API	Description
SpeechToText	Converts speech to text.

Converting audio file to text

This API is used to convert a specified audio file to text.

API prototype

```
ITMGPTT int SpeechToText(String fileID)
```

Parameter	Type	Description
fileID	String	Audio file URL

Sample code

```
ITMGContext.GetInstance().GetPttCtrl().SpeechToText(fileID);
```

Translating audio file into text in specified language

This API can specify a language for recognition or translate the text recognized in speech into a specified language and return the translation.

Notes

Translation incurs additional fees. For more information, see Purchase Guide.

API prototype

```
ITMGPTT int SpeechToText(String fileID,String speechLanguage)
ITMGPTT int SpeechToText(String fileID,String speechLanguage,String
translatelanguage)
```

Parameter	Type	Description
fileID	String	URL of the audio file, which will be retained on the server for 90 days.
speechLanguage	String	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translatelanguage	String	The language in which the audio file is to be translated into text. For parameters,

		see Language Parameter Reference List.

Sample code

```
ITMGContext.GetInstance().GetPttCtrl().SpeechToText(fileID,"cmn-Hans-CN","cmn-Hans-CN");
```

Callback for recognition

A callback will be executed through a delegate function to pass a message when a specified audio file is recognized and converted to text.

API prototype

```
public delegate void QAVSpeechToTextCallback(int code, string fileid, string result);
public abstract event QAVSpeechToTextCallback OnSpeechToTextComplete;
```

Parameter	Type	Description
code	int	`0` : Recording is completed.
fileid	string	URL of the audio file, which will be retained on the server for 90 days.
result	string	Converted text

Error codes

Error Code	Cause	Suggestion
32769	Internal error	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	Network connection failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	`appinfo` is not set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	The speech-to-text conversion	Check whether the API parameter `fileid` in the code is

	parameter is incorrect.	empty.
32785	A speech-to-text translation error occurred.	An error occurred in the voice messaging and speech-to-text feature on the backend. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

Sample code

```
// Listen on an event
ITMGContext.GetInstance().GetPttCtrl().OnSpeechToTextComplete += new
QAVSpeechToTextCallback(OnSpeechToTextComplete);
// Process the event listened on
void OnSpeechToTextComplete(int code, string fileid, string result){
    // Callback for recognition
}
```

Voice Message Volume Level APIs

API	Description
GetMicLevel	Gets the real-time mic volume level.
SetMicVolume	Sets the recording volume level.
GetMicVolume	Gets the recording volume level.
GetSpeakerLevel	Gets the real-time speaker volume level.
SetSpeakerVolume	This API is used to set the playback volume.
GetSpeakerVolume	Gets the playback volume level.

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT int GetMicLevel()
```

Sample code

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

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT int SetMicVolume(int vol)
```

Sample code

```
ITMGContext.GetInstance().GetPttCtrl().SetMicVolume(100);
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT int GetMicVolume()
```

Sample code

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

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT int GetSpeakerLevel()
```

Sample code

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

Setting the playback volume of voice message

This API is used to set the playback volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT int SetSpeakerVolume(int vol)
```

Sample code

```
ITMGContext.GetInstance().GetPttCtrl().SetSpeakerVolume(100);
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT int GetSpeakerVolume()
```

Sample code

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

Advanced APIs

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
ITMGContext  abstract string GetSDKVersion()
```

Sample code

```
ITMGContext.GetInstance().GetSDKVersion();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext  SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: `TMG_LOG_LEVEL_INFO` .
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: `TMG_LOG_LEVEL_ERROR` .

`ITMG_LOG_LEVEL` description:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContext.GetInstance().SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before initialization.

Platform	Path

Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents
Android	/sdcard/Android/data/xxx.xxx.xxx/files
macOS	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext  SetLogPath(string logDir)
```

Parameter	Type	Description
logDir	String	Path

Sample code

```
ITMGContext.GetInstance().SetLogPath(path);
```

Project Export

Last updated：2024-12-18 14:51:13

This document describes how to configure a Unity project export for the GME APIs for Unity.

Export for iOS

When exporting a Unity project as an Xcode project, you need to process GME dynamic libraries. The processing steps vary by Unity version.

1. Process dynamic libraries (for Unity 2019 or later)

Configuration principle

Create an `Editor OnPostprocessBuild` script and use the `UnityEditor.iOS.Xcode.Extensions.PBXProjectExtensions.AddFileToEmbedFrameworks` API, which will automatically copy the dynamic libraries to the `framework` directory of the final output bundle and sign them.
You can add or remove dynamic libraries based on the required features and determine the list of imported frameworks in the sample code based on the dynamic library list. For more information on dynamic library features, see SDK Version Upgrade Guide.

```
string[] framework_names = {
    "libgme_fdkaac.framework",
    "libgme_lamemp3.framework",
    "libgme_ogg.framework",
    "libgme_soundtouch.framework"
};
```

Sample code

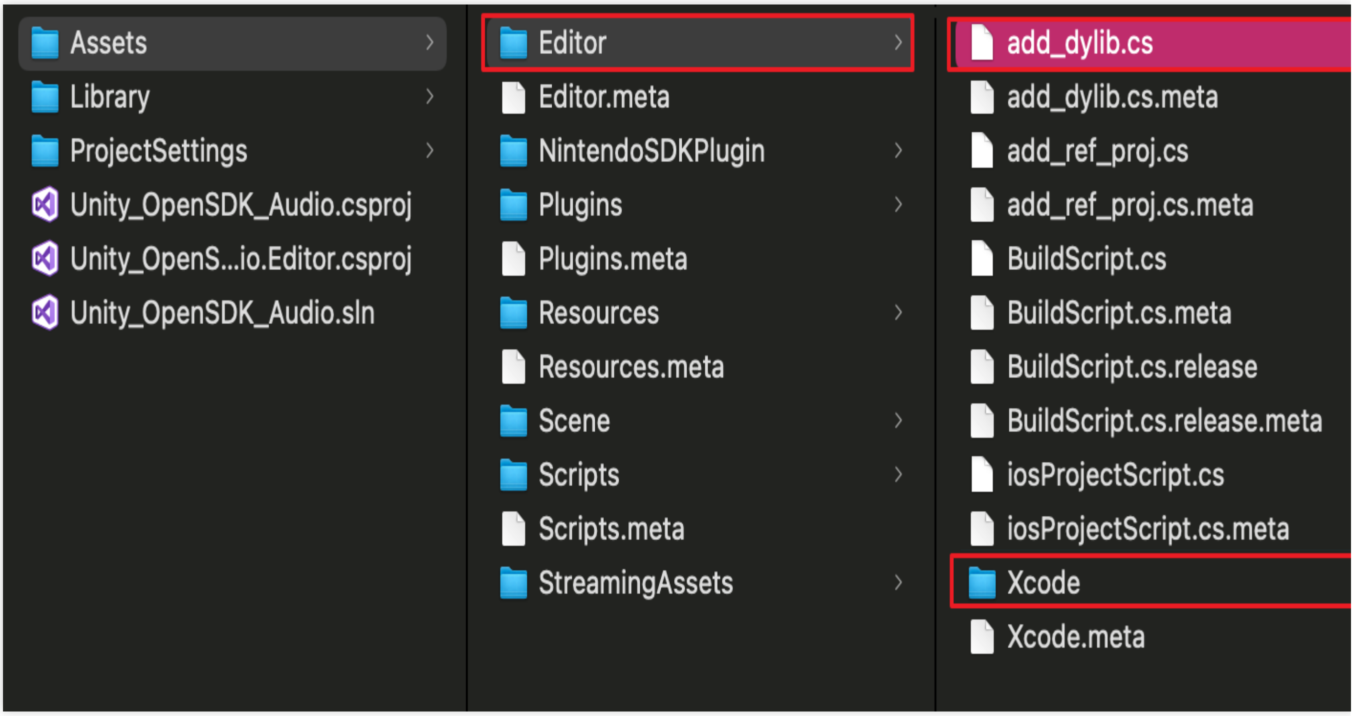
You can refer to the `add_dylib.cs` script file in the demo project and put this part of code in the `Editor` folder of the project as needed.

```
[UnityEditor.Callbacks.PostProcessBuild(1002)]
public  static void OnPostprocessBuild (UnityEditor.BuildTarget BuildTarget, string
    if (BuildTarget == UnityEditor.BuildTarget.iOS) {
        UnityEngine.Debug.Log ("OnPostprocessBuild add_dylib:" + path);
#if UNITY_EDITOR_OSX || UNITY_STANDALONE_OSX
        {
            string projPath = UnityEditor.iOS.Xcode.PBXProject.GetPBXProjectPath (p
```

```
            UnityEditor.iOS.Xcode.PBXProject proj = new UnityEditor.iOS.Xcode.PBXPr

            proj.ReadFromString (System.IO.File.ReadAllText (projPath));
            // string targetGuid = proj.TargetGuidByName (UnityEditor.iOS.Xcode.PBX
            string targetGuid = proj.GetUnityMainTargetGuid();  // 2019

            // Delete according to the imported frameworks
            string[] framework_names = {
                "libgme_fdkaac.framework",
                "libgme_lamemp3.framework",
                "libgme_ogg.framework",
                "libgme_soundtouch.framework"
            };

            for (int i = 0; i < framework_names.Length; i++)
            {
                string framework_name = framework_names[i];
                string dylibGuid = null;
                dylibGuid = proj.FindFileGuidByProjectPath("Frameworks/Plugins/iOS/

                if (dylibGuid == null) {
                    UnityEngine.Debug.LogWarning (framework_name + " guid not found
                } else {
                    UnityEngine.Debug.LogWarning (framework_name + " guid:" + dylib
                    // proj.AddDynamicFramework (targetGuid, dylibGuid);
                    UnityEditor.iOS.Xcode.Extensions.PBXProjectExtensions.AddFileTo

                    proj.AddBuildProperty(targetGuid, "LD_RUNPATH_SEARCH_PATHS", "@
                    System.IO.File.WriteAllText (projPath, proj.WriteToString ());
                }
            }
        }
#endif
    }
}
```

2. Process dynamic libraries (for Unity versions earlier than 2019)

Currently, only Unity 2019 or later can use `UnityEditor.iOS.Xcode.Extensions`. You can export the `UnityEditor.iOS.Xcode` package from a later version for use in an earlier version, or directly decompress the attachment UnityEditorAV.iOS.XCode.zip and place it in the `Editor` folder of the project directory.

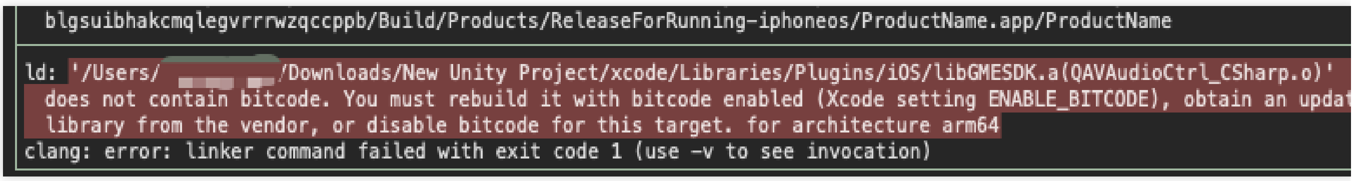


3. Export the Xcode project

Make sure that the Xcode version is 10.0 or later. Export the Xcode project from the Unity Editor.

4. Disable Bitcode

If the following error occurs during the compilation, disable Bitcode. Search for Bitcode in **Targets** > **Build Settings** and set the corresponding option to `NO` .



5. Add access to iOS

Required background modes: Allows running in the background (optional).

Microphone Usage Description: Allows access to microphone.

6. Add library files

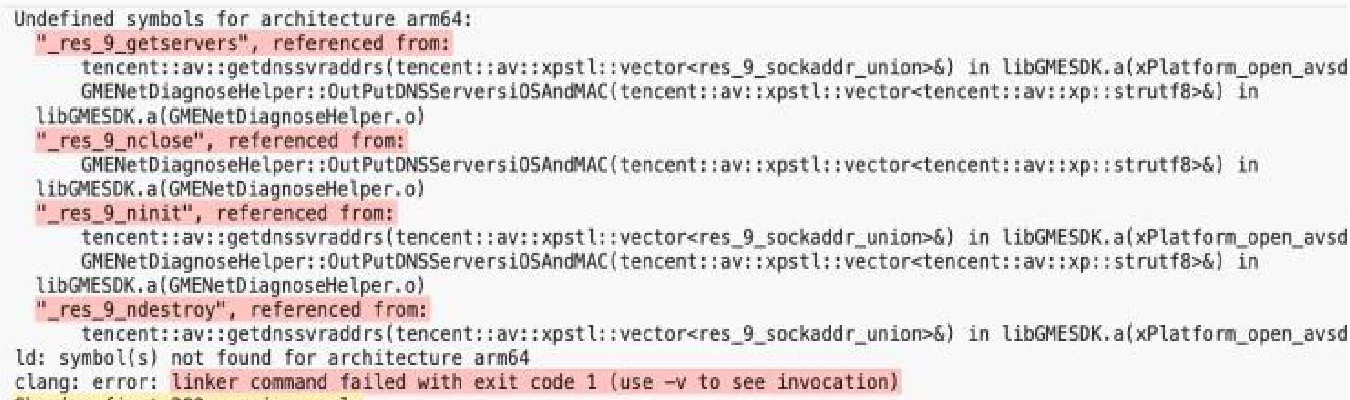
If the following error occurs during compilation, please complete the library files.

The list of library files is as follows:

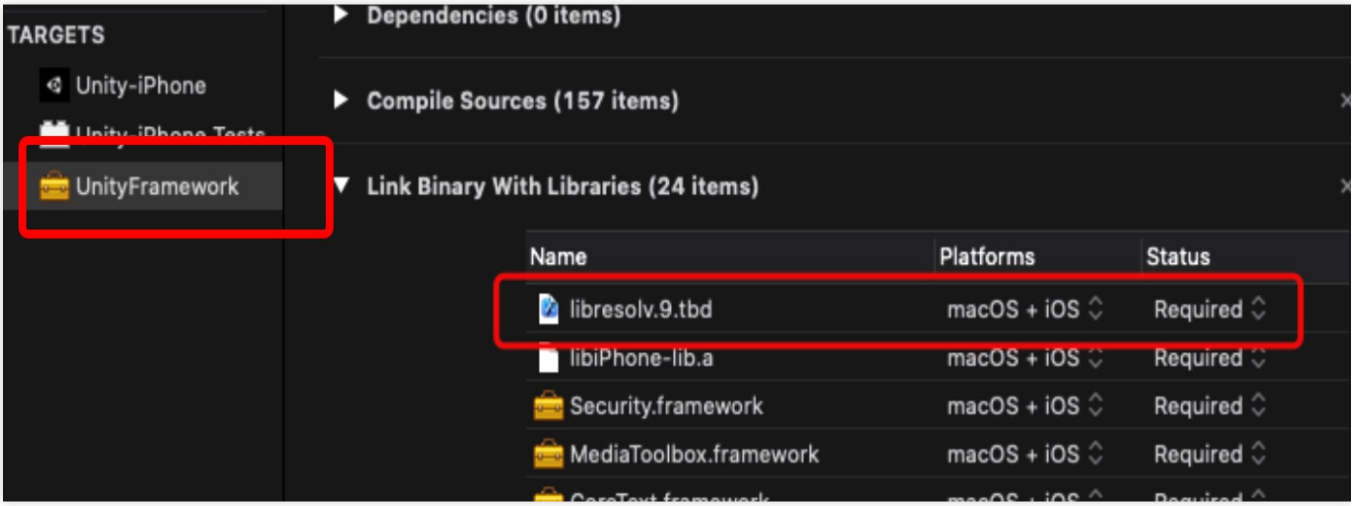
```
libc++.tbd
CoreMedia.framework
libresolv.tbd
AVFoundation.framework
Security.framework
CoreAudio.framework
AudioToolbox.framework
libiconv.tbd
libz.tbd
SystemConfiguration.framework
OpenAL.framework
```

7. Add `libresolv9.tbd`

If the following error occurs:



Add `libresolv9.tbd` to **UnityFramework**.



8. FAQs about export

If you have any questions during exporting, see Exporting for iOS.

Export for Android

1. Delete unnecessary .lib files

The GME SDK for Unity provides lib files for arm64-v8a, armeabi-v7a, and x86 by default. Please delete unnecessary files as needed.

Architecture loss

A crash will occur if the exported Android executable file lacks the specified architecture.

After the executable `apk` file is exported, if a black screen or crash occurs when you open it, it is generally caused by the lack of corresponding architecture `lib` file. Please add or delete the corresponding architecture `lib` file according to the project.

2. Configure permissions

2.1 Required permissions

Add the following permissions in the `AndroidManifest.xml` file of the project:

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

2.2 Adding permissions as needed

Add the following permissions in the `AndroidManifest.xml` file of the project as needed:

Read/Write permission

Bluetooth permission

The read/write permission is not required. Determine whether to add it according to the following rules:

If you use the default log path (/SDCARD/Android/Data/files), it means that you do not call SetLogPath, and do not need Write_External_Storage permission.

If you call the setLogPath API to set the log path to an external storage device, and the storage path of the voice message recording is an external storage device, you need to apply for the Write_External_Storage permission to the user and get the user's approval.

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Add the Bluetooth permission according to the following rules:

If `targetSDKVersion` in the project is v30 or earlier:

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
```

If `targetSDKVersion` in the project is v31 or later:

```
<uses-permission android:name="android.permission.BLUETOOTH"
android:maxSdkVersion="30" />
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
```

3. FAQs about export

If you have any questions during exporting, see Exporting for Android.

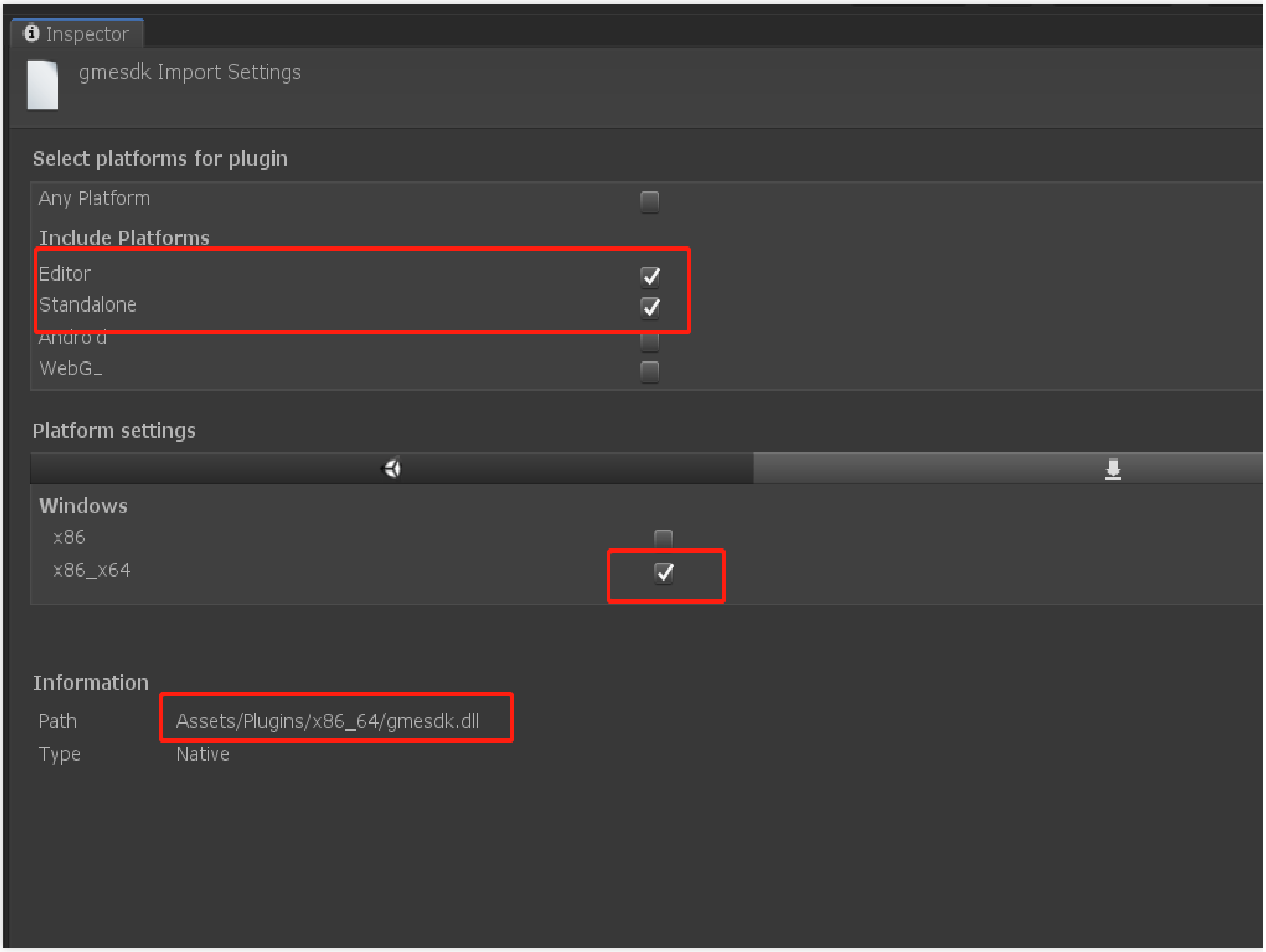
Export for Windows

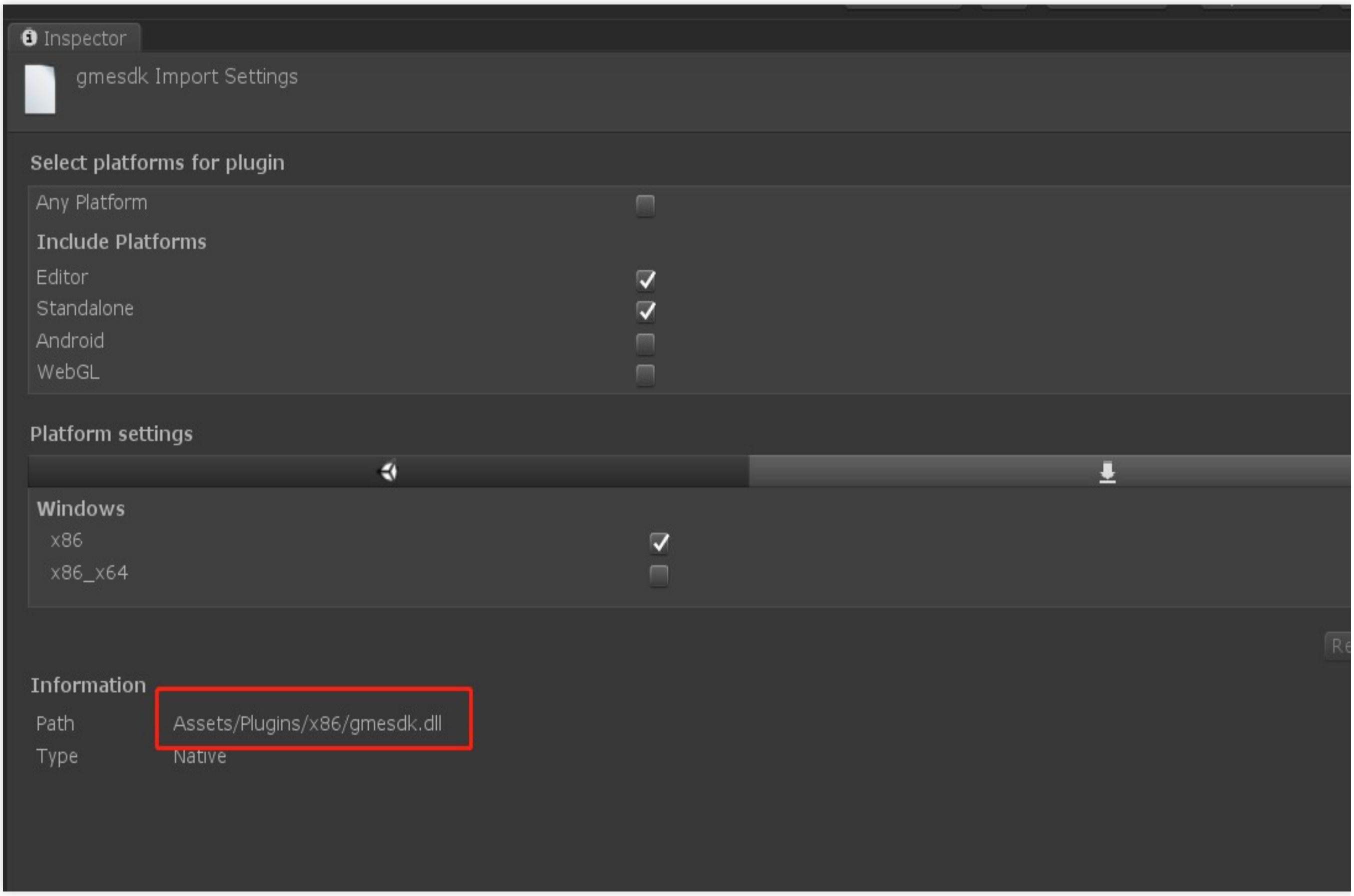
If you have any questions during exporting, see Exporting for Windows.

Export for WebGL

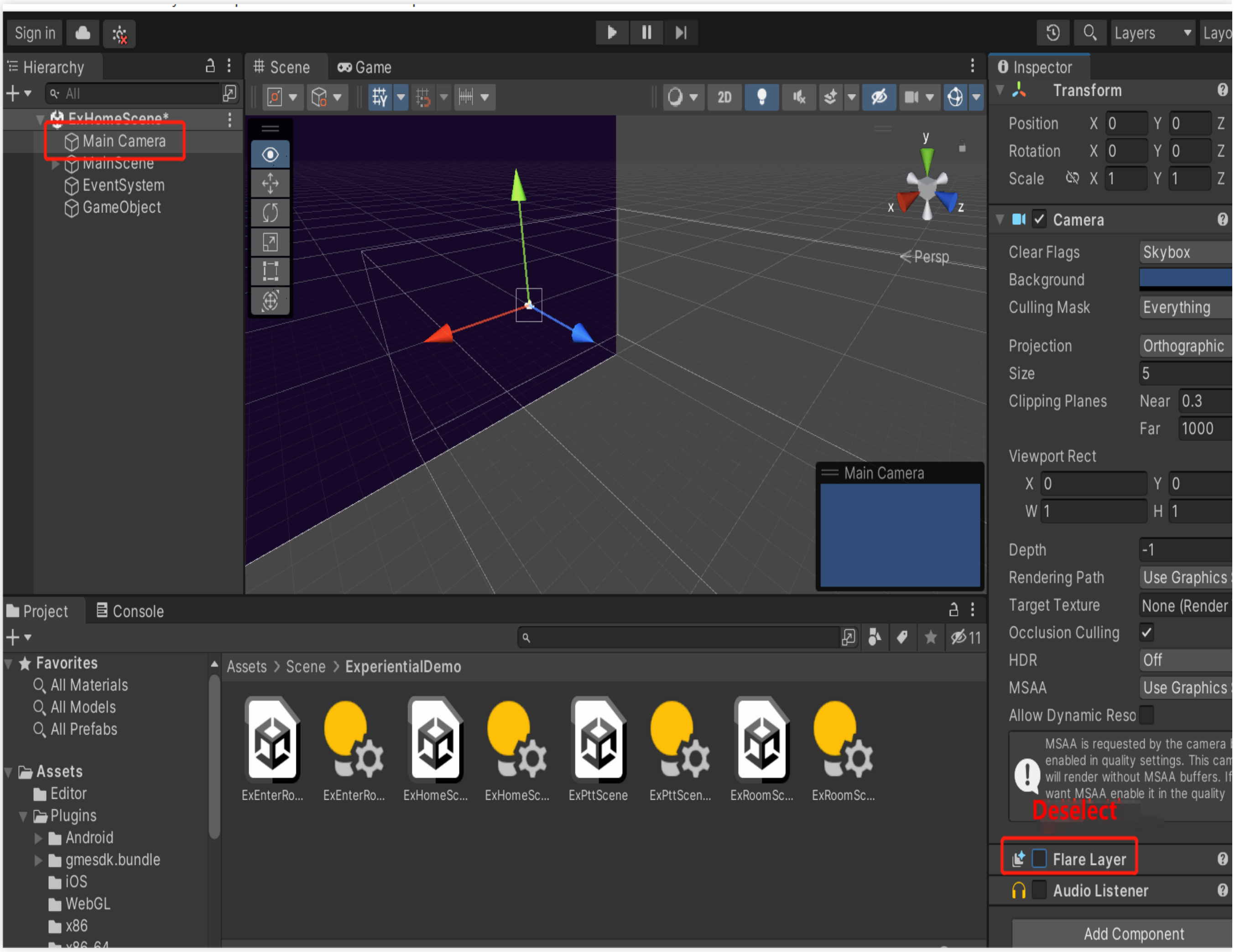
1. Configure WebGL plugins

Set the scope of `gmesdk.dll` on Windows to prevent it from conflicting with `gmesdk` on WebGL.

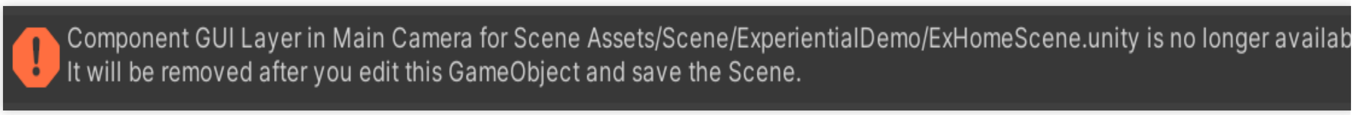

Inspector
gmesdk Import Settings
Select platforms for plugin
Any Platform
Include Platforms
Editor
Standalone
Android
WebGL
Platform settings
Windows
x86
x86_x64
Information
Path
Assets/Plugins/x86_64/gmesdk.dll
Type
Native



2. Disable Flare Layer (on Unity 2018 or later)

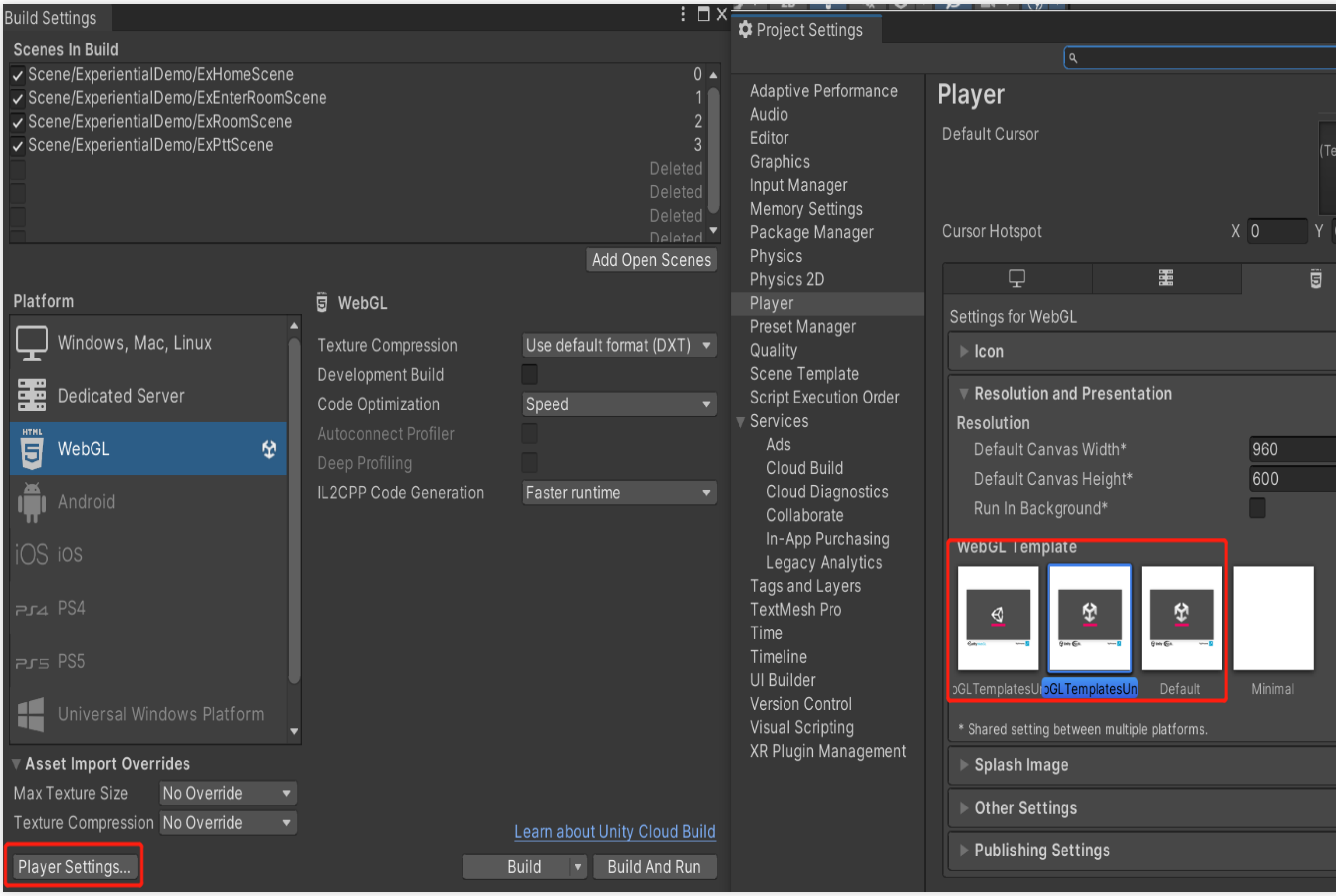


As certain Unity versions no longer support the Flare Layer mode in `MainCamera` , you need to deselect Flare Layer in the scene to be built; otherwise, the following error will be reported:



3. Select a template

When exporting to WebGL, select a WebGL template of GME to ensure that the relevant dependent libraries are imported to the build. The project will use the `GMEWebGLTemplatesUnity2018` template by default, which supports Unity 2018 and 2019. For Unity 2020 and 2021, you need to use `GMEWebGLTemplatesUnity2021` to create the build.



4. Import frontend libraries

Before importing GME for WebGL to your project, you need to manually import frontend libraries, place the library files in the corresponding import locations, and add the audio tag as shown below. If you want the above operations to be automatically completed every time you build a Unity project, you can add the corresponding template to your project by referring to the GME demo for WebGL.

```
<!DOCTYPE html>
<html lang="en-us">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>Unity WebGL Player | %UNITY_WEB_NAME%</title>
    <link rel="shortcut icon" href="TemplateData/favicon.ico">
    <link rel="stylesheet" href="TemplateData/style.css">
    <script src="https://code.jquery.com/jquery-3.5.0.min.js"></script>
    <script src="TemplateData/UnityProgress.js"></script>
    <script src="%UNITY_WEBGL_LOADER_URL%"></script>
    <script src="WebRTCService.js"></script>
    <script src="implementation.js"></script>
    <script>
      var unityInstance = UnityLoader.instantiate("unityContainer", "%UNITY_WEBGL_BUILD_URL%", {onProgress: UnityProgres
    </script>
  </head>
  <body>
    <div id="gme-audio-wrap"></div>
    <div class="webgl-content">
      <div id="unityContainer" style="width: %UNITY_WIDTH%px; height: %UNITY_HEIGHT%px"></div>
      <div class="footer">
        <div class="webgl-logo"></div>
        <div class="fullscreen" onclick="unityInstance.SetFullscreen(1)"></div>
        <div class="title">%UNITY_WEB_NAME%</div>
      </div>
    </div>
  </body>
</html>
```

1. Import frontend libraries

2. Add the audio tag

5. FAQs about export

If you have any questions during export, see Program Export.

SDK for Unreal Engine

Integrating SDK

Last updated：2024-01-18 15:02:24

Overview

This document describes how to configure a Unreal Engine project for the GME APIs for Unreal Engine.

SDK Download

1. Download the applicable demo and SDK. For more information, see SDK Download Guide.
2. Locate the SDK resources for Unreal Engine on the page.
3. Click **Download**. After decompression, the downloaded SDK resources include the following files:

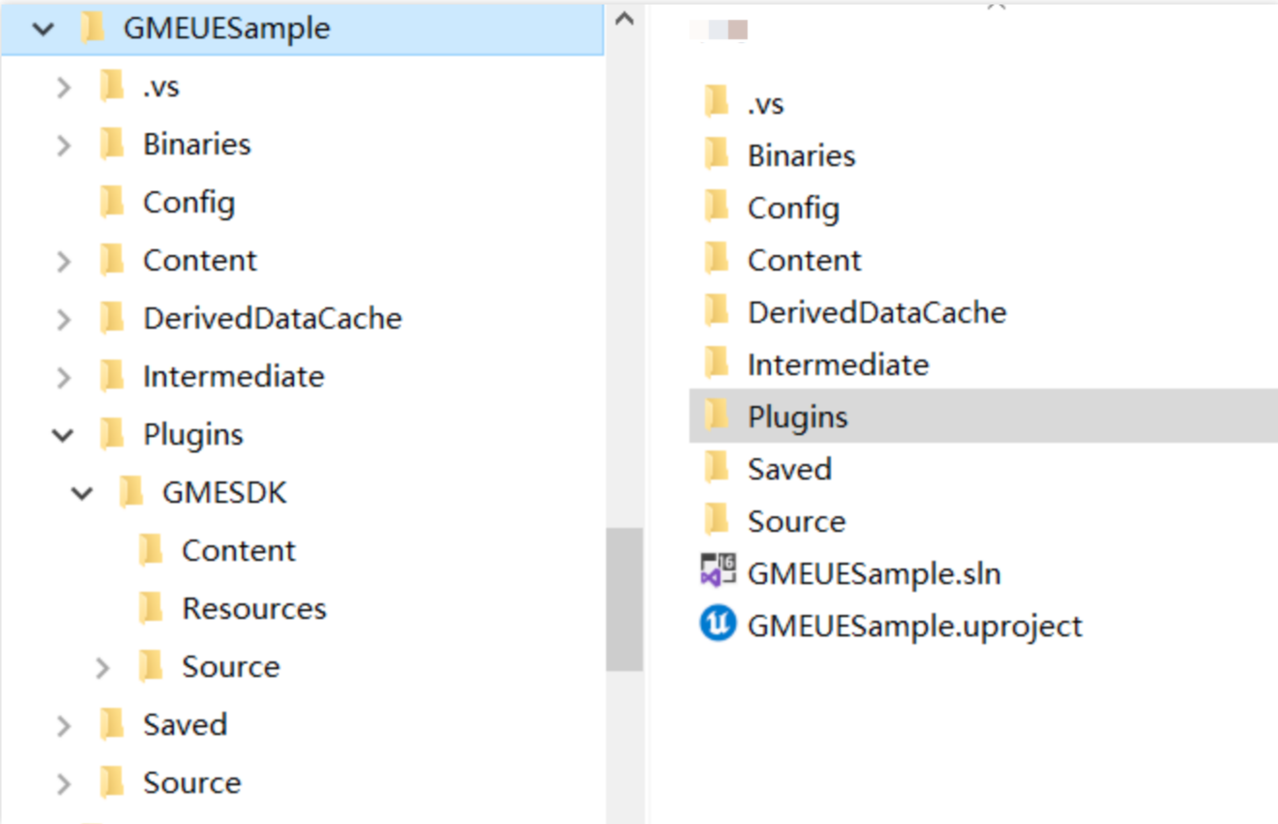
File name	Description	Purpose
GMESDK.uplugin	`.uplugin` file	Plugin configuration file
Resources	Plugin resource file	Plugin resource file
Source	SDK file	SDK library files and code files for various platforms, such as header files

Platforms supported by the SDK for Unreal Engine:
SDK for Unreal Engine has integrated Windows, macOS, Android, and iOS platform architectures. If you need console platform architectures, contact us.

Project Configuration

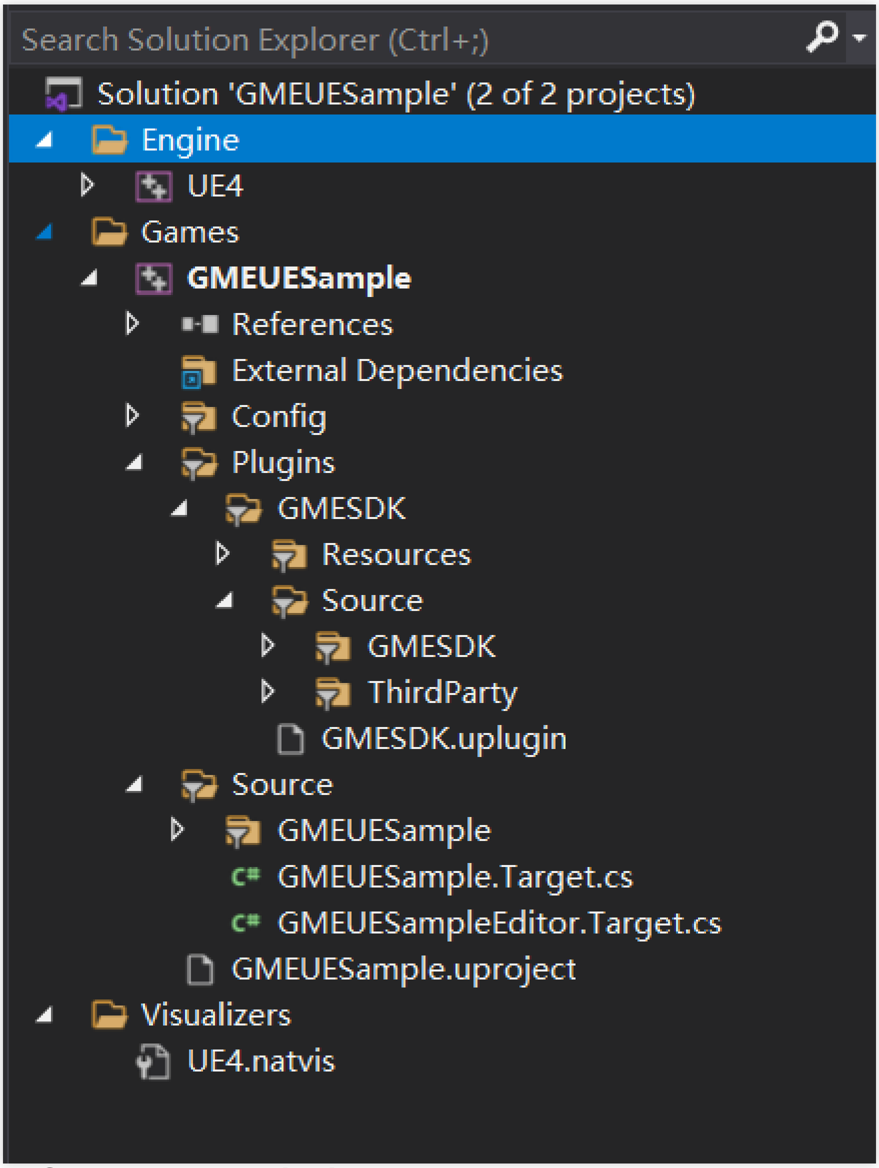
Step 1: Import Plugins files

If no `Plugins` folder exists in the root directory of the game project (the directory of the `*.uproject` file), create one first and copy the GME SDK to the `Plugins` folder. Then, you will see the directory structure as shown below:



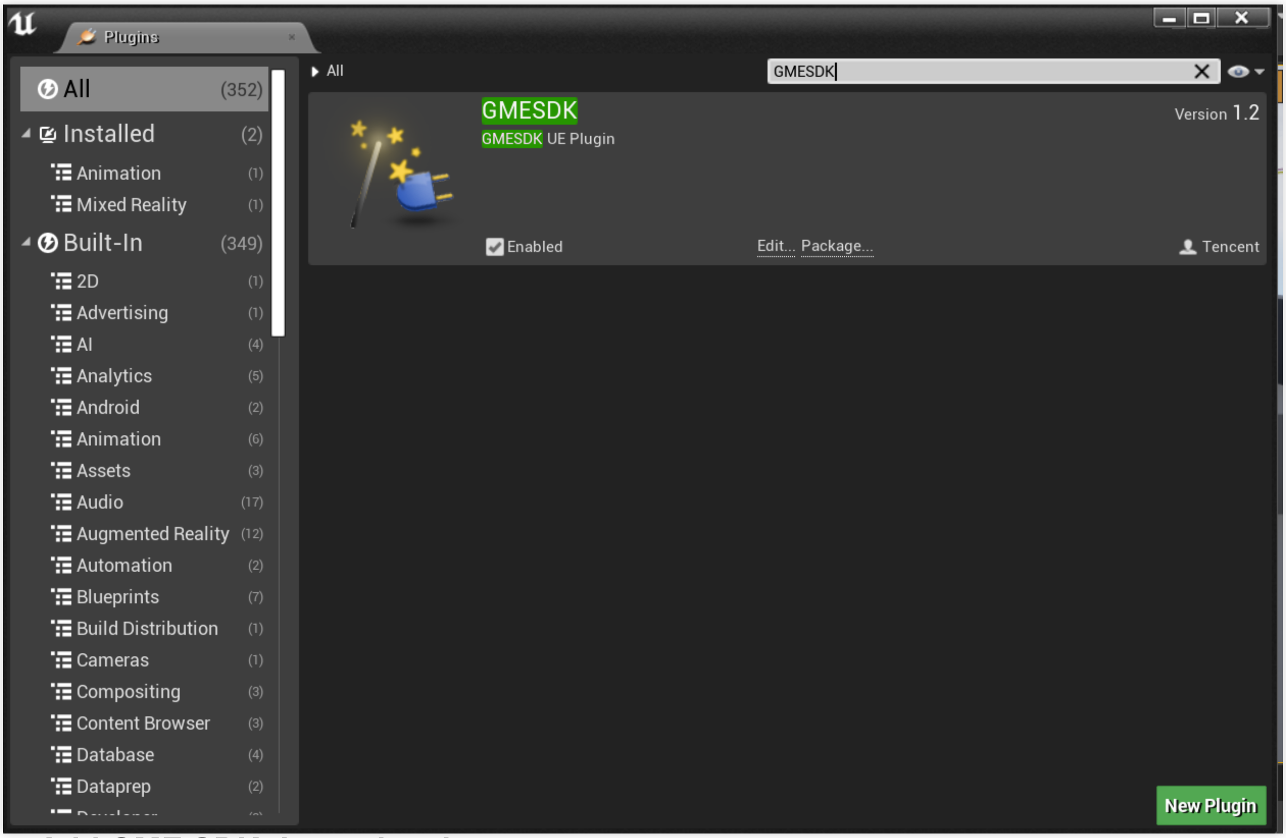
Step 2: Compile the plugin

Refresh the C++ project in Visual Studio/Xcode, open it, and you will see the directory structure like the following. Then, compile the game.



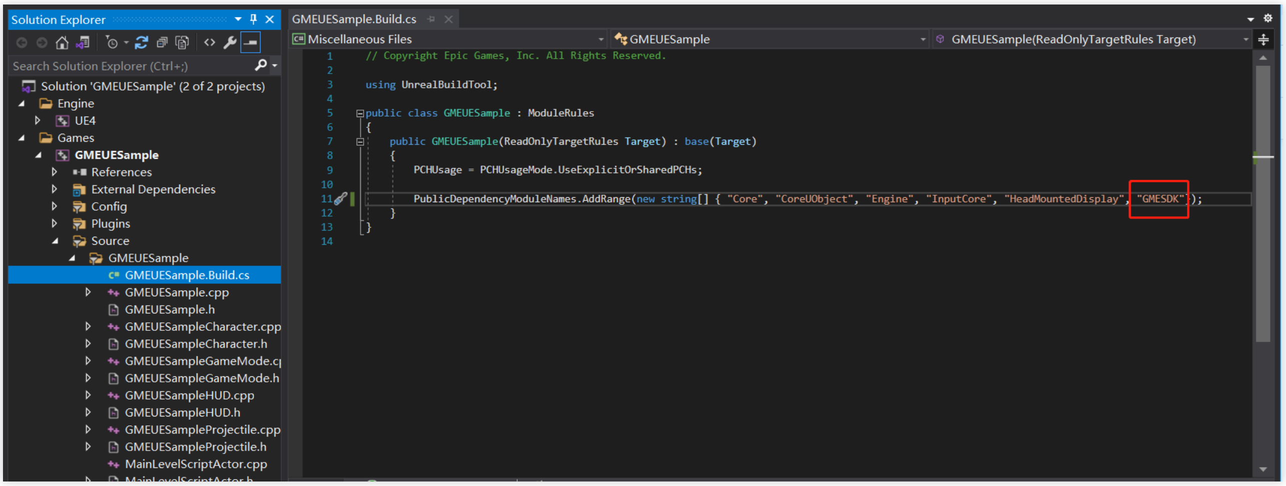
Step 3. Complete compilation

After completing the compilation, restart Unreal Engine Editor. In Unreal Engine Editor, open the plugin manager, and you can see that the GME SDK has been imported into the project. Make sure that the GME SDK is enabled.



Step 4. Add GME SDK dependencies

Add GME SDK dependencies to the `.build.cs` file of the game project.



Adaptations of Different Unreal Engine Versions

Unreal Engine 4.21 and later

If you are using Unreal Engine 4.21 or later, you need to add the following code after downloading the GME sample code for Unreal Engine:

```
AUEDemoLevelScriptActor::AUEDemoLevelScriptActor()
```

```
{
    PrimaryActorTick.bCanEverTick = true;
}
```

Note:

Tick is disabled by default and must be enabled manually.

Unreal Engine 4.26

If you are using Unreal Engine 4.26, you need to download the adaptation file and import it into the project. The downloaded file contains two folders: `Source` and `Plugins` .

For a demo project, import both folders into the project in an overwriting manner.

If you only need the GME SDK, import the `Plugins` folder only.

Speech-to-Text Service

Last updated：2024-01-18 15:02:24

This document describes how to integrate with and debug GME client APIs for the voice messaging and speech-to-text services for Unreal Engine.

Key Considerations for Using GME

GME provides the real-time voice and voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

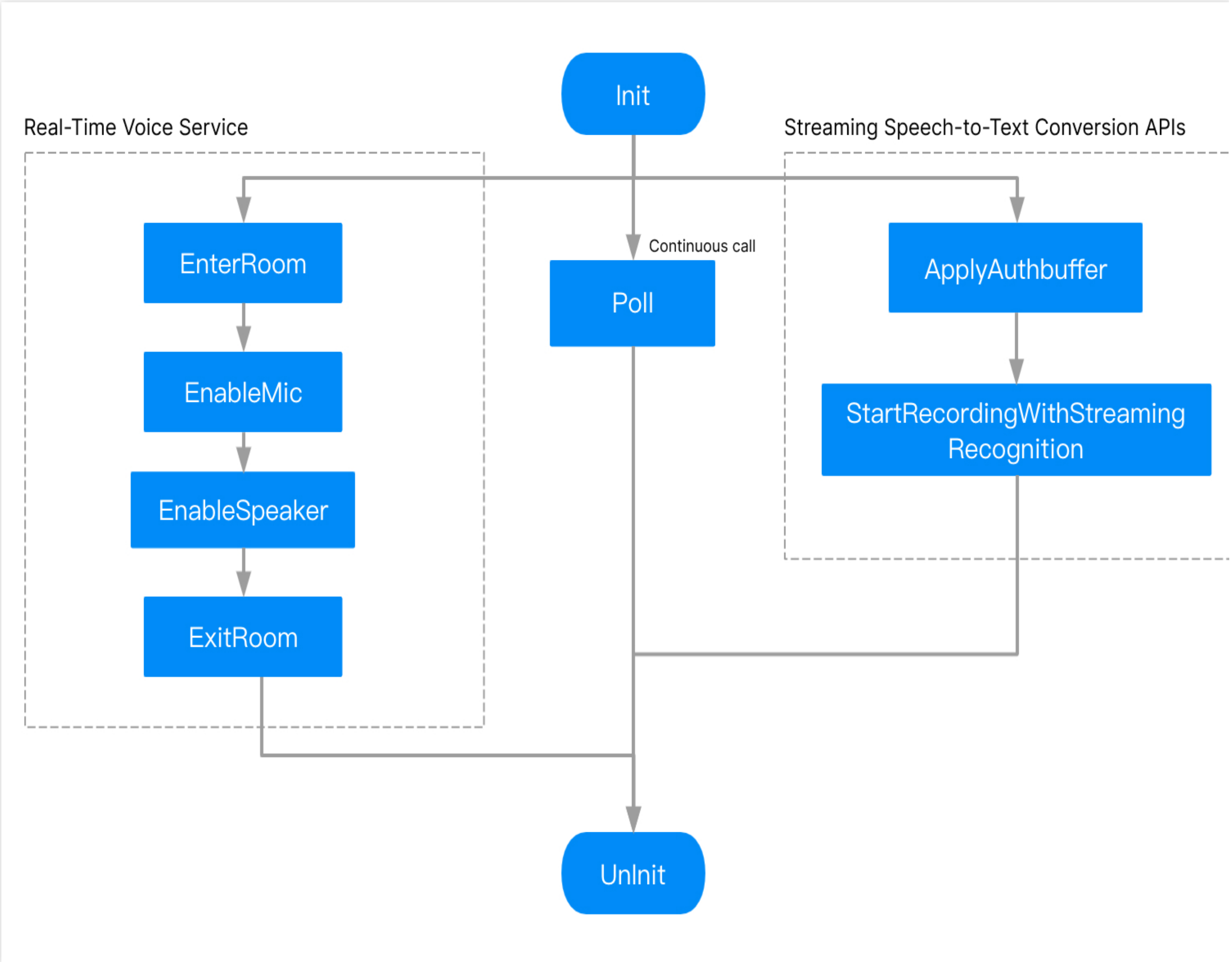
Notes

You have created a GME application and obtained the SDK AppID and key. For more information, see Activating Services.
You have activated **GME real-time voice and voice messaging and speech-to-text services**. For more information, see Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error code, see Error Codes.
Note:
There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.
Non-streaming speech-to-text API ***SpeechToText()***: There can be up to 10 concurrent requests per account.
Streaming speech-to-text API ***StartRecordingWithStreamingRecognition()***: There can be up to 50 concurrent requests per account.
Real-time streaming speech-to-text API ***StartRealTimeASR()***: There can be up to 50 concurrent requests per account.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Initializing authentication, API: ApplyPTTAuthbuffer
4. Starting streaming speech recognition, API: StartRecordingWithStreamingRecognition
5. Stop recording, API: StopRecording
6. Uninitializing GME, API: UnInit

C++ classes

Class	Description
ITMGContext	Key APIs
ITMGPTT	Voice messaging and speech-to-text APIs

Key APIs

API	Description
Init	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME

Preparations

You need to import the header file `tmg_sdk.h` first before you can access GME. The classes in the header file inherit `ITMGDelegate` for message delivery and callback.

Sample code

```
#include "tmg_sdk.h"

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

Setting 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);
```

Message delivery

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
}
```

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

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

Parameter	Type	Description
sdkAppId	const char*	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	const char*	`openID` can only be in `Int64` type, which is passed in after being converted to a `const char*`. 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.

Returned values

Returned Value	Description

AV_OK = 0	Initialized SDK successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

Notes on 7015 error code

The 7015 error code is judged by md5. If this error is reported during integration, please check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party reinforcement, Unity packaging mechanism and other factors, the md5 of the library file will be affected, resulting in misjudgment. **Please ignore this error in the logic for official release**, and try to avoid displaying it in the UI.

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());
```

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.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
class ITMGContext {
protected:
    virtual ~ITMGContext() {}

public:
    virtual void Poll()= 0;
}
```

Sample code

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

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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext int Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid` , switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

API prototype

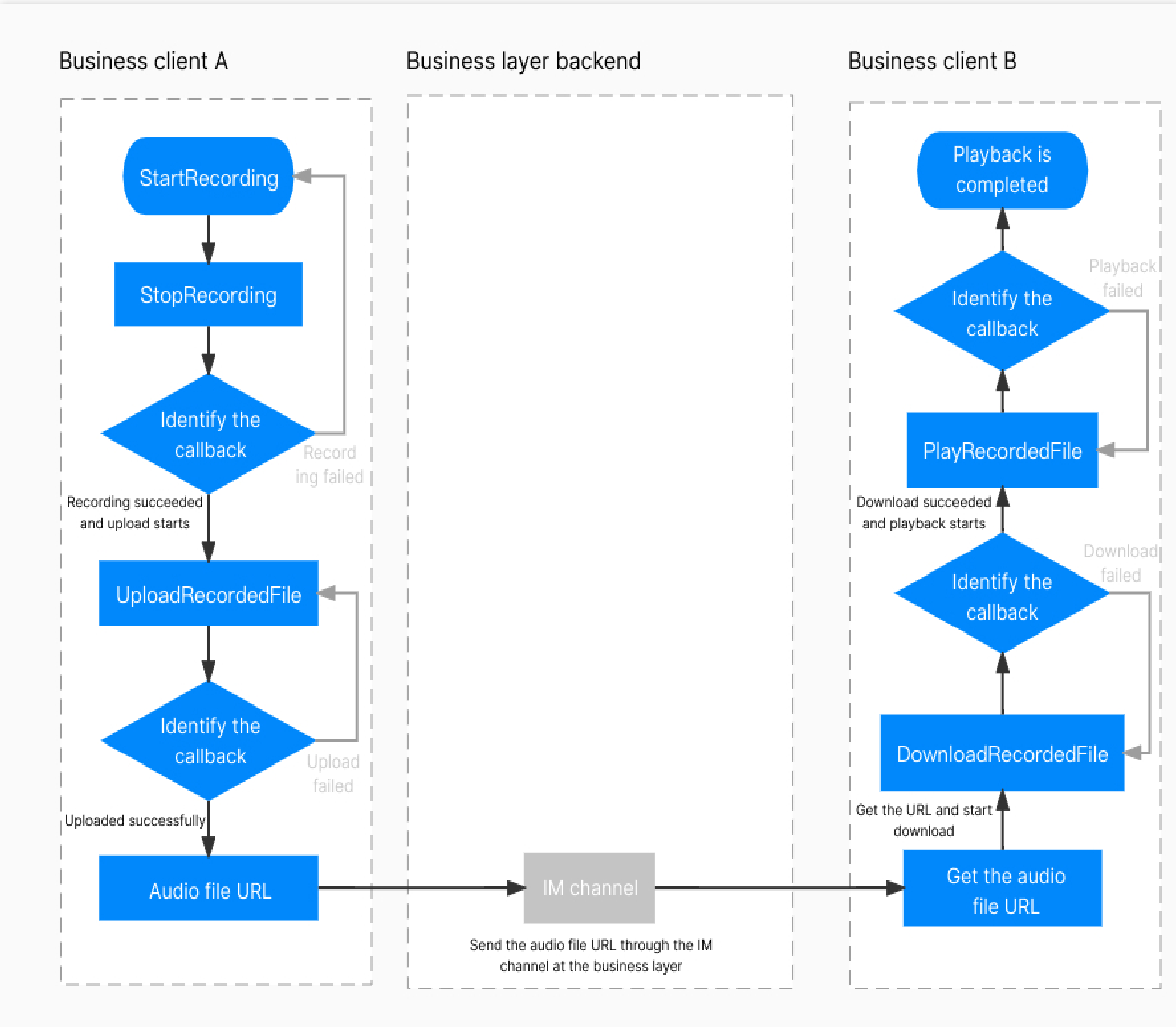
```
ITMGContext int Uninit()
```

Voice Messaging and Speech-to-Text Services

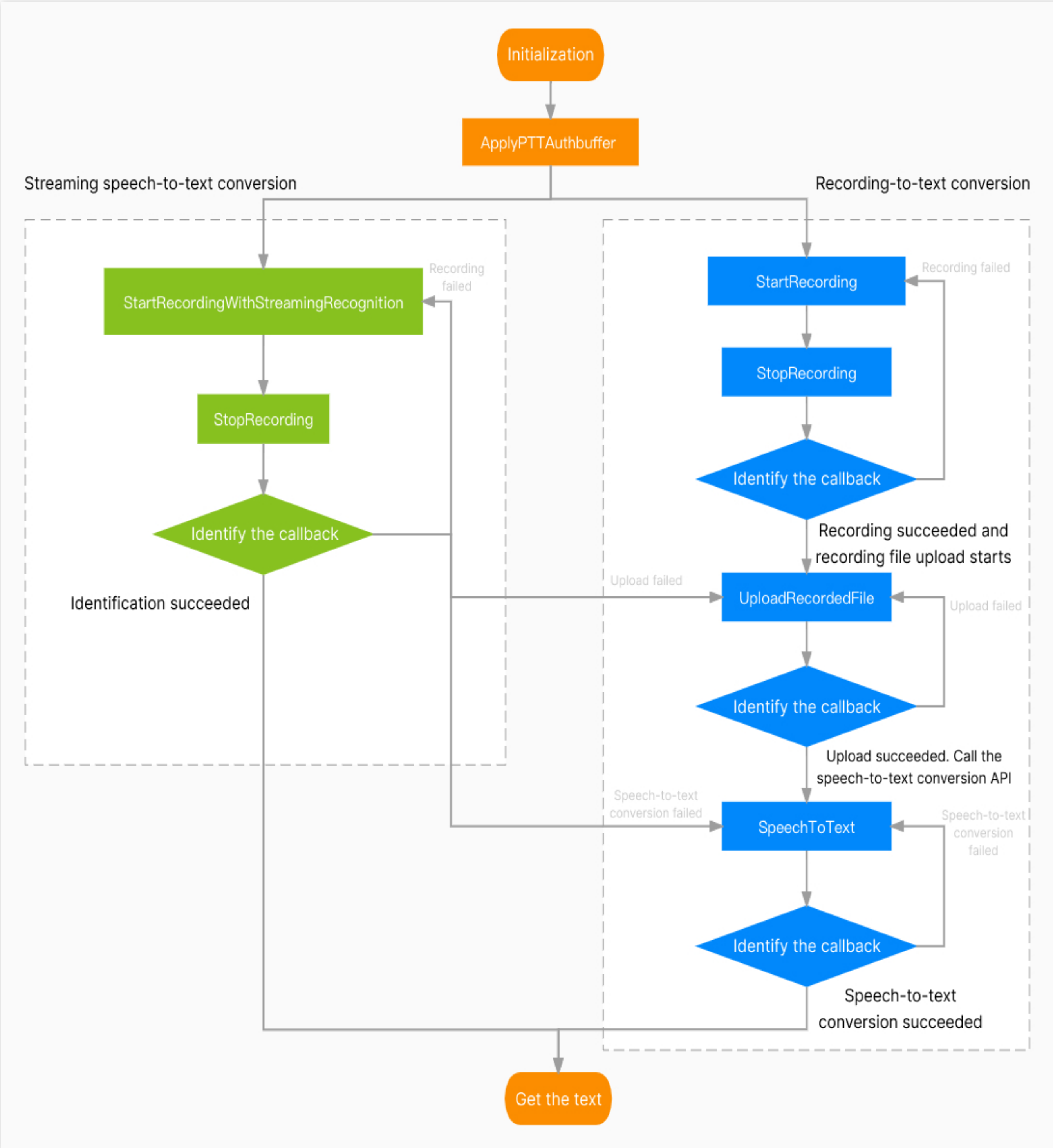
Note:

The speech-to-text service consists of fast recording-to-text conversion and streaming speech-to-text conversion.

You do not need to enter a voice chat room when using the voice message service.
The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum recording duration to 10 seconds, call the `SetMaxMessageLength` API to set it after initialization.



Flowchart for using the speech-to-text service



API	Description
GenAuthBuffer	Generates the local authentication key
ApplyPTTAuthbuffer	Initializes authentication
SetMaxMessageLength	Specifies the maximum length of voice message

Generating 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

```
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	`AppId` from the Tencent Cloud console.
strRoomID	const char*	Enter `null` or an empty string
strOpenID	const char*	User Identifier, which is the same as `openID` during initialization.
strKey	const char*	Permission key from the Tencent Cloud console.
strAuthBuffer	const char*	Returned `authbuff`
bufferLength	int	Length of the `authbuff` passed in. 500 is recommended.

Application authentication

After the authentication information is generated, the authentication is assigned to the SDK.

API prototype

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

Parameter	Type	Description
authBuffer	const char*	Authentication
authBufferLen	int	Authentication length

Sample code

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

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

API prototype

```
ITMGPTT virtual int SetMaxMessageLength(int msTime)
```

Parameter	Type	Description
msTime	int	Audio duration in ms. Value range: 1000 < msTime <= 58000

Sample code

```
int msTime = 10000;
ITMGContextGetInstance()->GetPTT()->SetMaxMessageLength(msTime);
```

Streaming Speech Recognition

Voice message and speech-to-text APIs

API	Description
StartRecordingWithStreamingRecognition	Starts streaming recording
StopRecording	Stops recording

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** Stop recording.

API 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	const char*	Path of stored audio file

speechLanguage	const char*	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translateLanguage	const char*	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

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

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen on callback messages in the `OnEvent` notification, which is as detailed below:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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` notification based on the actual needs. The passed parameters include the following four messages.

Message Name	Description
result	Return code indicating whether 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

Note:

The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

Error	Description	Suggested Solution

Code		
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 conversion 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.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

If the error code 4098 is reported, see FAQs for solutions.

Sample code

```
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
isComplete)
```

```
{
                    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 result
Json error")).c_str());
                    }
                    else
                    {
                        if (isComplete) {
                            ::SetWindowText(m_EditUpload.GetSafeHwnd(),
MByteToWChar(root["file_id"].asString()).c_str());
                        }
                        else {
                            std::string isruning =
"STREAMINGRECOGNITION_IS_RUNNING";
                            ::SetWindowText(m_EditUpload.GetSafeHwnd(),
MByteToWChar(isruning).c_str());
                        }
                    }
}
```

Voice Message Recording

The recording process is as follows: start recording -> stop recording -> return recording callback -> start the next recording.

Voice message and speech-to-text APIs

API	Description
StartRecording	Starts recording
PauseRecording	Pauses recording
ResumeRecording	Resumes recording
StopRecording	Stops recording
CancelRecording	Cancels recording

Starting recording

This API is used to start recording.

API prototype

```
ITMGPTT virtual int StartRecording(const char* fileDir)
```

Parameter	Type	Description
fileDir	const char*	Path of stored audio file

Sample code

```
char buffer[256]={0};
snprintf(buffer, sizeof(buffer), "%sunreal_ptt_local.file",
getFilePath().c_str());
ITMGContextGetInstance()->GetPTT()->StartRecording(buffer);
```

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 virtual int StopRecording()
```

Sample code

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

Callback for recording start

The recording start result will be returned through the callback.

To stop recording, call `StopRecording` . The callback for recording start will be returned after the recording is stopped.

Parameter	Type	Description
result	int32	0: recording is completed

filepath	FString	Path of stored recording file, which must be accessible and cannot be the `fileid`

Error codes

Error Code Value	Cause	Suggested Solution
4097	Parameter is empty.	Check whether the API parameters in the code are correct.
4098	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Ensure that the SDK recording feature is used at the right time.
4100	Audio data is not captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	Mic permission is required for using the SDK. To add the permission, please see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
    ...
        else if (eventType == ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE) {
            int32 result = JsonObject->GetIntegerField(TEXT("result"));
            FString filepath = JsonObject->GetStringField(TEXT("file_path"));
            std::string path = TCHAR_TO_UTF8(filepath.operator*());
            int duration = 0;
            int filesize = 0;
            if (result == 0) {
```

```
                duration = ITMGContextGetInstance()->GetPTT()-
>GetVoiceFileDuration(path.c_str());
                filesize = ITMGContextGetInstance()->GetPTT()-
>GetFileSize(path.c_str());
            }
            onPttRecordFileCompleted(result, filepath, duration, filesize);
        }
    }
}
```

Pausing recording

This API is used to pause recording. If you want to resume recording, please call the `ResumeRecording` API.

API prototype

```
ITMGPTT virtual int PauseRecording()
```

Sample code

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

Resuming recording

This API is used to resume recording.

API prototype

```
ITMGPTT virtual int ResumeRecording()
```

Sample code

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

Canceling recording

This API is used to cancel recording. **There is no callback after cancellation**.

API prototype

```
ITMGPTT virtual int CancelRecording()
```

Sample code

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

Voice Message Upload, Download, and Playback

API	Description
UploadRecordedFile	Uploads the audio file
DownloadRecordedFile	Downloads the audio file
PlayRecordedFile	Plays back the audio file
StopPlayFile	Stops playing back the audio file
GetFileSize	Gets audio file size
GetVoiceFileDuration	Gets the audio file duration

Uploading an audio file

This API is used to upload an audio file.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of uploaded audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->UploadRecordedFile(filePath);
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description

result	int32	0: recording is completed
filepath	FString	Path of stored recording file
fileid	FString	File URL path

Error codes

Error Code Value	Cause	Suggested Solution
8193	An error occurred while accessing the file during upload.	Ensure the existence of the file and the validity of the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
8195	A network error occurred.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8200	No `appinfo` is set.	Check whether the `apply` API is called or whether the input parameters are empty.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
    ...
        else if (eventType == ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE) {
            int32 result = JsonObject->GetIntegerField(TEXT("result"));
            FString filepath = JsonObject->GetStringField(TEXT("file_path"));
            FString fileid = JsonObject->GetStringField(TEXT("file_id"));
            onPttUploadFileCompleted(result, filepath, fileid);
        }
    }
```

```
}
```

Downloading the audio file

This API is used to download an audio file.

API prototype

```
ITMGPTT virtual int DownloadRecordedFile(const char* fileId, const char*
filePath)
```

Parameter	Type	Description
fileId	const char*	URL path of file
filePath	const char*	Local path of saved file

Sample code

```
ITMGContextGetInstance()->GetPTT()->DownloadRecordedFile(fileID,filePath);
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	int32	0: recording is completed
filepath	FString	Path of stored recording file
fileid	FString	URL path of file, which will be retained on the server for 90 days

Error codes

Error Code Value	Cause	Suggested Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text

		feature is initialized.
12291	Network storage system exception.	The server failed to get the audio file. Check whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	Server file system error.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed during the process of getting the download parameters.	Check whether the device can access the internet.
12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
    ...
        else if (eventType == ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE) {
            int32 result = JsonObject->GetIntegerField(TEXT("result"));
            FString filepath = JsonObject->GetStringField(TEXT("file_path"));
            FString fileid = JsonObject->GetStringField(TEXT("file_id"));
            onPttDownloadFileCompleted(result, filepath, fileid);
        }
    }
}
```

Playing back audio

This API is used to play back audio.

API prototype

```
ITMGPTT virtual int PlayRecordedFile(const char* filePath)
ITMGPTT virtual int PlayRecordedFile(const char* filePath, nt voiceType)
```

Parameter	Type	Description
filePath	const char*	Local audio file path
voicetype	int	Voice changer type. For more information, see Voice Changing Effects.

Error codes

Error Code Value	Cause	Suggested Solution
20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
ITMGContextGetInstance()->GetPTT()->PlayRecordedFile(filePath);
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameter includes `result` and `file_path` .

Parameter	Type	Description
code	int	0: playback is completed
filepath	FString	Path of stored recording file

Error codes

Error Code Value	Cause	Suggested Solution
20481	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.

20483	Parameter is empty.	Check whether the API parameters in the code are correct.
20484	Internal error.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
    ...
        else if (eventType == ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE) {
            int32 result = JsonObject->GetIntegerField(TEXT("result"));
            FString filepath = JsonObject->GetStringField(TEXT("file_path"));
            onPttPlayFileCompleted(result, filepath);
        }
    }
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

API prototype

```
ITMGPTT virtual int StopPlayFile()
```

Sample code

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

Getting audio file size

This API is used to get the size of an audio file.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->GetFileSize(filePath);
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->GetVoiceFileDuration(filePath);
```

Fast Recording-to-Text Conversion

API	Description
SpeechToText	Converts speech to text

Converting audio file to text

This API is used to convert a specified audio file to text.

API prototype

```
ITMGPTT virtual void SpeechToText(const char* fileID)
```

Parameter	Type	Description
fileID	const char*	Audio file URL

Sample code

```
ITMGContextGetInstance()->GetPTT()->SpeechToText(fileID);
```

Translating audio file into text in specified language

This API can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation.

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

API prototype

```
ITMGPTT virtual int SpeechToText(const char* fileID,const char* speechLanguage)
ITMGPTT virtual int SpeechToText(const char* fileID,const char*
speechLanguage,const char* translateLanguage)
```

Parameter	Type	Description
fileID	const char*	The URL of the audio file, which will be retained on the server for 90 days.
speechLanguage	const char*	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translatelanguage	const char*	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

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

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result`, `file_path` and `text` (recognized text).

Parameter	Type	Description
result	int32	0: recording is completed
fileid	FString	URL of recording file, which will be retained on the server for 90 days

text	FString	Converted text

Error codes

Error Code Value	Cause	Suggested Solution
32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	Network failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	Incorrect speech-to-text conversion parameter.	Check whether the API parameter `fileid` in the code is empty.
32785	Speech-to-text translation returned an error.	Error with the backend of voice messaging and speech-to-text feature. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
    ...
        else if (eventType == ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE) {
```

```
            int32 result = JsonObject->GetIntegerField(TEXT("result"));
            FString text = JsonObject->GetStringField(TEXT("text"));
            FString fileid = JsonObject->GetStringField(TEXT("file_id"));
            onPttSpeech2TextCompleted(result, fileid, text);
        }
    }
}
```

Voice Message Volume Level APIs

API	Description
GetMicLevel	Gets real-time mic volume level
SetMicVolume	Sets recording volume level
GetMicVolume	Gets recording volume level
GetSpeakerLevel	Gets real-time speaker volume
SetSpeakerVolume	Sets playback volume level
GetSpeakerVolume	Gets playback volume level

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetMicLevel()
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetMicLevel();
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT virtual int SetMicVolume(int vol)
```

Sample code

```
ITMGContextGetInstance()->GetPTT()->SetMicVolume(100);
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetMicVolume()
```

Sample code

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

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetSpeakerLevel()
```

Sample code

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

Setting the playback volume of voice message

This API is used to set the playback volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT virtual int SetSpeakerVolume(int vol)
```

Sample code

```
ITMGContextGetInstance()->GetPTT()->SetSpeakerVolume(100);
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetSpeakerVolume()
```

Sample code

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

Advanced APIs

Getting version number

This API is used to get the SDK version number for SDK usage analysis.

API prototype

```
ITMGContext virtual const char* GetSDKVersion()
```

Sample code

```
ITMGContextGetInstance()->GetSDKVersion();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext int SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	

		Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

`ITMG_LOG_LEVEL` description:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContextGetInstance()->SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

OS	Path
Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents
Android	/sdcard/Android/data/xxx.xxx.xxx/files
Mac	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext virtual int SetLogPath(const char* logDir)
```

Parameter	Type	Description
logDir	const char*	Path

Sample code

```
cosnt char* logDir = ""// Set a path by yourself
ITMGContext* context = ITMGContextGetInstance();
context->SetLogPath(logDir);
```

Callback Messages

Message	Description	Parameter
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room	result; error
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected for network or other reasons	result; error
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Room members were updated	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	The reconnection to the room started	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	The reconnection to the room succeeded	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status was	result; error_info;

	changed	sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker device was changed	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A new speaker device was added	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker device was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A new mic device was added	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic device was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic device was changed	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	The room quality changed	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Voice message recording was completed	result; file_

ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Voice message upload was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Voice message download was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Voice message playback was completed	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast speech-to-text conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion was completed	result; file_ text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	Streaming speech-to-text conversion is in progress	result; file_ text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed	result; text;file_id

Voice Chat

Last updated：2024-01-18 15:02:24

This document describes how to access and debug GME client APIs for the voice chat feature for Unreal Engine.

Key Considerations for Using GME

GME provides the real-time voice, voice message, and speech-to-text services, which all depend on core APIs such as `Init` and `Poll`.

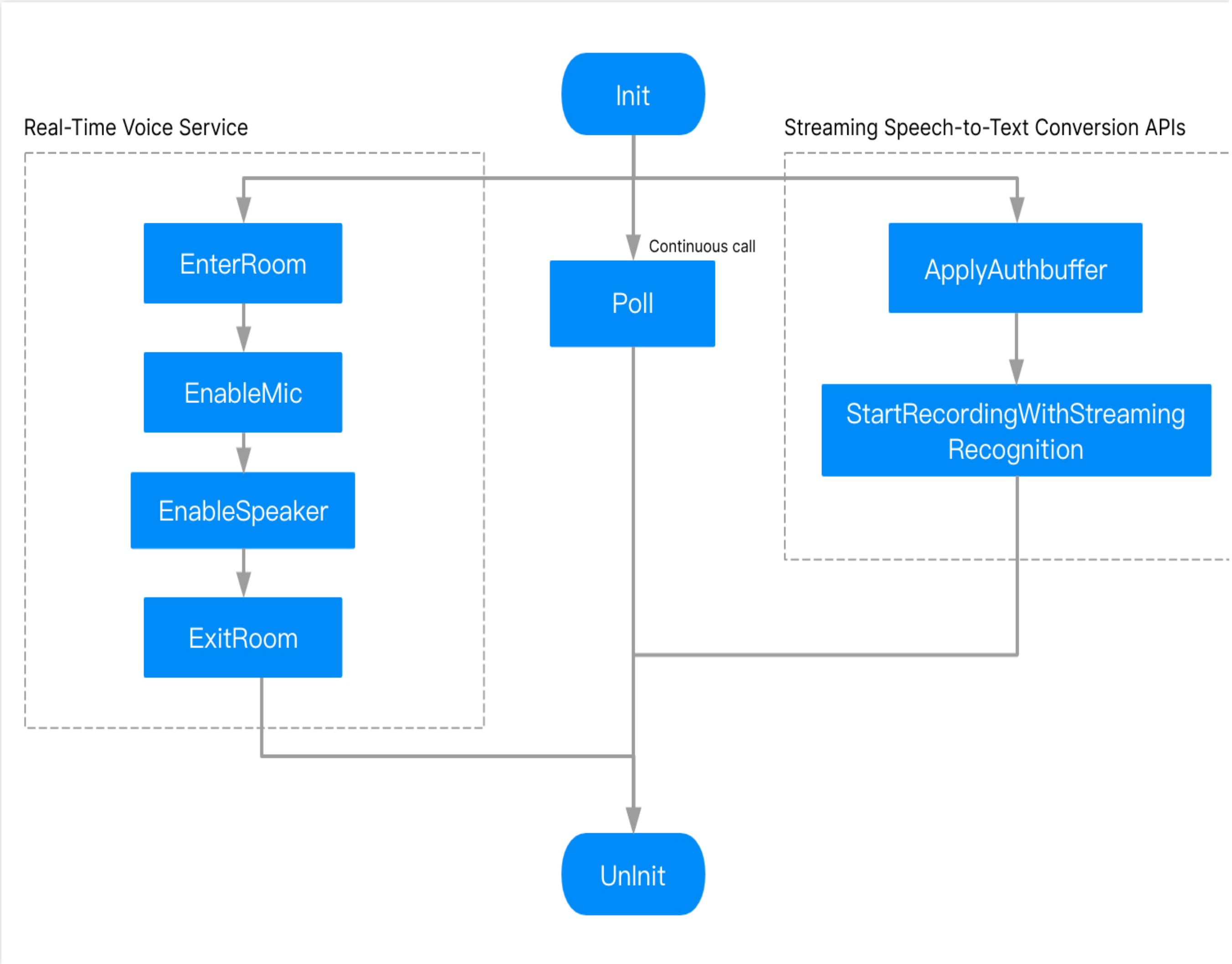
Key notes

You have created a GME application and obtained the `AppID` and `Key` of the SDK as instructed in Activating Services.
You have **activated the real-time voice, voice message, and speech-to-text services of GME** as instructed in Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error code, please see Error Codes.

Connecting to the SDK

Directions

Key processes involved in SDK connection are as follows:



1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Entering a voice chat room, API: EnterRoom
4. Enabling the microphone, API: EnableMic
5. Enabling the speaker, API: EnableSpeaker
6. Exiting a voice room, API: ExitRoom
7. Uninitializing GME, API: UnInit

C++ classes

Class	Description
ITMGContext	Key APIs
ITMGRoom	Room APIs
ITMGRoomManager	Room management APIs

ITMGAudioCtrl	Audio APIs
ITMGAudioEffectCtrl	Sound effect and accompaniment APIs

Key APIs

API	Description
Init	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME

Importing the header file

You need to import the header file `tmg_sdk.h` first before you can access GME. The classes in the header file inherit `ITMGDelegate` for message delivery and callback.

Sample code

```
#include "tmg_sdk.h"

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

Setting a singleton

You need to get `ITMGContext` first before you can call the `EnterRoom` function. All calls begin with `ITMGContext` , which is returned to the application through the `ITMGDelegate` callback and must be set first.

Sample code

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

Message delivery

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
}
```

Initializing the 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

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

Parameter	Type	Description
sdkAppId	const char*	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	const char*	`openID` can only be in `Int64` type, which is passed in after being converted to a `const char*`. 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.

Returned values

Returned Value	Description
AV_OK = 0	Initialized SDK successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

Notes on 7015 error code

The 7015 error code is judged by md5. If this error is reported during integration, please check the integrity and version of the SDK file as prompted.
The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.
Due to the third-party reinforcement, Unity packaging mechanism and other factors, the md5 of the library file will be affected, resulting in misjudgment. **Please ignore this error in the logic for official release**, and try to avoid displaying it in the UI.

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());
```

Triggering an event callback

Event callbacks can be triggered by periodically calling the `Poll` API in `update`. `Poll` is the message pump of GME, and the `Poll` API should be called periodically for GME to trigger event callbacks; otherwise, the entire SDK service will run exceptionally.
For details, please see UEDemoLevelScriptActor.cpp file in Demo.

Calling the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
class ITMGContext {
protected:
    virtual ~ITMGContext() {}
```

```
public:
    virtual void Poll()= 0;
}
```

Sample code

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

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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext int Resume()
```

Uninitializing the SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

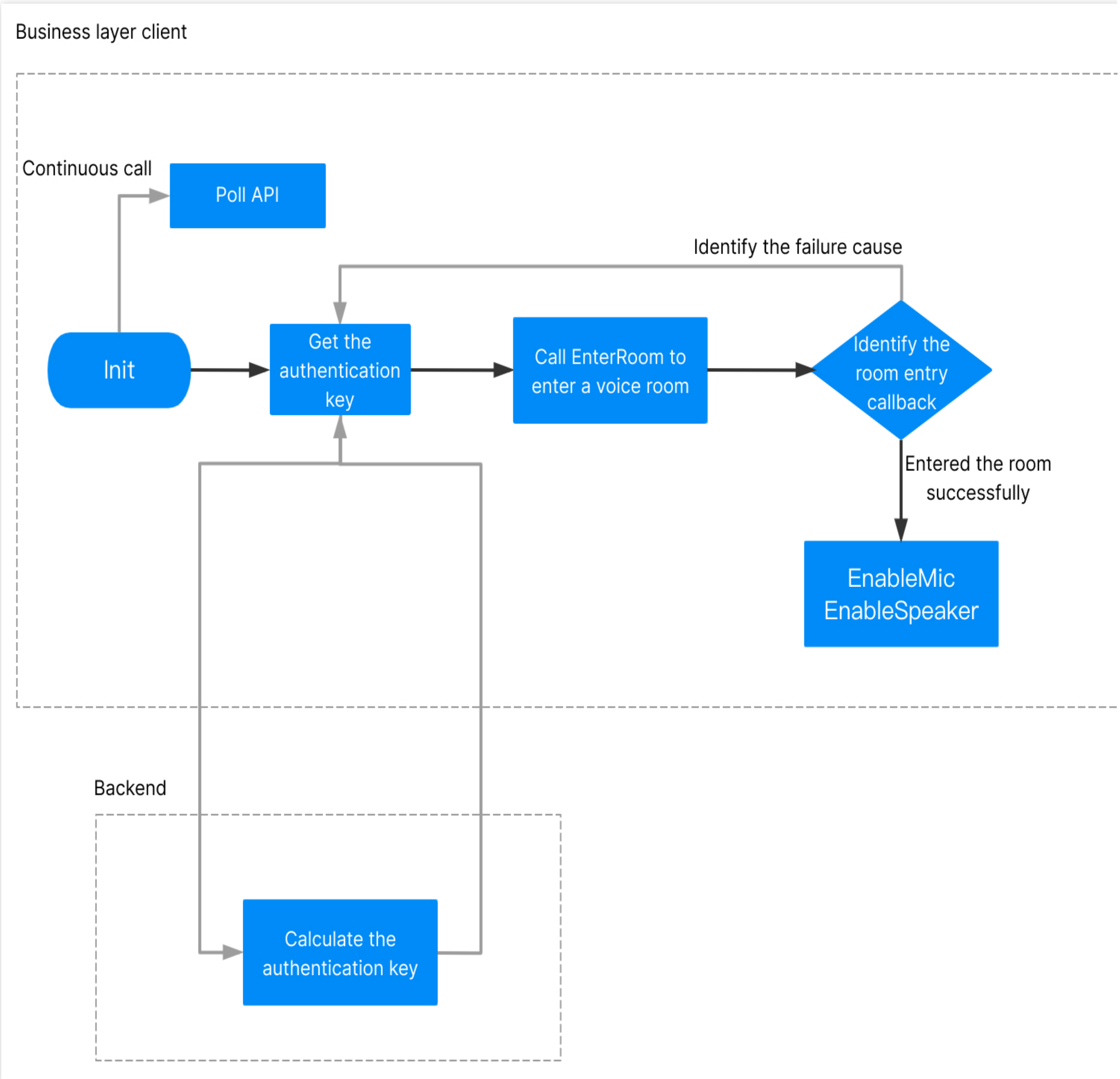
API prototype

```
ITMGContext int Uninit()
```

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, please see FAQs About Voice Chat.



API	Description
GenAuthBuffer	Calculates the local authentication key
EnterRoom	Enters a room

ExitRoom	Exits the room
IsRoomEntered	Determines whether room entry is successful
SwitchRoom	Switches the room quickly

Local authentication key calculation

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

```
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	unsigned int	`AppId` from the Tencent Cloud console
strRoomID	const char*	Room ID, which can contain up to 127 characters.
strOpenID	const char*	User ID, which is the same as `openID` during initialization.
strKey	const char*	Permission key from the Tencent Cloud console
strAuthBuffer	const char*	Returned `authbuff`
bufferLength	int	The length of the returned `authbuff`. `500` is recommended.

Sample code

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

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.

Note：

If the room entry callback result is `0` , the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.
The audio type of the room is determined by the first user entering the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will change to the smooth sound quality. Only after a member in the room calls the `ChangeRoomType` API will the audio type of the room be changed.

API prototype

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

Parameter	Type	Description
roomID	const char*	Room ID, which can contain up to 127 characters.
roomType	ITMG_ROOM_TYPE	Room type. We recommend you select `ITMG_ROOM_TYPE_FLUENCY` for games. For more information on room audio types, see Sound Quality.
authBuffer	const 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, 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

```
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_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);
    }
}
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnected callback prompt `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code Value	Cause and Suggested Solution
7006	Authentication failed: 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.

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API during API call; instead, you can directly call the `EnterRoom` API.

API prototype

```
ITMGContext virtual int ExitRoom()
```

Sample code

```
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` .

Sample code

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

```
            case ITMG_MAIN_EVENT_TYPE_EXIT_ROOM:
        {
        // Process
        break;
        }
    }
}
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"","result":0}

Determining whether a user has entered a room

This API is used to determine whether the user has entered a room. A value in bool type will be returned. Do not call this API during room entry.

API prototype

```
ITMGContext virtual bool IsRoomEntered()
```

Sample code

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

Switching a room

User can call this API to quickly switch the voice chat room after entering the room. After the room is switched, the device is not reset, that is, if the microphone is already enabled in this room, the microphone will keep enabled after the room is switched.

The callback for quickly switching rooms is `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM` , and the fields are `error_info` and `result` .

API prototype

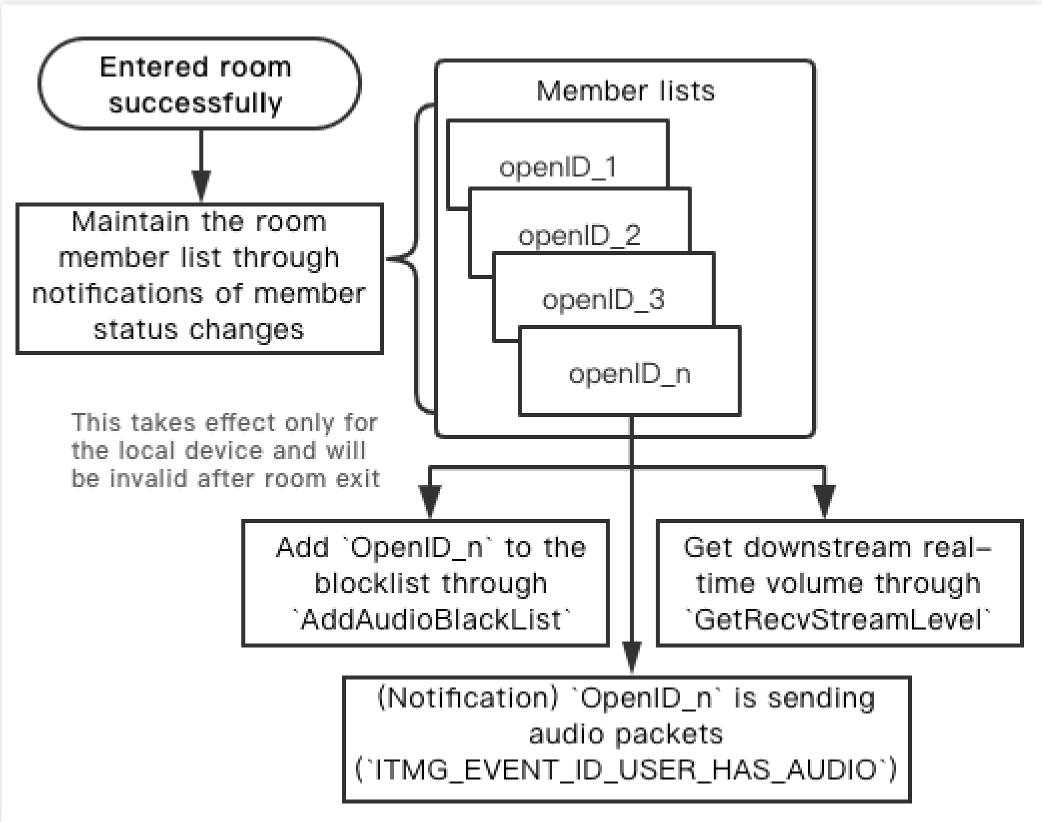
```
ITMGContext virtual int SwitchRoom(const char* targetRoomID, const char*
authBuff, int buffLen);
```

Type descriptions

Parameter	Type	Description
targetRoomID	const char*	ID of the room to enter
authBuffer	const char*	Generates a new authentication key with the ID of the room to enter
buffLen	int	Authentication key length

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed
AddAudioBlackList	Mutes a member in the room
RemoveAudioBlackList	Unmutes a user

Notification events of member room entry and speaking status

This API is used to obtain the user speaking in the room and display it in the UI, and to send a notification when someone entering or exiting the room.

Notification for this event will be sent only when the status changes. To get member status in real time, cache the notification when it is received at a higher layer. The event message is `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` , where the data contains `event_id` and `user_list` . The event message will be identified in the `OnEvent` function.

Notifications for audio events are subject to a threshold, and a notification will be sent only when this threshold is exceeded. The notification "A member has stopped sending audio packets" will be sent if no audio packets are received in more than two seconds.

event_id	Description	Maintenance
ITMG_EVENT_ID_USER_ENTER	Return the `openid` of the member entering the room.	Member list
ITMG_EVENT_ID_USER_EXIT	Return the `openid` of the member exiting the room.	Member list
ITMG_EVENT_ID_USER_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
ITMG_EVENT_ID_USER_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
            case ITMG_MAIN_EVNET_TYPE_USER_UPDATE:
        {
        // Process
        // Parse the parameter to get `eventID` and `user_list`
            switch (eventID)
             {
             case ITMG_EVENT_ID_USER_ENTER:
                  // A member enters the room
                  break;
             case ITMG_EVENT_ID_USER_EXIT:
                  // A member exits the room
```

```
                    break;
                case ITMG_EVENT_ID_USER_HAS_AUDIO:
                    // A member sends audio packets
                    break;
                case ITMG_EVENT_ID_USER_NO_AUDIO:
                    // A member stops sending audio packets
                    break;
                 default:
                    break;
                }
            break;
            }
        }
}
```

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B;

If B blocks neither A nor C, B can hear both of them;

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
ITMGContext ITMGAudioCtrl int AddAudioBlackList(const char* openId)
```

Parameter	Type	Description
openId	char*	`openid` of the user to be blocked

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->AddAudioBlackList(openId);
```

Unmuting

This API is used to remove an ID from the audio data blacklist. A returned value of 0 indicates the call is successful.

API prototype

```
ITMGContext ITMGAudioCtrl int RemoveAudioBlackList(const char* openId)
```

Parameter	Type	Description
openId	char*	ID to be unblocked

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->RemoveAudioBlackList(openId);
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.
When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend you call the `EnableMic` or `EnableSpeaker` API.
To enable the user to press the mic button on the UI to speak and release it to stop speaking, we recommend you call `EnableAudioCaptureDevice` once during room entry and call `EnableAudioSend` to enable the user to speak while pressing the button.

API	Description
EnableMic	Enables/Disables the mic
GetMicState	Gets the mic status
EnableAudioCaptureDevice	Enables/Disables the capturing device
IsAudioCaptureDeviceEnabled	Gets the capturing device status
EnableAudioSend	Enables/Disables audio upstreaming
IsAudioSendEnabled	Gets the audio upstreaming status
GetMicLevel	Gets the real-time mic volume level
GetSendStreamLevel	Gets real-time audio upstreaming volume
SetMicVolume	Sets the mic volume level
GetMicVolume	Gets the mic volume level

Enabling or disabling mic

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

EnableMic = EnableAudioCaptureDevice + EnableAudioSend

API prototype

```
ITMGAudioCtrl virtual int EnableMic(bool bEnabled)
```

Parameter	Type	Description
bEnabled	bool	To enable the mic, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
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_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);
    }
}
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
ITMGAudioCtrl virtual int GetMicState()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicState();
```

Enabling or disabling the capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.
This API can only be called after room entry. The device will be disabled automatically after room exit.
Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioCaptureDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To enable the capturing device, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable capturing device
ITMGContextGetInstance()->GetAudioCtrl()->EnableAudioCaptureDevice(true);
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
ITMGContext virtual bool IsAudioCaptureDeviceEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioCaptureDeviceEnabled();
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain mute. For more information on how to enable/disable the capturing device, please see the `EnableAudioCaptureDevice` API.

API prototype

```
ITMGContext  virtual int EnableAudioSend(bool bEnable)
```

Parameter	Type	Description
bEnable	bool	To enable audio upstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

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

Getting the audio upstreaming status

This API is used to get the status of audio upstreaming.

API prototype

```
ITMGContext virtual bool IsAudioSendEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioSendEnabled();
```

Getting the real-time mic volume

This API is used to get the real-time mic volume. An int-type value in the range of 0-100 will be returned. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl virtual int GetMicLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicLevel();
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume. An int-type value in the range of 0-100 will be returned.

API prototype

```
ITMGAudioCtrl virtual int GetSendStreamLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSendStreamLevel();
```

Setting the mic software volume

This API is used to set the mic volume level. The corresponding parameter is `volume` , which is equivalent to attenuating or gaining the captured sound.

API prototype

```
ITMGAudioCtrl virtual int SetMicVolume(int vol)
```

Parameter	Type	Description
vol	int	Value range: 0–200. Default value: 100. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
int micVol = (int)(value * 100);
ITMGContextGetInstance()->GetAudioCtrl()->SetMicVolume(vol);
```

Getting the mic software volume

This API is used to obtain the microphone volume. An "int" value is returned. Value 101 represents API SetMicVolume has not been called.

API prototype

```
ITMGAudioCtrl virtual int GetMicVolume()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicVolume();
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker
GetSpeakerState	Gets the speaker status
EnableAudioPlayDevice	Enables/Disables the playback device
IsAudioPlayDeviceEnabled	Gets playback device status
EnableAudioRecv	Enables/Disables audio downstreaming
IsAudioRecvEnabled	Gets the audio downstreaming status
GetSpeakerLevel	Gets the real-time speaker volume level
GetRecvStreamLevel	Gets the real-time downstreaming audio volume levels of other members in the room
SetSpeakerVolume	Sets the speaker volume level
GetSpeakerVolume	Gets the speaker volume level

Enabling or disabling speaker

This API is used to enable/disable the speaker. **EnableSpeaker = EnableAudioPlayDevice + EnableAudioRecv**

API prototype

```
ITMGAudioCtrl virtual int EnableSpeaker(bool enable)
```

Parameter	Type	Description
enable	bool	To disable the speaker, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

```
// Enable the speaker
ITMGContextGetInstance()->GetAudioCtrl()->EnableSpeaker(true);
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerState()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerState();
```

Enabling or disabling the playback device

This API is used to enable/disable a playback device.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioPlayDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To disable the playback device, set this parameter to `false`; otherwise, set it to `true`.

Sample code

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

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
ITMGAudioCtrl virtual bool IsAudioPlayDeviceEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioPlayDeviceEnabled();
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain mute. For more information on how to enable/disable the playback device, please see the `EnableAudioPlayDevice` API.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioRecv(bool enable)
```

Parameter	Type	Description
enable	bool	To enable audio downstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

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

Getting the audio downstreaming status

This API is used to get the status of audio downstreaming.

API prototype

```
ITMGAudioCtrl virtual bool IsAudioRecvEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioRecvEnabled();
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume. An int-type value will be returned to indicate the volume. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerLevel();
```

Getting the real-time downstreaming audio levels of other members in the room

This API is used to get the real-time audio downstreaming volume of other members in the room. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGAudioCtrl virtual int GetRecvStreamLevel(const char* openId)
```

Parameter	Type	Description
openId	char*	`openId` of other members in the room

Sample code

```
iter->second.level = ITMGContextGetInstance()->GetAudioCtrl()-
>GetRecvStreamLevel(iter->second.openid.c_str());
```

Setting the speaker volume

This API is used to set the speaker volume.

API prototype

```
ITMGAudioCtrl virtual int SetSpeakerVolume(int vol)
```

Parameter	Type	Description
vol	int	Volume level. Value range: 0–200. Default value: 100. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
int vol = 100;
ITMGContextGetInstance()->GetAudioCtrl()->SetSpeakerVolume(vol);
```

Getting the speaker volume

This API is used to get the speaker volume. An int-type value will be returned to indicate the volume. 101 indicates that the `SetSpeakerVolume` API has not been called.
"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerVolume()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerVolume();
```

Device Selection APIs

Device selection APIs can be used only on PC.

API	Description
GetMicListCount	Gets the number of mics
GetMicList	Lists mics
GetSpeakerListCount	Gets the number of speakers
GetSpeakerList	Lists speakers
SelectMic	Selects mics
SelectSpeaker	Selects speakers

Getting the number of mics

This API is used to get the number of mics.

Function prototype

```
ITMGAudioCtrl virtual int GetMicListCount()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicListCount();
```

Enumerating mics

This API is used together with the `GetMicListCount` API to enumerate mics.

Function prototype

```
ITMGAudioCtrl virtual int GetMicList(TMGAudioDeviceInfo* ppDeviceInfoList, int
nCount)

class TMGAudioDeviceInfo
{
public:
    const char* pDeviceID;
    const char* pDeviceName;
};
```

Parameter	Type	Description
ppDeviceInfoList	TMGAudioDeviceInfo	Device list
nCount	int	Number of the mics

`TMGAudioDeviceInfo` Parameter	Type	Description
pDeviceID	const char*	Device ID
pDeviceName	const char*	Device name

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicList(ppDeviceInfoList,nCount);
```

Selecting a mic

This API is used to select a mic. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default mic will be selected.

The 0th device id returned in the GetMicList API is the default device of the call device. If there is a selected call device, it will be maintained by service. If it is unplugged, the call device will be changed back into the default device.

Function prototype

```
ITMGAudioCtrl virtual int SelectMic(const char* pMicID)
```

Parameter	Type	Description
pMicID	const char*	Mic ID, which is from the list returned by `GetMicList` .

Sample code

```
const char* pMicID ="{0.0.1.00000000}.{7b0b712d-3b46-4f7a-bb83-bf9be4047f0d}";
ITMGContextGetInstance()->GetAudioCtrl()->SelectMic(pMicID);
```

Getting the number of speakers

This API is used to get the number of speakers.

Function prototype

```
ITMGAudioCtrl virtual int GetSpeakerListCount()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerListCount();
```

Enumerating speakers

This API is used together with the `GetSpeakerListCount` API to enumerate speakers.

Function prototype

```
ITMGAudioCtrl virtual int GetSpeakerList(TMGAudioDeviceInfo* ppDeviceInfoList,
int nCount)
```

Parameter	Type	Description
ppDeviceInfoList	TMGAudioDeviceInfo	Device list
nCount	int	Number of the speakers

`TMGAudioDeviceInfo` Parameter	Type	Description
pDeviceID	const char*	Device ID
pDeviceName	const char*	Device name

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()-
>GetSpeakerList(ppDeviceInfoList,nCount);
```

Selecting a speaker

This API is used to select a playback device. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default playback device will be selected.

Function prototype

```
ITMGAudioCtrl virtual int SelectSpeaker(const char* pSpeakerID)
```

Parameter	Type	Description
pSpeakerID	const char*	Speaker ID, which is from the list returned by `GetSpeakerList` .

Sample code

```
const char* pSpeakerID ="{0.0.1.00000000}.{7b0b712d-3b46-4f7a-bb83-
bf9be4047f0d}";
ITMGContextGetInstance()->GetAudioCtrl()->SelectSpeaker(pSpeakerID);
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

API prototype


```
ITMGAudioCtrl virtual int EnableLoopBack(bool enable)
```

Parameter	Type	Description
enable	bool	Specifies whether to enable

Sample code

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

Getting a user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, please see the `EnterRoom` API.

API prototype

```
class ITMGRoom {
public:
    virtual ~ITMGRoom() {} ;
    virtual int GetRoomType() = 0;

};
```

Sample code

```
ITMGContext* context = ITMGContextGetInstance();
ITMGContextGetInstance()->GetRoom()->GetRoomType();
```

Changing the room type

This API is used to modify a user's room audio type. For the result, please see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` . The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

API prototype

```
IITMGContext TMGRoom public int ChangeRoomType((ITMG_ROOM_TYPE roomType)
```

Parameter	Type	Description

roomType	ITMG_ROOM_TYPE	Room type to be switched to the target type. For room audio types, please see the `EnterRoom` API.

Sample code

```
ITMGContext* context = ITMGContextGetInstance();
ITMGContextGetInstance()->GetRoom()->ChangeRoomType(ITMG_ROOM_TYPE_FLUENCY);
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result; error_info; new_room_type	{"error_info":"","new_room_type":0

Callback for room type setting completion

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result` , `error_info` , and `new_room_type` . The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description
ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	Indicates that the existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	Indicates that a user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).
ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	Indicates that a user is already in the room and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	Indicates that a room member calls the `ChangeRoomType` API to request a change of room audio type.

Sample code

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

```
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE
== type)
        {
        // Process room type events
      }
}
```

The monitoring event of room call quality

This is the quality monitoring event, which is triggered once every two seconds after room entry, and its message is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which are as detailed below:

Parameter	Type	Description
weight	int	Value range: 1-50. 50 indicates excellent sound quality, 1 indicates very poor (barely usable) sound quality, and 0 represents an initial meaningless value. Generally, if the value is below 30, the business layer will remind users that the network is poor and recommend them to switch the network.
loss	double	Upstream packet loss rate
delay	int	Voice chat delay in ms

Getting the version number

This API is used to get the SDK version number for analysis.

API prototype

```
ITMGContext virtual const char* GetSDKVersion()
```

Sample code

```
ITMGContextGetInstance()->GetSDKVersion();
```

Setting the log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext int SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL
levelPrint)
```


Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

`ITMG_LOG_LEVEL` is as detailed below:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContextGetInstance()->SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

OS	Path
Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents
Android	/sdcard/Android/data/xxx.xxx.xxx/files
Mac	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext virtual int SetLogPath(const char* logDir)
```


Parameter	Type	Description
logDir	const char*	Path

Sample code

```
cosnt char* logDir = ""// Set a path by yourself

ITMGContext* context = ITMGContextGetInstance();
context->SetLogPath(logDir);
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

API prototype

```
ITMGRoom virtual const char* GetQualityTips()
```

Sample code

```
ITMGContextGetInstance()->GetRoom()->GetQualityTips();
```

Callback Messages

Message	Description	Parameter
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room	result; error
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was	result; error

	disconnected for network or other reasons	
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The room members were updated	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	Room reconnection started	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	Room reconnection succeeded	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status changed	result; error_info; sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker was changed	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A speaker was added	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker was lost	result; error

ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A mic was added	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic was changed	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	Room quality message	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Recording of a voice message was completed	result; file_p
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Upload of a voice message was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Download of a voice message was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Playback of a voice message was completed	result; file_p
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast recording-to-text conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming	result; file_p

	speech-to-text conversion was completed	text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	A voice message is being converted into text in a streaming manner	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed	result; text;file_id

Cocos2D SDK

Project Configuration

Last updated：2024-01-18 15:11:45

This document describes how to configure a Cocos2d project for the GME APIs for Cocos2d.

SDK Preparation

1. Download the applicable demo and SDK. For more information, see SDK Download Guide.
2. Decompress the obtained SDK resources.
3. The folder contains:

GMESDK: GME SDK framework file.

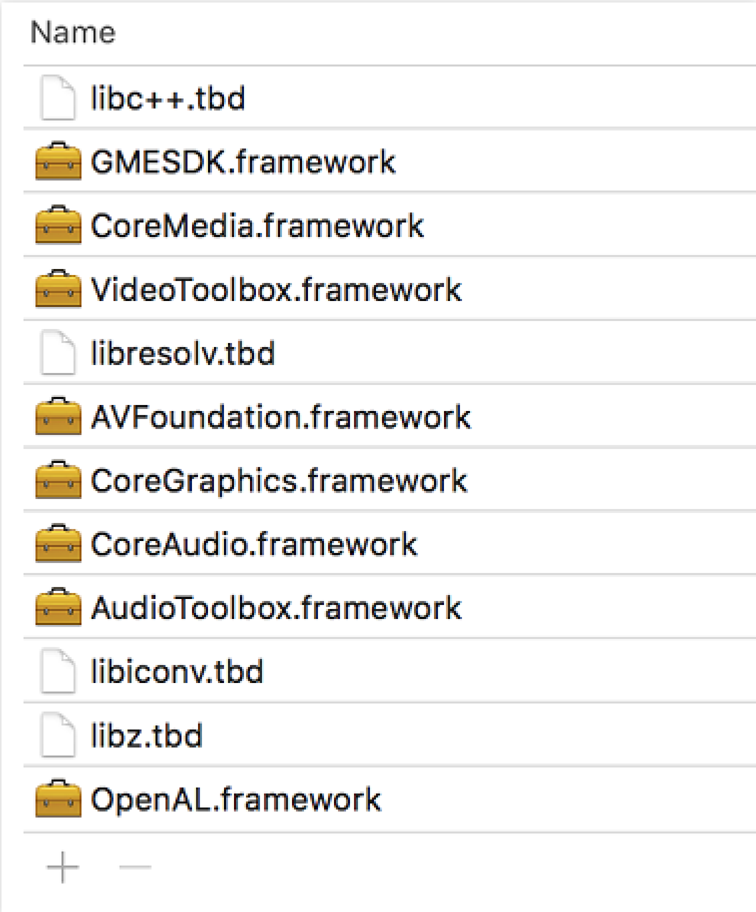
GMECocosDemo: GME SDK demo project.

Note:

The SDK supports compilation on macOS.

iOS Xcode Configuration

1. Add the framework to the Xcode project and set the header file import location (the framework file in the `GMESDK` folder must be added to the project).
2. Add dependent libraries as shown below:



Android Configuration

1. Add `gmesdk.jar` to the libs library.
2. Import the `so` file into `Activity` as shown below:

```
public class AppActivity extends Cocos2dxActivity {
 static final String TAG = "AppActivity";
 static OpensdkGameWrapper gameWrapper ;
 static {
      OpensdkGameWrapper.loadSdkLibrary();
 }
}
```

3. Initialize in the `oncreate` function exactly in the following sequence:

```
protected void onCreate(Bundle savedInstanceState) {
      super.setEnableVirtualButton(false);
      super.onCreate(savedInstanceState);
      // Initialize exactly in the following sequence
      gameWrapper = new OpensdkGameWrapper(this);
```

```
        runOnGLThread(new Runnable() {
            @Override
            public void run() {
                gameWrapper.initOpensdk();
            }
        });
  }
```

4. Configure your project for compilation options by referring to the `Android.mk` in the GME Demo for Cocos.
Path: GMECocos/GMECocosDemo/proj.android-studio/app/jni/Android.mk
Path to the `preBuild.mk` file: /Users/username/Downloads/GMECocos/GMESDK/android/bin/preBuild.mk

Exporting to Different Platforms

Project configuration is required before you can export executables from the Cocos2d engine for different platforms:

Android

Configuring required permissions

Add the following permissions in the `AndroidManifest.xml` file of the project:

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

To use the voice messaging and speech-to-text feature, add the following under the `application` node in the manifest file:

```
<application android:usesCleartextTraffic="true" >
```

Adding permissions as needed

Add the following permissions in the `AndroidManifest.xml` file of the project as needed:
Read/Write permission
Bluetooth permission
The read/write permission is not required. Determine whether to add it according to the following rules:
If you use the default log path (/SDCARD/Android/Data/files), it means that you do not call SetLogPath, and do not need Write_External_Storage permission.
If you call the setLogPath API to set the log path to an external storage device, and the storage path of the voice message recording is an external storage device, you need to apply for the Write_External_Storage permission to the

user and get the user's approval.

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Add the Bluetooth permission according to the following rules:

If `targetSDKVersion` in the project is v30 or earlier:

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
```

If `targetSDKVersion` in the project is v31 or later:

```
<uses-permission android:name="android.permission.BLUETOOTH"
android:maxSdkVersion="30" />
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
```

iOS

Add permissions:

Microphone Usage Description: Allows access to microphone.

Grant the `Allow Arbitrary Loads` permission as shown below:

Key	Type	Value
▼ Information Property List	Dictionary	(19 items)
Localization native development re...	String	$(DEVELOPMENT_LANGUAGE)
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Application requires iPhone enviro...	Boolean	YES
▼ App Transport Security Settings	Dictionary	(1 item)
Allow Arbitrary Loads	Boolean	YES
Privacy - Camera Usage Description	String	Privacy - Camera Usage Description
Privacy - Microphone Usage Desc...	String	NSMicrophoneUsageDescription
▶ Required background modes	Array	(1 item)
Application supports iTunes file sh...	Boolean	YES
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
▶ Required device capabilities	Array	(1 item)
▶ Supported interface orientations	Array	(3 items)
▶ Supported interface orientations (i...	Array	(4 items)

Windows

You need to download the SDK for Windows as instructed in SDK Download Guide and import it into the project.

Getting Started

Last updated：2024-01-18 15:11:45

This document describes how to get started with the GME APIs for Cocos2d.

This document only describes the most important APIs to help you get started with GME. For more information on APIs, please see the API documentation.

Important API	Description
Init	Initializes GME
Poll	Triggers event callback
EnterRoom	Enters room
EnableMic	Enables mic
EnableSpeaker	Enables speaker

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `AV_OK` will be returned with the value being 0.

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For GME callback messages, please see the callback message list.

Operation on devices should be performed after successful room entry.

For detailed error codes, please see Error Codes.

Directions for Quick Access

1. Get a singleton

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

Sample code

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

2. Initialize the SDK

For more information on how to get parameters, please see Access Guide.
This API requires the `AppID` from the Tencent Cloud Console and the `openID` as parameters. The `openID` uniquely identifies a user with the rules stipulated by the application developer and must be unique in the application (currently, only INT64 is supported).

Note:

The SDK must be initialized so that a room can be entered

Function prototype

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

Parameter	Type	Description
sdkAppId	char*	`AppId` from the Tencent Cloud Console
openId	char*	`OpenID` can only be in Int64 type (converted to char*) with a value greater than 10,000, which is used to identify the user

Sample code

```
#define SDKAPPID3RD "1400089356"
cosnt char* openId="10001";
ITMGContext* context = ITMGContextGetInstance();
context->Init(SDKAPPID3RD, openId);
```

3. Trigger event callback

Event callbacks can be triggered by periodically calling the `Poll` API in `update` .

Function prototype

```
class ITMGContext {
protected:
    virtual ~ITMGContext() {}

public:
    virtual void Poll()= 0;
}
```

Sample code

```
ITMGContextGetInstance()->Poll();
```

4. Authenticate

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.To get authentication for voice messaging 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	`AppId` from the Tencent Cloud Console.
strRoomID	char*	Room ID, which can contain up to 127 characters (for voice messaging and speech-to-text feature, enter `null`).
strOpenID	char*	User ID, which is the same as `openID` during initialization.
strKey	char*	Permission key from the Tencent Cloud Console.
strAuthBuffer	char*	Returned `authbuff` .
bufferLength	int	Length of the `authbuff` 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);
```

5. Enter a room

When a client enters a room with the generated authentication information, the `ITMG_MAIN_EVENT_TYPE_ENTER_ROOM` message will be received as a callback. Mic and speaker are not enabled by default after room entry. The returned value of `AV_OK` indicates a success.

For entering a common voice chat room that does not involve range voice, use the common room entry API. For more information, please see Range Voice.

Function prototype

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

For more information on how to choose a room audio type, please see Sound Quality Selection.

Sample code

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

6. Receive callback for room entry

After the client enters the room, the message `ITMG_MAIN_EVENT_TYPE_ENTER_ROOM` will be sent and identified in the `OnEvent` function.

Sample code

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

7. Enable/Disable the mic

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

Function prototype

```
ITMGAudioCtrl virtual int EnableMic(bool bEnabled)
```

Parameter	Type	Description
bEnabled	bool	To enable the mic, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

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

8. Enable/Disable the speaker

This API is used to enable/disable the speaker.

Function prototype

```
ITMGAudioCtrl virtual int EnableSpeaker(bool enable )
```

Parameter	Type	Description
enable	bool	To disable the speaker, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

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


Voice Chat

Last updated：2024-01-18 15:11:45

This document describes how to access and debug GME client APIs for the voice chat feature for Cocos2d.

Key Considerations for Using GME

GME provides the real-time voice, voice message, and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

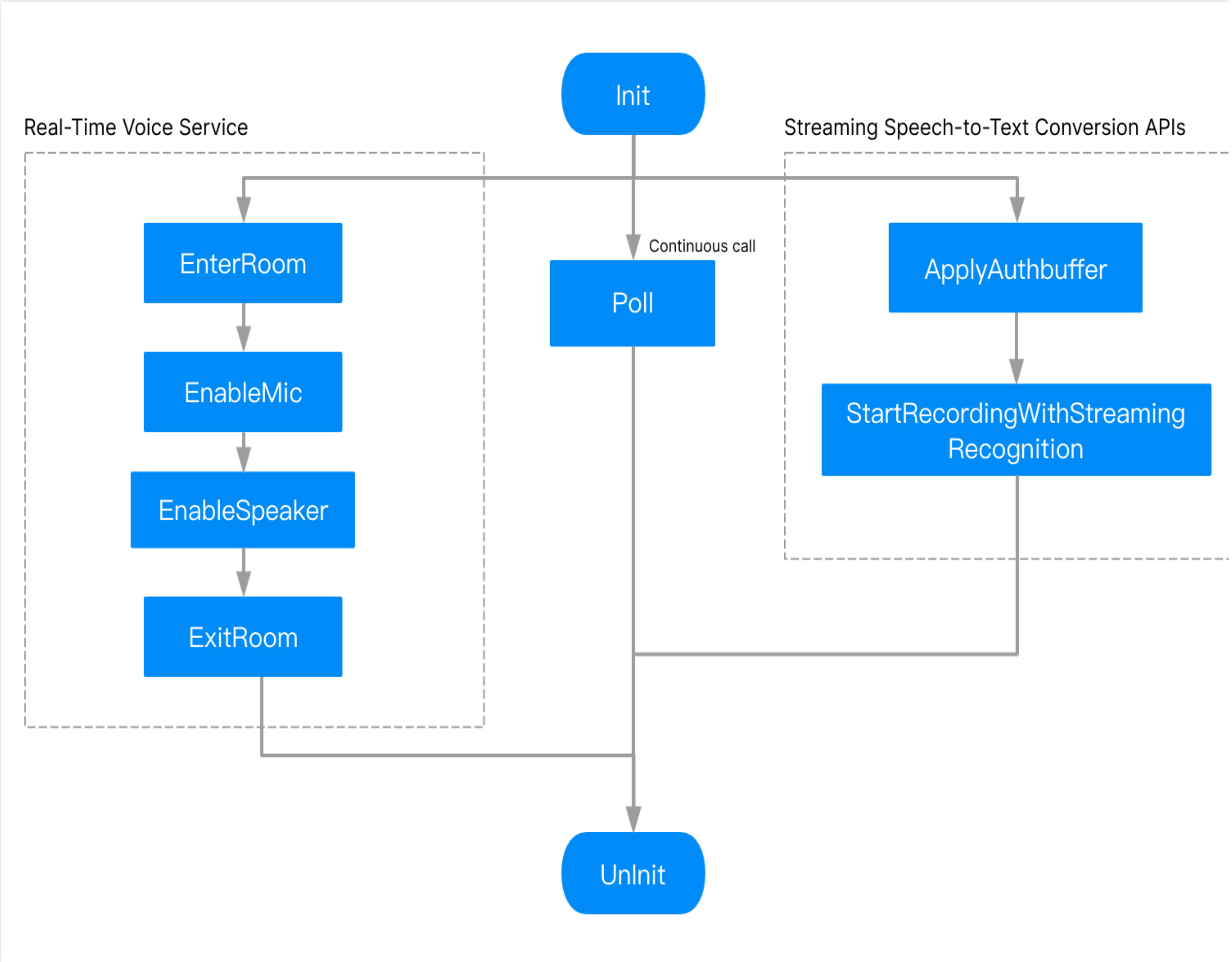
Key notes

You have created a GME application and obtained the `AppID` and `Key` of the SDK as instructed in Activating Services.
You have **activated the real-time voice, voice message, and speech-to-text services of GME** as instructed in Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error code, please see Error Codes.

Connecting to the SDK

Directions

Key processes involved in SDK connection are as follows:



1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Entering a voice chat room, API: EnterRoom
4. Enabling the microphone, API: EnableMic
5. Enabling the speaker, API: EnableSpeaker
6. Exiting a voice room, API: ExitRoom
7. Uninitializing GME, API: UnInit

C++ classes

Class	Description
ITMGContext	Key APIs
ITMGRoom	Room APIs
ITMGRoomManager	Room management APIs

ITMGAudioCtrl	Audio APIs
ITMGAudioEffectCtrl	Sound effect and accompaniment APIs

Key APIs

API	Description
Init	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME

Preparations

You need to import the header file `tmg_sdk.h` first before you can access GME. The classes in the header file inherit `ITMGDelegate` for message delivery and callback.

Sample code

```
#include "tmg_sdk.h"

class TMGTestScene : public cocos2d::Scene,public ITMGDelegate
{
public:
...
private:
...
}
```

Setting a singleton

You need to get `ITMGContext` first before you can call the `EnterRoom` function. All calls begin with `ITMGContext`, which is returned to the application through the `ITMGDelegate` callback and must be set first.

Sample code

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

```
context->SetTMGDelegate(this);
```

Message delivery

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:
//TMGTestScene.h:
class TMGTestScene : public cocos2d::Scene,public ITMGDelegate
{
public:
    void OnEvent(ITMG_MAIN_EVENT_TYPE eventType, const char* data);
}

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

Initializing the 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

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

Parameter	Type	Description
sdkAppId	const char*	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	const char*	`openID` can only be in `Int64` type, which is passed in after being converted to a `const char*`. 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.

Returned values

Returned Value	Description
AV_OK = 0	Initialized SDK successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

Notes on 7015 error code:

The 7015 error code is judged by md5. If this error is reported during integration, please check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party reinforcement, Unity packaging mechanism and other factors, the md5 of the library file will be affected, resulting in misjudgment. **Please ignore this error in the logic for official release**, and try to avoid displaying it in the UI.

Sample code

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

Triggering event callback

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

You can refer to the `EnginePollHelper.cpp` file in the demo.

Calling the `Poll` API periodically:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
class ITMGContext {
protected:
    virtual ~ITMGContext() {}

public:
    virtual void Poll()= 0;
}
```

Sample code

```
// Declaration in the header file
class TMGTestScene : public cocos2d::Scene,public ITMGDelegate
{
void update(float delta);
}

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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext int Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

API prototype

```
ITMGContext int Uninit()
```

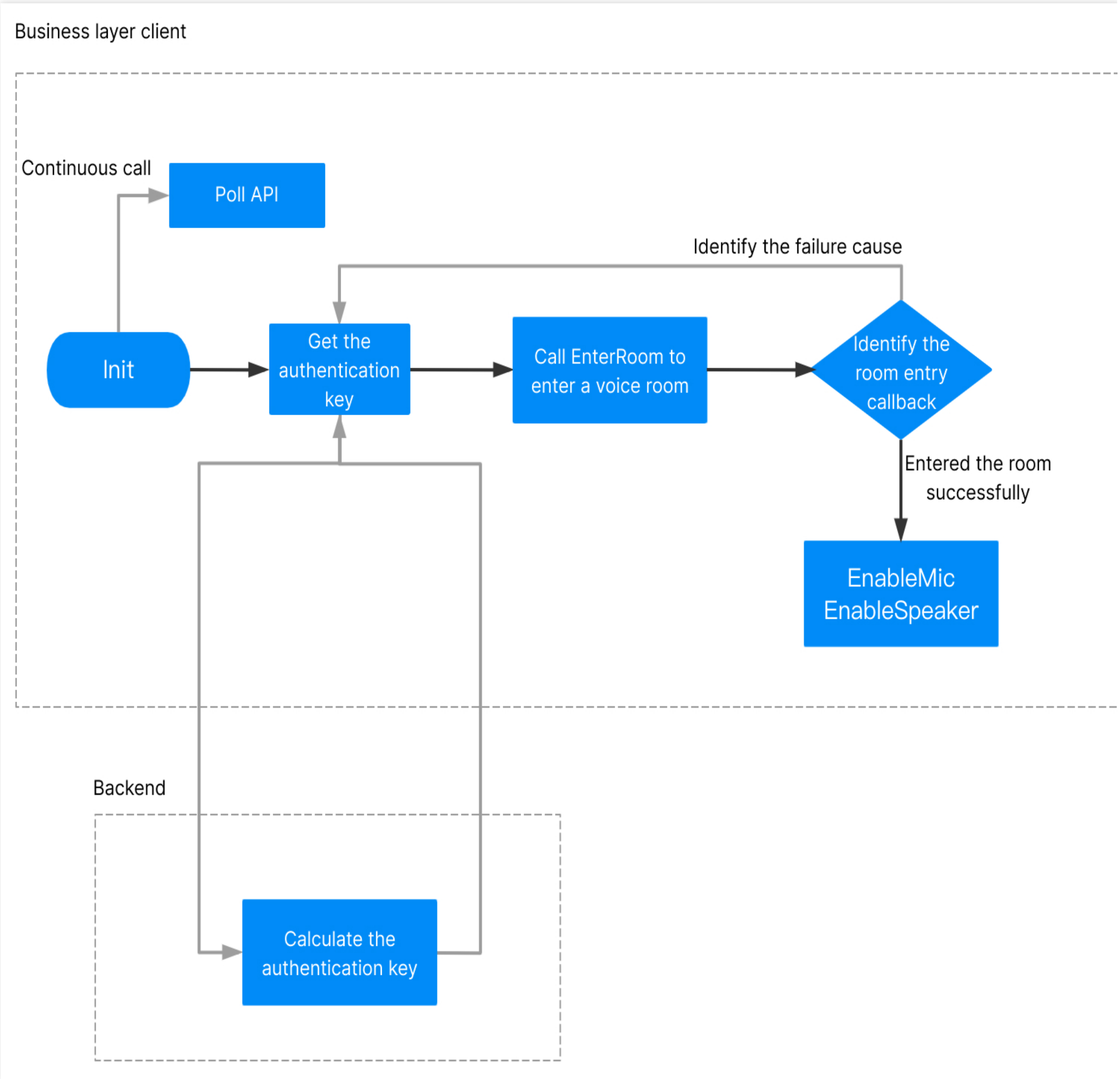
Sample code

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

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, please see FAQs About Voice Chat.



API	Description
GenAuthBuffer	Calculates the local authentication key
EnterRoom	Enters a room
ExitRoom	Exits the room
IsRoomEntered	Determines whether room entry is successful
SwitchRoom	Switches the room quickly

Local authentication key calculation

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

```
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	unsigned int	`AppId` from the Tencent Cloud console
strRoomID	const char*	Room ID, which can contain up to 127 characters.
strOpenID	const char*	User ID, which is the same as `openID` during initialization.
strKey	const char*	Permission key from the Tencent Cloud console
strAuthBuffer	const char*	Returned `authbuff`
bufferLength	int	The length of the returned `authbuff`. `500` is recommended.

Sample code

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

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.

Note：

If the room entry callback result is `0`, the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.

The audio type of the room is determined by the first user entering the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will change to the smooth sound quality. Only after a member in the room calls the `ChangeRoomType` API will the audio type of the room be changed.

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

Parameter	Type	Description
roomID	const char*	Room ID, which can contain up to 127 characters.
roomType	ITMG_ROOM_TYPE	Room type. We recommend you select `ITMG_ROOM_TYPE_FLUENCY` for games. For more information on room audio types, see Sound Quality.
authBuffer	const 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, 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

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

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnected callback prompt `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code Value	Cause and Suggested Solution
7006	Authentication failed: 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

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API during API call; instead, you can directly call the `EnterRoom` API.

API prototype

```
ITMGContext virtual int ExitRoom()
```

Sample code

```
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`.

Sample code

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

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"","result":0}

Determining whether a user has entered a room

This API is used to determine whether the user has entered a room. A value in bool type will be returned. Do not call this API during room entry.

API prototype

```
ITMGContext virtual bool IsRoomEntered()
```

Sample code

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

Switching room

User can call this API to quickly switch the voice chat room after entering the room. After the room is switched, the device is not reset, that is, if the microphone is already enabled in this room, the microphone will keep enabled after the room is switched.

The callback for quickly switching rooms is `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM` , and the fields are `error_info` and `result` .

API prototype

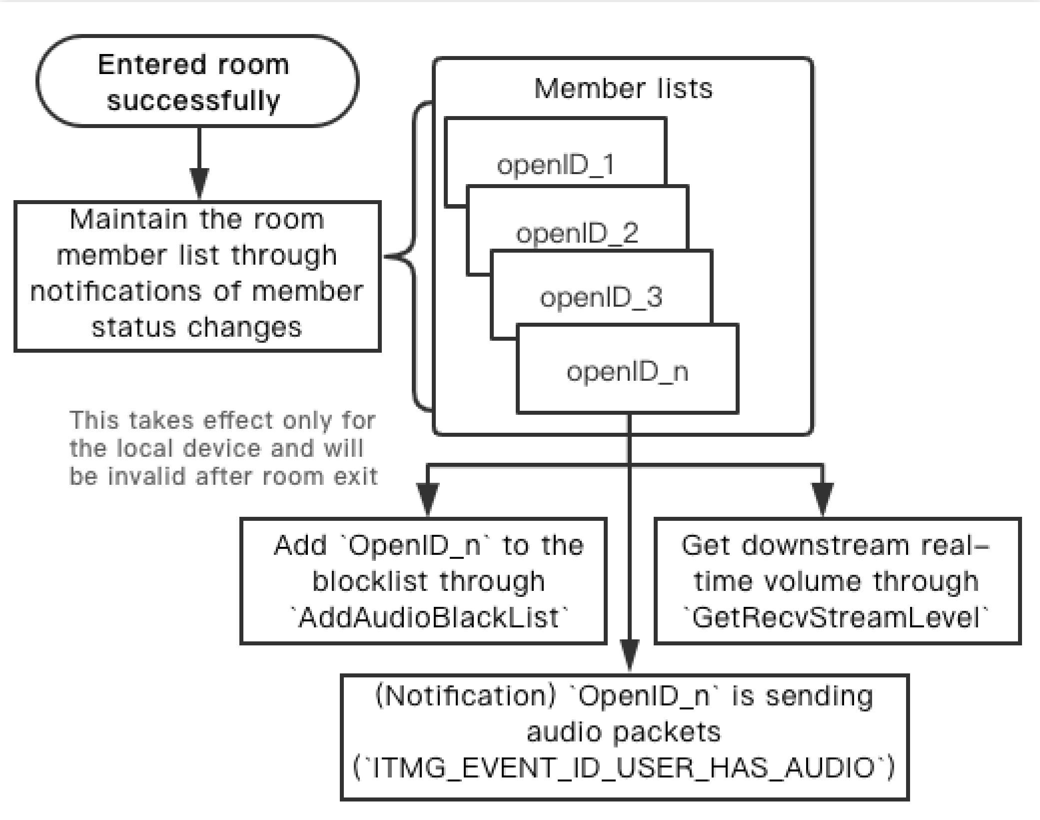
```
ITMGContext virtual int SwitchRoom(const char* targetRoomID, const char* authBuff, int buffLen);
```

Type descriptions

Parameter	Type	Description
targetRoomID	const char*	ID of the room to enter
authBuffer	const char*	Generates a new authentication key with the ID of the room to enter
buffLen	int	Authentication key length

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed
AddAudioBlackList	Mutes a member in the room
RemoveAudioBlackList	Unmutes a user

Notification events of member room entry and speaking status

This API is used to obtain the user speaking in the room and display it in the UI, and to send a notification when someone entering or exiting the room.

Notification for this event will be sent only when the status changes. To get member status in real time, cache the notification when it is received at a higher layer. The event message is `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` , where the data contains `event_id` and `user_list` . The event message will be identified in the `OnEvent` function.

Notifications for audio events are subject to a threshold, and a notification will be sent only when this threshold is exceeded. The notification "A member has stopped sending audio packets" will be sent if no audio packets are received in more than two seconds.

event_id	Description	Maintenance
ITMG_EVENT_ID_USER_ENTER	Return the `openid` of the member entering the room.	Member list
ITMG_EVENT_ID_USER_EXIT	Return the `openid` of the member exiting the room.	Member list
ITMG_EVENT_ID_USER_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
ITMG_EVENT_ID_USER_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
            case ITMG_MAIN_EVNET_TYPE_USER_UPDATE:
        {
        // Process
        // Parse the parameter to get `eventID` and `user_list`
            switch (eventID)
             {
             case ITMG_EVENT_ID_USER_ENTER:
                  // A member enters the room
                  break;
             case ITMG_EVENT_ID_USER_EXIT:
                  // A member exits the room
                break;
            case ITMG_EVENT_ID_USER_HAS_AUDIO:
                // A member sends audio packets
                break;
            case ITMG_EVENT_ID_USER_NO_AUDIO:
                // A member stops sending audio packets
                break;
             default:
                break;
            }
        break;
        }
```



```
    }
}
```

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B;

If B blocks neither A nor C, B can hear both of them;

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
ITMGContext ITMGAudioCtrl int AddAudioBlackList(const char* openId)
```

Parameter	Type	Description
openId	char*	`openid` of the user to be blocked

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->AddAudioBlackList(openId);
```

Unmuting

This API is used to remove an ID from the audio data blacklist. A returned value of 0 indicates the call is successful.

API prototype

```
ITMGContext ITMGAudioCtrl int RemoveAudioBlackList(const char* openId)
```

Parameter	Type	Description
openId	char*	ID to be unblocked

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->RemoveAudioBlackList(openId);
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.

When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend you call the `EnableMic` or `EnableSpeaker` API.

To enable the user to press the mic button on the UI to speak and release it to stop speaking, we recommend you call `EnableAudioCaptureDevice` once during room entry and call `EnableAudioSend` to enable the user to speak while pressing the button.

API	Description
EnableMic	Enables/Disables the mic
GetMicState	Gets the mic status
EnableAudioCaptureDevice	Enables/Disables the capturing device
IsAudioCaptureDeviceEnabled	Gets the capturing device status
EnableAudioSend	Enables/Disables audio upstreaming
IsAudioSendEnabled	Gets the audio upstreaming status
GetMicLevel	Gets the real-time mic volume level
GetSendStreamLevel	Gets real-time audio upstreaming volume
SetMicVolume	Sets the mic volume level
GetMicVolume	Gets the mic volume level

Enabling or disabling mic

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

EnableMic = EnableAudioCaptureDevice + EnableAudioSend

API prototype

```
ITMGAudioCtrl virtual int EnableMic(bool bEnabled)
```

Parameter	Type	Description
bEnabled	bool	To enable the mic, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable mic
ITMGContextGetInstance()->GetAudioCtrl()->EnableMic(true);
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
ITMGAudioCtrl virtual int GetMicState()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicState();
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.
This API can only be called after room entry. The device will be disabled automatically after room exit.
Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioCaptureDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To enable the capturing device, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable capturing device
ITMGContextGetInstance()->GetAudioCtrl()->EnableAudioCaptureDevice(true);
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
ITMGContext virtual bool IsAudioCaptureDeviceEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioCaptureDeviceEnabled();
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain mute. For more information on how to enable/disable the capturing device, please see the `EnableAudioCaptureDevice` API.

API prototype

```
ITMGContext  virtual int EnableAudioSend(bool bEnable)
```

Parameter	Type	Description
bEnable	bool	To enable audio upstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

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

Getting audio upstreaming status

This API is used to get the status of audio upstreaming.

API prototype

```
ITMGContext virtual bool IsAudioSendEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioSendEnabled();
```

Getting the real-time mic volume

This API is used to get the real-time mic volume. An int-type value in the range of 0-100 will be returned. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl virtual int GetMicLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicLevel();
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume. An int-type value in the range of 0-100 will be returned.

API prototype

```
ITMGAudioCtrl virtual int GetSendStreamLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSendStreamLevel();
```

Setting the mic software volume

This API is used to set the mic volume level. The corresponding parameter is `volume` , which is equivalent to attenuating or gaining the captured sound.

API prototype

```
ITMGAudioCtrl virtual int SetMicVolume(int vol)
```

Parameter	Type	Description
vol	int	Value range: 0–200. Default value: 100. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
int micVol = (int)(value * 100);
ITMGContextGetInstance()->GetAudioCtrl()->SetMicVolume(vol);
```

Getting the mic software volume

This API is used to obtain the microphone volume. An "int" value is returned. Value 101 represents API SetMicVolume has not been called.

API prototype

```
ITMGAudioCtrl virtual int GetMicVolume()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicVolume();
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker
GetSpeakerState	Gets the speaker status
EnableAudioPlayDevice	Enables/Disables the playback device
IsAudioPlayDeviceEnabled	Gets playback device status
EnableAudioRecv	Enables/Disables audio downstreaming
IsAudioRecvEnabled	Gets the audio downstreaming status
GetSpeakerLevel	Gets the real-time speaker volume level
GetRecvStreamLevel	Gets the real-time downstreaming audio volume levels of other members in the room
SetSpeakerVolume	Sets the speaker volume level
GetSpeakerVolume	Gets the speaker volume level

Enabling or disabling speaker

This API is used to enable/disable the speaker. **EnableSpeaker = EnableAudioPlayDevice + EnableAudioRecv**

API prototype

```
ITMGAudioCtrl virtual int EnableSpeaker(bool enable)
```

Parameter	Type	Description
enable	bool	To disable the speaker, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

```
// Enable the speaker
ITMGContextGetInstance()->GetAudioCtrl()->EnableSpeaker(true);
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerState()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerState();
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioPlayDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To disable the playback device, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

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

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
ITMGAudioCtrl virtual bool IsAudioPlayDeviceEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioPlayDeviceEnabled();
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain mute. For more information on how to enable/disable the playback device, please see the `EnableAudioPlayDevice` API.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioRecv(bool enable)
```

Parameter	Type	Description
enable	bool	To enable audio downstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

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

Getting audio downstreaming status

This API is used to get the status of audio downstreaming.

API prototype

```
ITMGAudioCtrl virtual bool IsAudioRecvEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioRecvEnabled();
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume. An int-type value will be returned to indicate the volume. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerLevel();
```

Getting the real-time downstreaming audio levels of other members in the room

This API is used to get the real-time audio downstreaming volume of other members in the room. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGAudioCtrl virtual int GetRecvStreamLevel(const char* openId)
```

Parameter	Type	Description
openId	char*	`openId` of other members in the room

Sample code

```
iter->second.level = ITMGContextGetInstance()->GetAudioCtrl()-
>GetRecvStreamLevel(iter->second.openid.c_str());
```

Setting the speaker volume

This API is used to set the speaker volume.

API prototype

```
ITMGAudioCtrl virtual int SetSpeakerVolume(int vol)
```

Parameter	Type	Description
vol	int	Volume level. Value range: 0–200. Default value: 100. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
int vol = 100;
ITMGContextGetInstance()->GetAudioCtrl()->SetSpeakerVolume(vol);
```

Getting the speaker volume

This API is used to get the speaker volume. An int-type value will be returned to indicate the volume. 101 indicates that the `SetSpeakerVolume` API has not been called.
"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerVolume()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerVolume();
```

Device Selection APIs

Device selection APIs can be used only on PC.

API	Description
GetMicListCount	Gets the number of mics
GetMicList	Lists mics
GetSpeakerListCount	Gets the number of speakers
GetSpeakerList	Lists speakers
SelectMic	Selects mics
SelectSpeaker	Selects speakers

Getting the number of mics

This API is used to get the number of mics.

Function prototype

```
ITMGAudioCtrl virtual int GetMicListCount()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicListCount();
```

Enumerating mics

This API is used together with the `GetMicListCount` API to enumerate mics.

Function prototype

```
ITMGAudioCtrl virtual int GetMicList(TMGAudioDeviceInfo* ppDeviceInfoList, int
nCount)

class TMGAudioDeviceInfo
{
public:
    const char* pDeviceID;
    const char* pDeviceName;
};
```

Parameter	Type	Description
ppDeviceInfoList	TMGAudioDeviceInfo	Device list
nCount	int	Number of the mics

`TMGAudioDeviceInfo` Parameter	Type	Description
pDeviceID	const char*	Device ID
pDeviceName	const char*	Device name

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicList(ppDeviceInfoList,nCount);
```

Selecting a mic

This API is used to select a mic. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default mic will be selected.
The 0th device id returned in the GetMicList API is the default device of the call device. If there is a selected call device, it will be maintained by service. If it is unplugged, the call device will be changed back into the default device.

Function prototype

```
ITMGAudioCtrl virtual int SelectMic(const char* pMicID)
```

Parameter	Type	Description
pMicID	const char*	Mic ID, which is from the list returned by `GetMicList` .

Sample code

```
const char* pMicID ="{0.0.1.00000000}.{7b0b712d-3b46-4f7a-bb83-bf9be4047f0d}";
ITMGContextGetInstance()->GetAudioCtrl()->SelectMic(pMicID);
```

Getting the number of speakers

This API is used to get the number of speakers.

Function prototype

```
ITMGAudioCtrl virtual int GetSpeakerListCount()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerListCount();
```

Enumerating speakers

This API is used together with the `GetSpeakerListCount` API to enumerate speakers.

Function prototype

```
ITMGAudioCtrl virtual int GetSpeakerList(TMGAudioDeviceInfo* ppDeviceInfoList,
int nCount)
```

Parameter	Type	Description

ppDeviceInfoList	TMGAudioDeviceInfo	Device list
nCount	int	Number of the speakers

`TMGAudioDeviceInfo` Parameter	Type	Description
pDeviceID	const char*	Device ID
pDeviceName	const char*	Device name

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()-
>GetSpeakerList(ppDeviceInfoList,nCount);
```

Selecting a speaker

This API is used to select a playback device. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default playback device will be selected.

Function prototype

```
ITMGAudioCtrl virtual int SelectSpeaker(const char* pSpeakerID)
```

Parameter	Type	Description
pSpeakerID	const char*	Speaker ID, which is from the list returned by `GetSpeakerList` .

Sample code

```
const char* pSpeakerID ="{0.0.1.00000000}.{7b0b712d-3b46-4f7a-bb83-
bf9be4047f0d}";
ITMGContextGetInstance()->GetAudioCtrl()->SelectSpeaker(pSpeakerID);
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

API prototype

```
ITMGAudioCtrl virtual int EnableLoopBack(bool enable)
```

Parameter	Type	Description
enable	bool	Specifies whether to enable

Sample code

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

Getting a user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, please see the `EnterRoom` API.

API prototype

```
class ITMGRoom {
public:
    virtual ~ITMGRoom() {} ;
    virtual int GetRoomType() = 0;

};
```

Sample code

```
ITMGContext* context = ITMGContextGetInstance();
ITMGContextGetInstance()->GetRoom()->GetRoomType();
```

Changing the room type

This API is used to modify a user's room audio type. For the result, please see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` . The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

API prototype

```
IITMGContext TMGRoom public int ChangeRoomType((ITMG_ROOM_TYPE roomType)
```

Parameter	Type	Description
roomType	ITMG_ROOM_TYPE	Room type to be switched to the target type. For room audio types, please see the `EnterRoom` API.

Sample code

```
ITMGContext* context = ITMGContextGetInstance();
ITMGContextGetInstance()->GetRoom()->ChangeRoomType(ITMG_ROOM_TYPE_FLUENCY);
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result; error_info; new_room_type	{"error_info":"","new_room_type":0

Callback for room type setting completion

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result` , `error_info` , and `new_room_type` . The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description
ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	Indicates that the existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	Indicates that a user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).
ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	Indicates that a user is already in the room and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	Indicates that a room member calls the `ChangeRoomType` API to request a

		change of room audio type.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE
== type)
        {
        // Process room type events
      }
}
```

The monitoring event of room call quality

This is the quality monitoring event, which is triggered once every two seconds after room entry, and its message is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which are as detailed below:

Parameter	Type	Description
weight	int	Value range: 1-50. 50 indicates excellent sound quality, 1 indicates very poor (barely usable) sound quality, and 0 represents an initial meaningless value. Generally, if the value is below 30, the business layer will remind users that the network is poor and recommend them to switch the network.
loss	double	Upstream packet loss rate
delay	int	Voice chat delay in ms

Getting the version number

This API is used to get the SDK version number for analysis.

API prototype

```
ITMGContext virtual const char* GetSDKVersion()
```

Sample code

```
ITMGContextGetInstance()->GetSDKVersion();
```

Setting the log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext int SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL
levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

`ITMG_LOG_LEVEL` is as detailed below:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContextGetInstance()->SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

OS	Path
Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents

Android	/sdcard/Android/data/xxx.xxx.xxx/files
Mac	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext virtual int SetLogPath(const char* logDir)
```

Parameter	Type	Description
logDir	const char*	Path

Sample code

```
cosnt char* logDir = ""// Set a path by yourself

ITMGContext* context = ITMGContextGetInstance();
context->SetLogPath(logDir);
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

API prototype

```
ITMGRoom virtual const char* GetQualityTips()
```

Sample code

```
ITMGContextGetInstance()->GetRoom()->GetQualityTips();
```

Callback Messages

Message	Description	Parameter
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the	result; error

	audio room	
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected for network or other reasons	result; error
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The room members were updated	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	Room reconnection started	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	Room reconnection succeeded	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status changed	result; error_info; sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker was changed	result; error

ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A speaker was added	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A mic was added	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic was changed	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	Room quality message	weight; loss; delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Recording of a voice message was completed	result; file_p
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Upload of a voice message was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Download of a voice message was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Playback of a voice message was completed	result; file_p

ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast recording-to-text conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion was completed	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	A voice message is being converted into text in a streaming manner	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed	result; text;file_id

Speech-to-Text Service

Last updated：2024-01-18 15:11:45

This document describes how to integrate with and debug GME client APIs for the voice messaging and speech-to-text services for Cocos2d.

Key Considerations for Using GME

GME provides the real-time voice service and voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll`.

Caution:

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.

Non-streaming speech-to-text API ***SpeechToText()***: There can be up to 10 concurrent requests per account.

Streaming speech-to-text API ***StartRecordingWithStreamingRecognition()***: There can be up to 50 concurrent requests per account.

Real-time streaming speech-to-text API ***StartRealTimeASR()***: There can be up to 50 concurrent requests per account.

Notes

You have created a GME application and obtained the SDK AppID and key. For more information, see Activating Services.

You have activated **GME real-time voice service and voice messaging and speech-to-text services**. For more information, see Activating Services.

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `AV_OK` will be returned with the value being 0.

GME APIs should be called in the same thread.

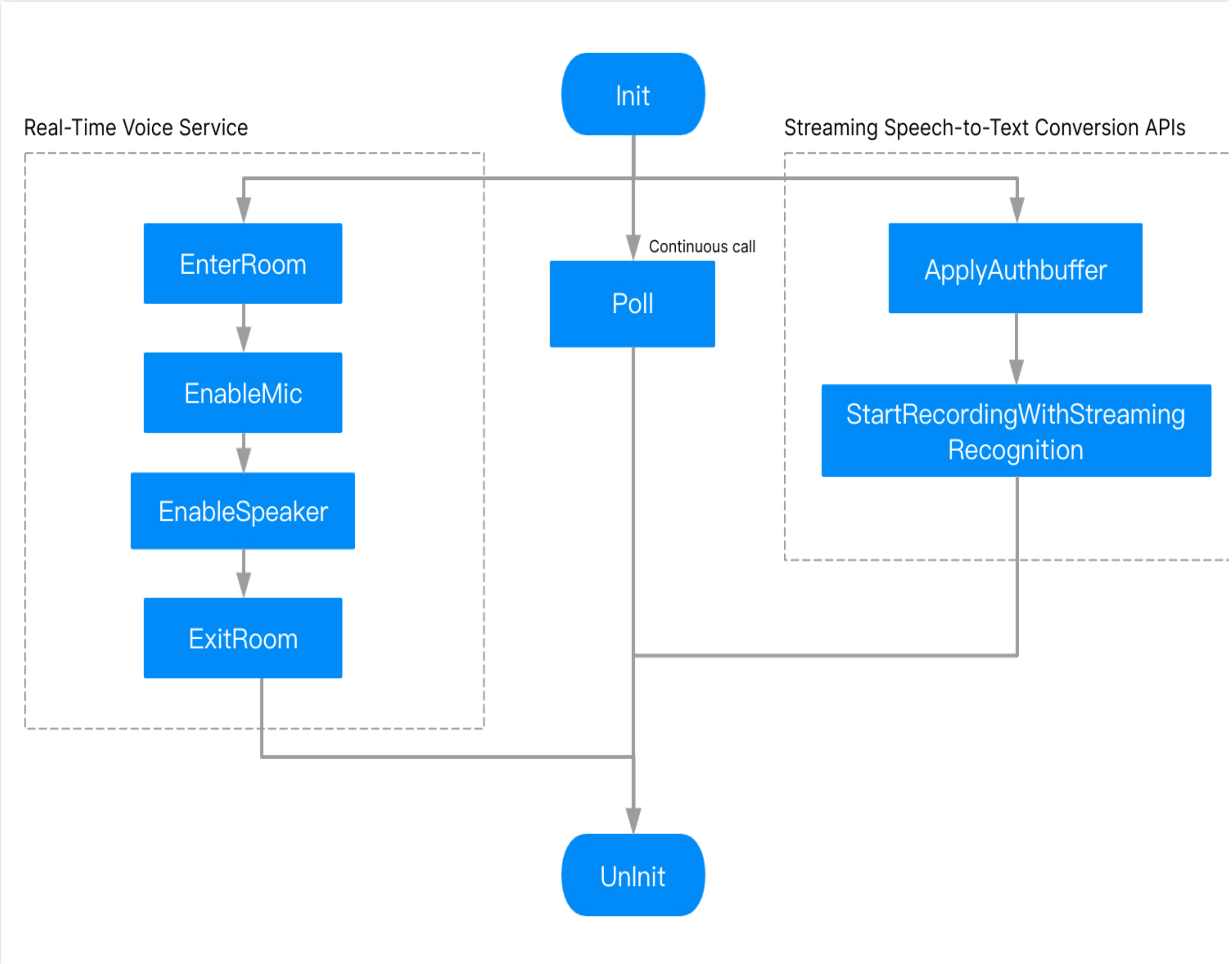
The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error code, see Error Codes.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Initializing authentication, API: ApplyPTTAuthbuffer
4. Starting streaming speech recognition, API: StartRecordingWithStreamingRecognition
5. Stop recording, API: StopRecording
6. Uninitializing GME, API: UnInit

C++ classes

Class	Description
ITMGContext	Key APIs
ITMGPTT	Voice messaging and speech-to-text APIs

Key APIs

API	Description
Init	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME

Importing the header file

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

Callback

Setting callback sample

```
// When initializing the SDK
m_pTmgContext = ITMGContextGetInstance();
m_pTmgContext->SetTMGDelegate(this);

// In the destructor
CTMGSDK_For_AudioDlg::~CTMGSDK_For_AudioDlg()
{
            ITMGContextGetInstance()->SetTMGDelegate(NULL);
}
```

Message delivery

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.

```
// 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;
            }
}
```

Getting singleton

The GME SDK is provided in the form of a singleton, all calls begin with `ITMGContext` , and callbacks are passed to the application through `ITMGDelegate` , which should be configured first.

Sample code

```
ITMGContext* m_pTmgContext;
m_pTmgContext->Init(AppID, OpenID);
```

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

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

Parameter	Type	Description
sdkAppId	const char*	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	const char*	`openID` can only be in `Int64` type, which is passed in after being converted to a `const char*` . 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.

Returned values

Returned Value	Description
AV_OK = 0	Initialized SDK successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

Notes on 7015 error code:

The 7015 error code is judged by md5. If this error is reported during integration, please check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party reinforcement, Unity packaging mechanism and other factors, the md5 of the library file will be affected, resulting in misjudgment. **Please ignore this error in the logic for official release**, and try to avoid displaying it in the UI.

Sample code

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

Triggering event callback

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

You can refer to the `EnginePollHelper.cpp` file in the demo.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
class ITMGContext {
protected:
    virtual ~ITMGContext() {}

public:
    virtual void Poll()= 0;
}
```

Sample code

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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext int Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

API prototype

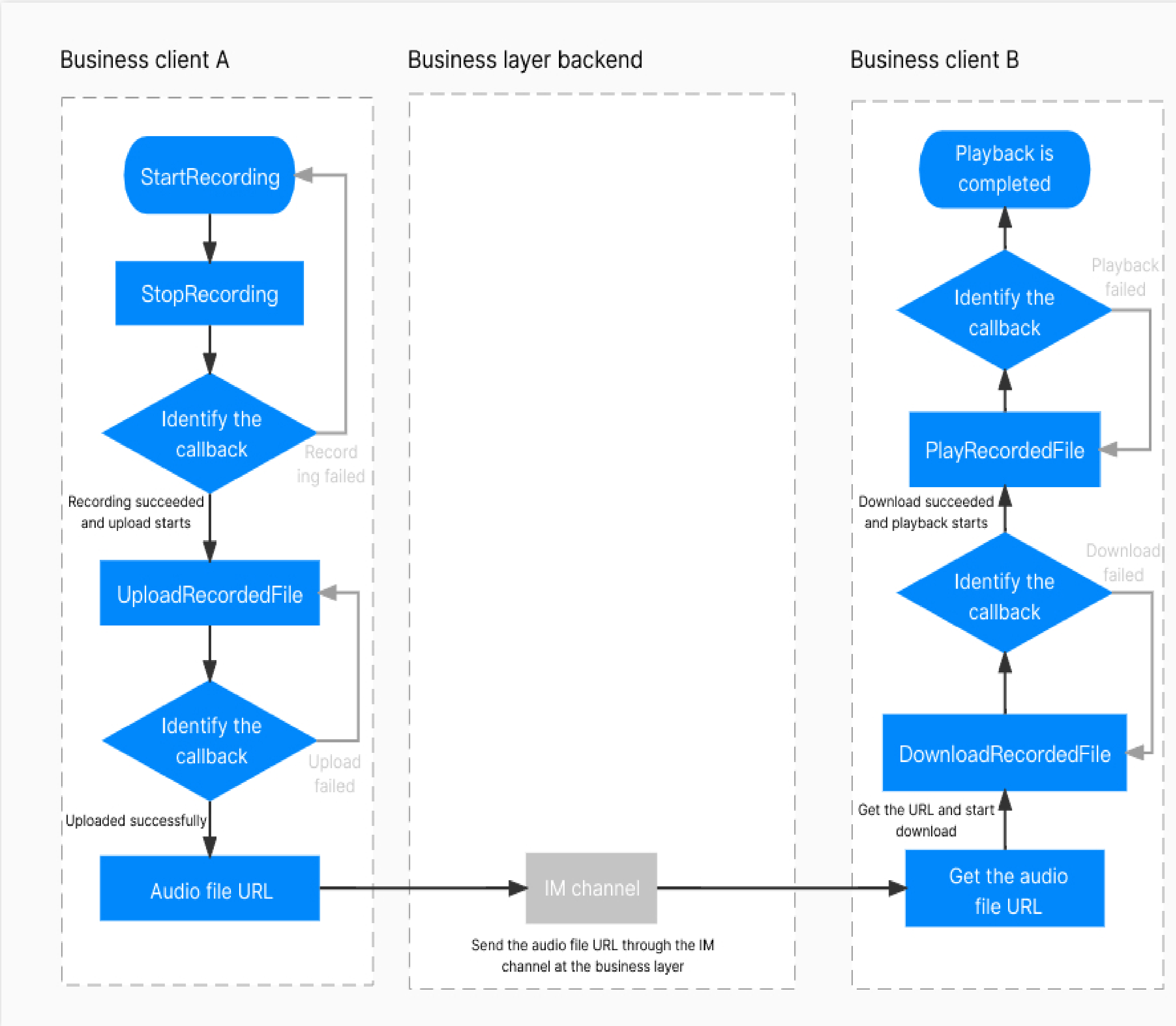
```
ITMGContext int Uninit()
```

Voice Messaging and Speech-to-Text Services

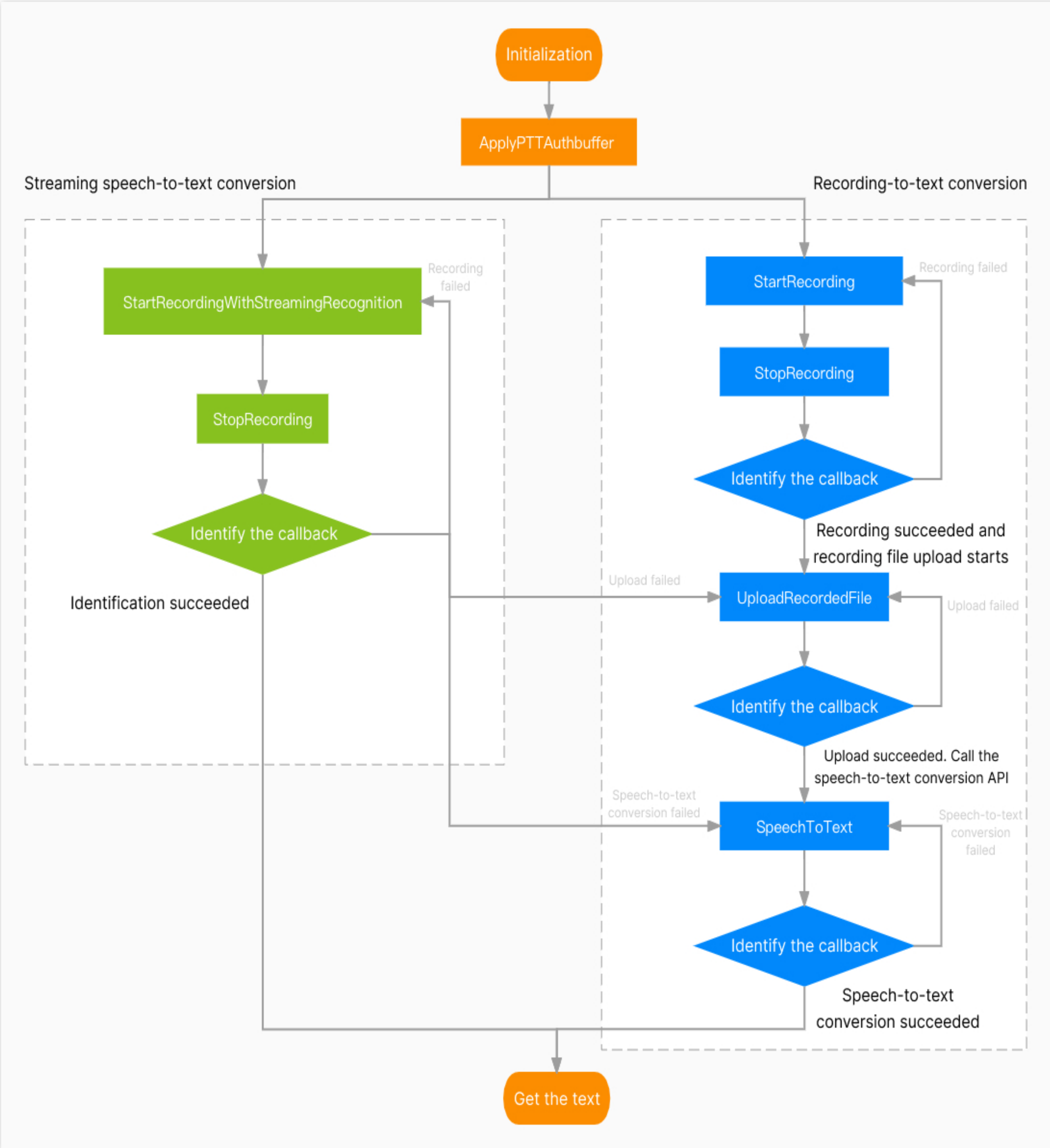
Note:

The speech-to-text service consists of fast recording-to-text conversion and streaming speech-to-text conversion. You do not need to enter a voice chat room when using the voice message service.

The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum recording duration to 10 seconds, call the `SetMaxMessageLength` API to set it after initialization.



Flowchart for using the speech-to-text service



API	Description
GenAuthBuffer	Generates the local authentication key
ApplyPTTAuthbuffer	Initializes authentication
SetMaxMessageLength	Specifies the maximum length of voice message

Generating 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

```
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	`AppId` from the Tencent Cloud console.
strRoomID	const char*	Enter `null` or an empty string
strOpenID	const char*	User Identifier, which is the same as `openID` during initialization.
strKey	const char*	Permission key from the Tencent Cloud console.
strAuthBuffer	const char*	Returned `authbuff` .
bufferLength	int	Length of the `authbuff` passed in. 500 is recommended.

Application authentication

After the authentication information is generated, the authentication is assigned to the SDK.

API prototype

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

Parameter	Type	Description
authBuffer	const char*	Authentication
authBufferLen	int	Authentication length

Sample code

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

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

API prototype

```
ITMGPTT virtual int SetMaxMessageLength(int msTime)
```

Parameter	Type	Description
msTime	int	Audio duration in ms. Value range: 1000 < msTime <= 58000

Sample code

```
int msTime = 10000;
ITMGContextGetInstance()->GetPTT()->SetMaxMessageLength(msTime);
```

Streaming Speech Recognition

Voice messaging and speech-to-text APIs

API	Description
StartRecordingWithStreamingRecognition	Starts streaming recording
StopRecording	Stops recording

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** Stop recording.

API 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	const char*	Path of stored audio file

speechLanguage	const char*	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translateLanguage	const char*	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

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

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen on callback messages in the `OnEvent` notification, which is as detailed below:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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` notification based on the actual needs. The passed parameters include the following four messages.

Message Name	Description
result	Return code indicating whether 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

Note:

The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

Error	Description	Suggested Solution

Code		
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 conversion 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.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

If the error code 4098 is reported, see FAQs for solutions.

Sample code

```
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
isComplete)
```

```
{
                    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 result
Json error")).c_str());
                    }
                    else
                    {
                        if (isComplete) {
                            ::SetWindowText(m_EditUpload.GetSafeHwnd(),
MByteToWChar(root["file_id"].asString()).c_str());
                        }
                        else {
                            std::string isruning =
"STREAMINGRECOGNITION_IS_RUNNING";
                            ::SetWindowText(m_EditUpload.GetSafeHwnd(),
MByteToWChar(isruning).c_str());
                        }
                    }
}
```

Voice Message Recording

The recording process is as follows: start recording > stop recording > return recording callback > start the next recording.

Voice message and speech-to-text APIs

API	Description
StartRecording	Starts recording
PauseRecording	Pauses recording
ResumeRecording	Resumes recording
StopRecording	Stops recording
CancelRecording	Cancels recording

Starting recording

This API is used to start recording.

API prototype

```
ITMGPTT virtual int StartRecording(const char* fileDir)
```

Parameter	Type	Description
fileDir	const char*	Path of stored audio file

Sample code

```
char buffer[256]={0};
snprintf(buffer, sizeof(buffer), "%sunreal_ptt_local.file",
getFilePath().c_str());
ITMGContextGetInstance()->GetPTT()->StartRecording(buffer);
```

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 virtual int StopRecording()
```

Sample code

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

Callback for recording start

The recording start result will be returned through the callback.
To stop recording, call `StopRecording`. The callback for recording start will be returned after the recording is stopped.

Parameter	Type	Description
result	int32	0: recording is completed
filepath	FString	Path of stored recording file, which must be accessible and cannot be the `fileid`

Error codes

Error Code Value	Cause	Suggested Solution
4097	Parameter is empty.	Check whether the API parameters in the code are correct.
4098	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Ensure that the SDK recording feature is used at the right time.
4100	Audio data is not captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	Mic permission is required for using the SDK. To add the permission, please see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

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

```
}
```

Pausing recording

This API is used to pause recording. If you want to resume recording, please call the `ResumeRecording` API.

API prototype

```
ITMGPTT virtual int PauseRecording()
```

Sample code

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

Resuming recording

This API is used to resume recording.

API prototype

```
ITMGPTT virtual int ResumeRecording()
```

Sample code

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

Canceling recording

This API is used to cancel recording. **There is no callback after cancellation**.

API prototype

```
ITMGPTT virtual int CancelRecording()
```

Sample code

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

Voice Message Upload, Download, and Playback

API	Description
UploadRecordedFile	Uploads the audio file
DownloadRecordedFile	Downloads the audio file
PlayRecordedFile	Plays back the audio file
StopPlayFile	Stops playing back the audio file
GetFileSize	Gets audio file size
GetVoiceFileDuration	Gets the audio file duration

Uploading an audio file

This API is used to upload an audio file.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of uploaded audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->UploadRecordedFile(filePath);
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	int32	0: recording is completed
filepath	FString	Path of stored recording file
fileid	FString	File URL path

Error codes

Error Code Value	Cause	Suggested Solution
8193	An error occurred while accessing the file during upload.	Ensure the existence of the file and the validity of the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
8195	A network error occurred.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8200	No `appinfo` is set.	Check whether the `apply` API is called or whether the input parameters are empty.

Sample code

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

Downloading the audio file

This API is used to download an audio file.

API prototype

```
ITMGPTT virtual int DownloadRecordedFile(const char* fileId, const char*
filePath)
```

Parameter	Type	Description
fileId	const char*	URL path of file
filePath	const char*	Local path of saved file

Sample code

```
ITMGContextGetInstance()->GetPTT()->DownloadRecordedFile(fileID,filePath);
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	int32	0: recording is completed
filepath	FString	Path of stored recording file
fileid	FString	URL path of file, which will be retained on the server for 90 days

Error codes

Error Code Value	Cause	Suggested Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
12291	Network storage system exception.	The server failed to get the audio file. Check whether the API parameter `fileid` is

		correct, whether the network is normal, and whether the file exists in COS.
12292	Server file system error.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed during the process of getting the download parameters.	Check whether the device can access the internet.
12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.

Sample code

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

Playing back audio

This API is used to play back audio.

API prototype

```
ITMGPTT virtual int PlayRecordedFile(const char* filePath)
ITMGPTT virtual int PlayRecordedFile(const char* filePath, nt voiceType)
```

Parameter	Type	Description
filePath	const char*	Local audio file path
voicetype	int	Voice changer type. For more information, see Voice Changing Effects.

Error codes

Error Code Value	Cause	Suggested Solution
20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
ITMGContextGetInstance()->GetPTT()->PlayRecordedFile(filePath);
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameter includes `result` and `file_path` .

Parameter	Type	Description
code	int	0: playback is completed
filepath	FString	Path of stored recording file

Error codes

Error Code Value	Cause	Suggested Solution
20481	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	Parameter is empty.	Check whether the API parameters in the code are correct.
20484	Internal error.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
    ...
        else if (eventType == ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE) {
            int32 result = JsonObject->GetIntegerField(TEXT("result"));
            FString filepath = JsonObject->GetStringField(TEXT("file_path"));
            onPttPlayFileCompleted(result, filepath);
        }
    }
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

API prototype

```
ITMGPTT virtual int StopPlayFile()
```

Sample code

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

Getting audio file size

This API is used to get the size of an audio file.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->GetFileSize(filePath);
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->GetVoiceFileDuration(filePath);
```

Fast Recording-to-Text Conversion

API	Description
SpeechToText	Converts speech to text

Converting audio file to text

This API is used to convert a specified audio file to text.

API prototype

```
ITMGPTT virtual void SpeechToText(const char* fileID)
```

Parameter	Type	Description
fileID	const char*	Audio file URL

Sample code

```
ITMGContextGetInstance()->GetPTT()->SpeechToText(fileID);
```

Translating audio file into text in specified language

This API can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation.

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

API prototype

```
ITMGPTT virtual int SpeechToText(const char* fileID,const char* speechLanguage)
ITMGPTT virtual int SpeechToText(const char* fileID,const char*
speechLanguage,const char* translateLanguage)
```

Parameter	Type	Description
fileID	const char*	The URL of the audio file, which will be retained on the server for 90 days.
speechLanguage	const char*	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translatelanguage	const char*	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

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

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result` , `file_path` and `text` (recognized text).

Parameter	Type	Description
result	int32	0: recording is completed
fileid	FString	URL of recording file, which will be retained on the server for 90 days
text	FString	Converted text

Error codes

Error Code Value	Cause	Suggested Solution
32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.

32770	Network failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	Incorrect speech-to-text conversion parameter.	Check whether the API parameter `fileid` in the code is empty.
32785	Speech-to-text translation returned an error.	Error with the backend of voice messaging and speech-to-text feature. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
        switch (eventType) {
            case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
            {
            // Process
            break;
                }
            ...
                    case ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE:
            {
            // Process
            break;
            }
        }
```



```
}
```

Voice Message Volume Level APIs

API	Description
GetMicLevel	Gets real-time mic volume level
SetMicVolume	Sets recording volume level
GetMicVolume	Gets recording volume level
GetSpeakerLevel	Gets real-time speaker volume
SetSpeakerVolume	Sets playback volume level
GetSpeakerVolume	Gets playback volume level

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetMicLevel()
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetMicLevel();
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT virtual int SetMicVolume(int vol)
```

Sample code

```
ITMGContextGetInstance()->GetPTT()->SetMicVolume(100);
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetMicVolume()
```

Sample code

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

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetSpeakerLevel()
```

Sample code

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

Setting the playback volume of voice message

This API is used to set the playback volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT virtual int SetSpeakerVolume(int vol)
```

Sample code

```
ITMGContextGetInstance()->GetPTT()->SetSpeakerVolume(100);
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetSpeakerVolume()
```

Sample code

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

Advanced APIs

Getting version number

This API is used to get the SDK version number for SDK usage analysis.

API prototype

```
ITMGContext virtual const char* GetSDKVersion()
```

Sample code

```
ITMGContextGetInstance()->GetSDKVersion();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext int SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL
levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

`ITMG_LOG_LEVEL` description:

ITMG_LOG_LEVEL	Description

TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContextGetInstance()->SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

OS	Path
Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents
Android	/sdcard/Android/data/xxx.xxx.xxx/files
Mac	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext virtual int SetLogPath(const char* logDir)
```

Parameter	Type	Description
logDir	const char*	Path

Sample code

```
cosnt char* logDir = ""// Set a path by yourself
ITMGContext* context = ITMGContextGetInstance();
context->SetLogPath(logDir);
```


Callback Messages

Message	Description	Parameter
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room	result; error
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected for network or other reasons	result; error
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Room members were updated	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	The reconnection to the room started	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	The reconnection to the room succeeded	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status was changed	result; error_info; sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started	result;

ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker device was changed	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A new speaker device was added	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker device was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A new mic device was added	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic device was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic device was changed	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	The room quality changed	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Voice message recording was completed	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Voice message upload was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Voice message	result; file_path;file

	download was completed	
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Voice message playback was completed	result; file_p
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast speech-to-text conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion was completed	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	Streaming speech-to-text conversion is in progress	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed	result; text;file_id

SDK for Windows
Project Configuration

Last updated：2024-01-18 15:11:45

This document describes how to configure a Windows project for GME APIs for Windows.

SDK Preparations

1. Download the applicable demo and SDK. For more information, please see Download Guide.
2. Decompress the obtained SDK resources.
3. The folder contains:
include: GME SDK header files.
libs: GME SDK dll files.

Configuration Guide

1. Copy the `include` and `libs` folder into the project folder.
2. On the project's property page, add the SDK file address to the `additional include` directory.

Voice Chat

Last updated：2024-01-18 15:11:45

This document describes how to access and debug GME client APIs for the voice chat feature for Windows.

Key Considerations for Using GME

GME provides the real-time voice, voice message, and speech-to-text conversion services, which all depend on core APIs such as `Init` and `Poll` .

Key notes

You have created a GME application and obtained the `AppID` and `Key` of the SDK as instructed in Activating Services.

You have **activated the real-time voice, voice message, and speech-to-text services of GME** as instructed in Activating Services.

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.

GME APIs should be called in the same thread.

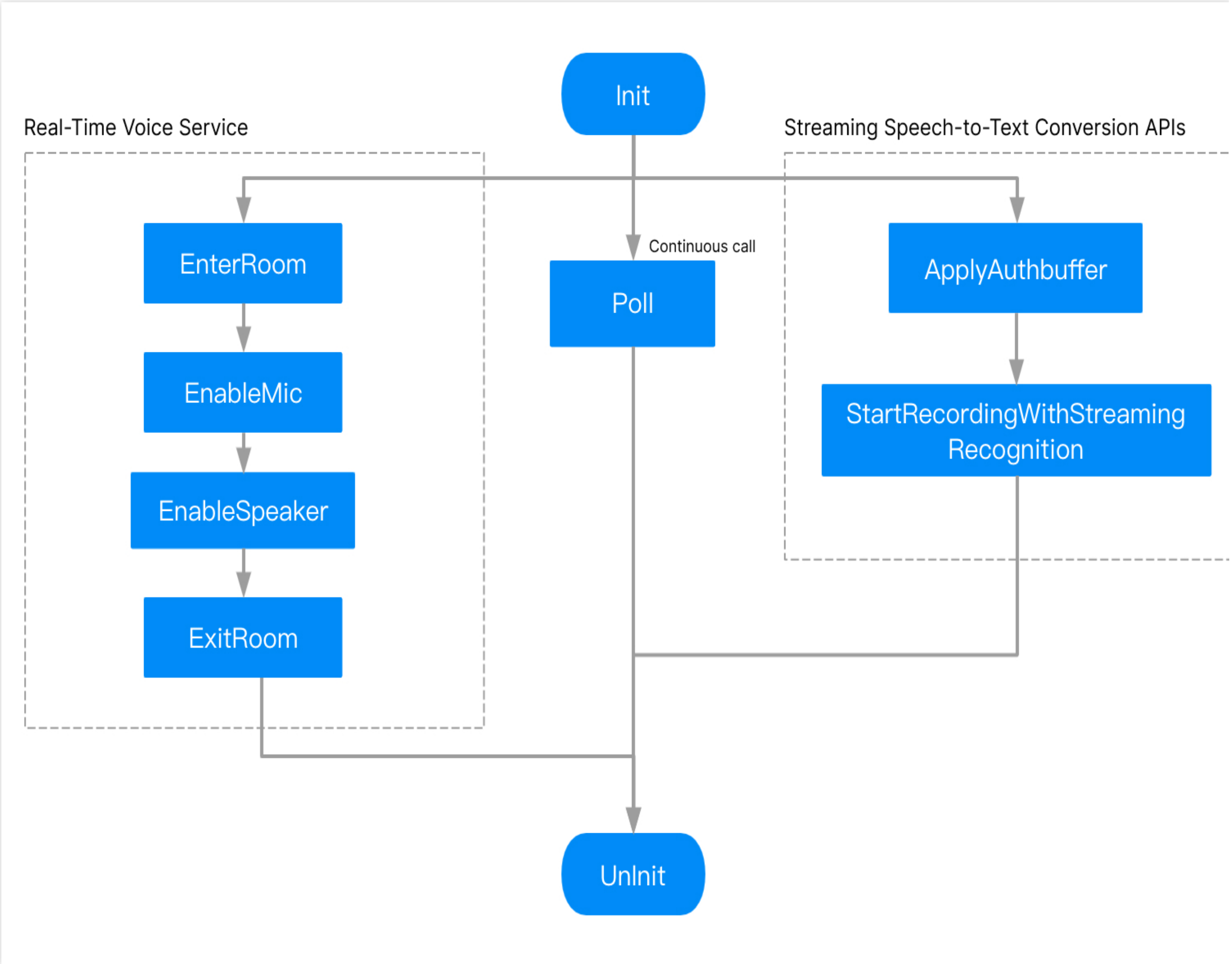
The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error code, please see Error Codes.

Connecting to the SDK

Directions

Key processes involved in SDK connection are as follows:



1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Entering a voice chat room, API: EnterRoom
4. Enabling the microphone, API: EnableMic
5. Enabling the speaker, API: EnableSpeaker
6. Exiting a voice room, API: ExitRoom
7. Uninitializing GME, API: UnInit

C++ classes

Class	Description
ITMGContext	Key APIs
ITMGDelegate	Callback APIs
ITMGRoom	Room APIs

ITMGRoomManager	Room management APIs as described in Integrating GME Chat Room Management
ITMGAudioCtrl	Audio APIs
ITMGAudioEffectCtrl	Sound effect and accompaniment APIs
ITMGPTT	Voice message and speech-to-text conversion APIs

Key APIs

API	Description
Init	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME

Imported header files

You need to import the header file `tmg_sdk.h` first before you can access GME. The classes in the header file inherit `ITMGDelegate` for message delivery and callback.

Sample code

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

Callback

Setting callback sample

```
// When initializing the SDK
m_pTmgContext = ITMGContextGetInstance();
m_pTmgContext->SetTMGDelegate(this);

// In the destructor
CTMGSDK_For_AudioDlg::~CTMGSDK_For_AudioDlg()
```

```
{
            ITMGContextGetInstance()->SetTMGDelegate(NULL);
}
```

Message delivery

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.

```
// 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;
            }
}
```

Getting singleton

The GME SDK is provided in the form of a singleton, all calls begin with `ITMGContext` , and callbacks are passed to the application through `ITMGDelegate` , which should be configured first.

Sample code

```
ITMGContext* m_pTmgContext;
m_pTmgContext->Init(AppID, OpenID);
```

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

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

Parameter	Type	Description
sdkAppId	const char*	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	const char*	`openID` can only be in `Int64` type, which is passed in after being converted to a `const char*`. 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.

Returned values

Returned Value	Description
AV_OK = 0	Initialized SDK successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

Notes on 7015 error code

The 7015 error code is judged by md5. If this error is reported during integration, please check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party reinforcement, Unity packaging mechanism and other factors, the md5 of the library file will be affected, resulting in misjudgment. **Please ignore this error in the logic for official release**, and try to avoid displaying it in the UI.

Sample code

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

Triggering event callback

Event callbacks can be triggered by periodically calling the `Poll` API in `update`. `Poll` is the message pump of GME, and the `Poll` API should be called periodically for GME to trigger event callbacks; otherwise, the entire

SDK service will run exceptionally.

You can refer to the `EnginePollHelper.cpp` file in the demo.

Calling the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
class ITMGContext {
protected:
    virtual ~ITMGContext() {}

public:
    virtual void Poll()= 0;
}
```

Sample code

```
// Declaration in the header file

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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext int Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

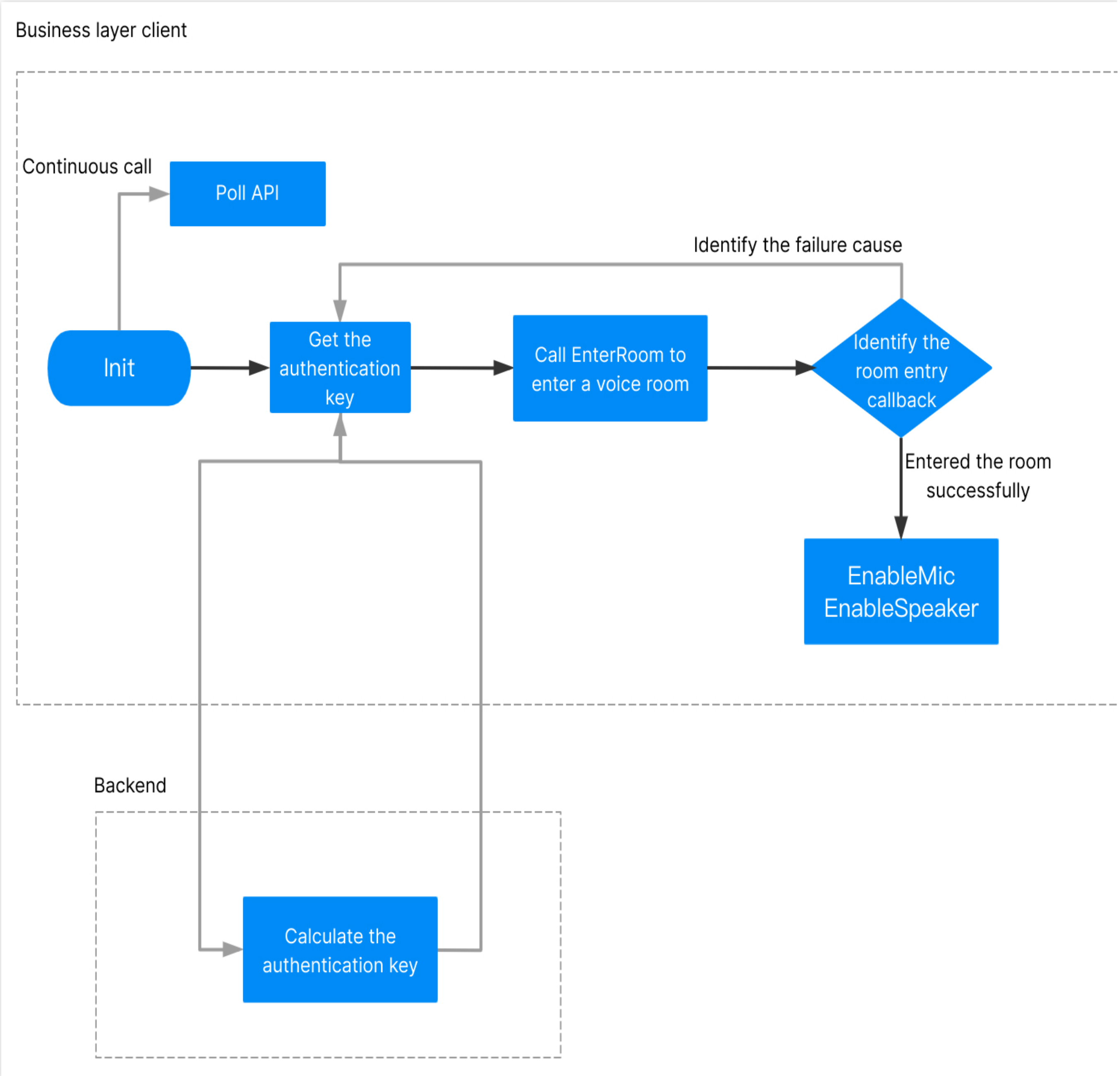
API prototype

```
ITMGContext int Uninit()
```

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, please see FAQs About Voice Chat.



API	Description
GenAuthBuffer	Calculates the local authentication key
EnterRoom	Enters a room
ExitRoom	Exits the room
IsRoomEntered	Determines whether room entry is successful
SwitchRoom	Switches the room quickly

Local authentication key calculation

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

```
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	unsigned int	`AppId` from the Tencent Cloud console
strRoomID	const char*	Room ID, which can contain up to 127 characters.
strOpenID	const char*	User ID, which is the same as `openID` during initialization.
strKey	const char*	Permission key from the Tencent Cloud console
strAuthBuffer	const char*	Returned `authbuff`
bufferLength	int	Length of the `authbuff` 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);
```

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.

Note:

If the room entry callback result is `0`, the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.

The audio type of the room is determined by the first user entering the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will change to the smooth sound quality. Only after a member in the room calls the `ChangeRoomType` API will the audio type of the room be changed.

API prototype

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

Parameter	Type	Description
roomID	const char*	Room ID, which can contain up to 127 characters.
roomType	ITMG_ROOM_TYPE	Room type. We recommend you select `ITMG_ROOM_TYPE_FLUENCY` for games. For more information on room audio types, see Sound Quality.
authBuffer	const 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, 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

```
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());
                                        }
                            }
}
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnected callback prompt `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code Value	Cause and Suggested Solution
7006	Authentication failed: 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.

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API; instead, you can directly call the `EnterRoom` API.

API prototype

```
ITMGContext virtual int ExitRoom()
```

Sample code

```
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`.

Sample code

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

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"","result":0}

Determining whether user has entered room

This API is used to determine whether the user has entered a room. A value in bool type will be returned. Do not call this API during room entry.

API prototype

```
ITMGContext virtual bool IsRoomEntered()
```

Sample code

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

Switching room

User can call this API to quickly switch the voice chat room after entering the room. After the room is switched, the device is not reset, that is, if the microphone is already enabled in this room, the microphone will keep enabled after the room is switched.

The callback for quickly switching rooms is `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM` , and the fields are `error_info` and `result` .

API prototype

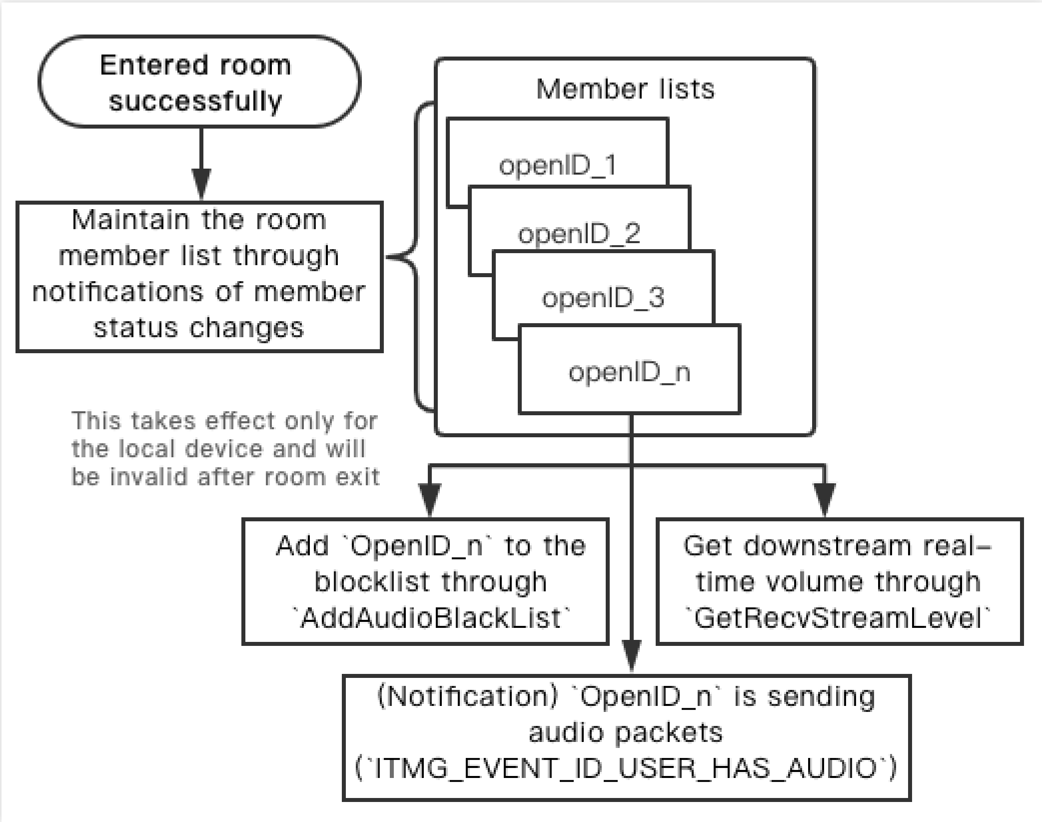
```
ITMGContext virtual int SwitchRoom(const char* targetRoomID, const char*
authBuff, int buffLen);
```

Type descriptions

Parameter	Type	Description
targetRoomID	const char*	ID of the room to enter
authBuffer	const char*	Generates a new authentication key with the ID of the room to enter
buffLen	int	Authentication key length

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed
AddAudioBlackList	Mutes a member in the room
RemoveAudioBlackList	Unmutes a user

Notification events of member room entry and speaking status

This API is used to obtain the user speaking in the room and display it in the UI, and to send a notification when someone entering or exiting the room.

Notification for this event will be sent only when the status changes. To get member status in real time, cache the notification when it is received at a higher layer. The event message is `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` , where the data contains `event_id` and `user_list` . The event message will be identified in the `OnEvent` function.

Notifications for audio events are subject to a threshold, and a notification will be sent only when this threshold is exceeded. The notification "A member has stopped sending audio packets" will be sent if no audio packets are received in more than two seconds.

event_id	Description	Maintenance
ITMG_EVENT_ID_USER_ENTER	Return the `openid` of the member entering the	Member list

	room.	
ITMG_EVENT_ID_USER_EXIT	Return the `openid` of the member exiting the room.	Member list
ITMG_EVENT_ID_USER_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
ITMG_EVENT_ID_USER_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
            case ITMG_MAIN_EVNET_TYPE_USER_UPDATE:
        {
        // Process
        // Parse the parameter to get `eventID` and `user_list`
            switch (eventID)
             {
             case ITMG_EVENT_ID_USER_ENTER:
                  // A member enters the room
                  break;
             case ITMG_EVENT_ID_USER_EXIT:
                  // A member exits the room
                break;
            case ITMG_EVENT_ID_USER_HAS_AUDIO:
                // A member sends audio packets
                break;
            case ITMG_EVENT_ID_USER_NO_AUDIO:
                // A member stops sending audio packets
                break;
             default:
                break;
            }
        break;
        }
    }
}
```

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B;

If B blocks neither A nor C, B can hear both of them;

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
ITMGContext ITMGAudioCtrl int AddAudioBlackList(const char* openId)
```

Parameter	Type	Description
openId	char*	`openid` of the user to be blocked

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->AddAudioBlackList(openId);
```

Unmuting

This API is used to remove an ID from the audio data blacklist. A returned value of 0 indicates the call is successful.

API prototype

```
ITMGContext ITMGAudioCtrl int RemoveAudioBlackList(const char* openId)
```

Parameter	Type	Description
openId	char*	ID to be unblocked

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->RemoveAudioBlackList(openId);
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.

When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend you call the `EnableMic` or `EnableSpeaker` API.
To enable the user to press the mic button on the UI to speak and release it to stop speaking, we recommend you call `EnableAudioCaptureDevice` once during room entry and call `EnableAudioSend` to enable the user to speak while pressing the button.

API	Description
EnableMic	Enables/Disables the mic
GetMicState	Gets the mic status
EnableAudioCaptureDevice	Enables/Disables the capturing device
IsAudioCaptureDeviceEnabled	Gets the capturing device status
EnableAudioSend	Enables/Disables audio upstreaming
IsAudioSendEnabled	Gets the audio upstreaming status
GetMicLevel	Gets the real-time mic volume level
GetSendStreamLevel	Gets real-time audio upstreaming volume
SetMicVolume	Sets the mic volume level
GetMicVolume	Gets the mic volume level

Enabling or disabling mic

This API is used to enable/disable the mic. The mic and speaker are not enabled by default after room entry.
EnableMic = EnableAudioCaptureDevice + EnableAudioSend

API prototype

```
ITMGAudioCtrl virtual int EnableMic(bool bEnabled)
```

Parameter	Type	Description
bEnabled	bool	To enable the mic, set this parameter to `true`, otherwise, set it to `false`.

Sample code

```
// Enable mic
ITMGContextGetInstance()->GetAudioCtrl()->EnableMic(true);
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
ITMGAudioCtrl virtual int GetMicState()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicState();
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.
This API can only be called after room entry. The device will be disabled automatically after room exit.
Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioCaptureDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To enable the capturing device, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable capturing device
ITMGContextGetInstance()->GetAudioCtrl()->EnableAudioCaptureDevice(true);
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
ITMGContext virtual bool IsAudioCaptureDeviceEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioCaptureDeviceEnabled();
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain mute. For more information on how to enable/disable the capturing device, please see the `EnableAudioCaptureDevice` API.

API prototype

```
ITMGContext  virtual int EnableAudioSend(bool bEnable)
```

Parameter	Type	Description
bEnable	bool	To enable audio upstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

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

Getting audio upstreaming status

This API is used to get the status of audio upstreaming.

API prototype

```
ITMGContext virtual bool IsAudioSendEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioSendEnabled();
```

Getting the real-time mic volume

This API is used to get the real-time mic volume. An int-type value in the range of 0-100 will be returned. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl virtual int GetMicLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicLevel();
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume. An int-type value in the range of 0-100 will be returned.

API prototype

```
ITMGAudioCtrl virtual int GetSendStreamLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSendStreamLevel();
```

Setting the mic software volume

This API is used to set the mic volume level. The corresponding parameter is `volume` , which is equivalent to attenuating or gaining the captured sound.

API prototype

```
ITMGAudioCtrl virtual int SetMicVolume(int vol)
```

Parameter	Type	Description
vol	int	Value range: 0–200. Default value: 100. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
int micVol = (int)(value * 100);
ITMGContextGetInstance()->GetAudioCtrl()->SetMicVolume(vol);
```

Getting the mic software volume

This API is used to obtain the microphone volume. An "int" value is returned. Value 101 represents API SetMicVolume has not been called.

API prototype

```
ITMGAudioCtrl virtual int GetMicVolume()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetMicVolume();
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker
GetSpeakerState	Gets the speaker status
EnableAudioPlayDevice	Enables/Disables the playback device
IsAudioPlayDeviceEnabled	Gets playback device status
EnableAudioRecv	Enables/Disables audio downstreaming
IsAudioRecvEnabled	Gets the audio downstreaming status
GetSpeakerLevel	Gets the real-time speaker volume level
GetRecvStreamLevel	Gets the real-time downstreaming audio volume levels of other members in the room
SetSpeakerVolume	Sets the speaker volume level
GetSpeakerVolume	Gets the speaker volume level

Enabling or disabling speaker

This API is used to enable/disable the speaker. **EnableSpeaker = EnableAudioPlayDevice + EnableAudioRecv**

API prototype

```
ITMGAudioCtrl virtual int EnableSpeaker(bool enable)
```

Parameter	Type	Description
enable	bool	To disable the speaker, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

```
// Enable the speaker
ITMGContextGetInstance()->GetAudioCtrl()->EnableSpeaker(true);
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerState()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerState();
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioPlayDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To disable the playback device, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

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

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
ITMGAudioCtrl virtual bool IsAudioPlayDeviceEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioPlayDeviceEnabled();
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain mute. For more information on how to enable/disable the playback device, please see the `EnableAudioPlayDevice` API.

API prototype

```
ITMGAudioCtrl virtual int EnableAudioRecv(bool enable)
```

Parameter	Type	Description
enable	bool	To enable audio downstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

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

Getting audio downstreaming status

This API is used to get the status of audio downstreaming.

API prototype

```
ITMGAudioCtrl virtual bool IsAudioRecvEnabled()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->IsAudioRecvEnabled();
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume. An int-type value will be returned to indicate the volume. It is recommended to call this API once every 20 ms.

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerLevel()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerLevel();
```

Getting the real-time downstreaming audio levels of other members in room

This API is used to get the real-time audio downstreaming volume of other members in the room. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGAudioCtrl virtual int GetRecvStreamLevel(const char* openId)
```

Parameter	Type	Description
openId	char*	`openId` of other members in the room

Sample code

```
iter->second.level = ITMGContextGetInstance()->GetAudioCtrl()-
>GetRecvStreamLevel(iter->second.openid.c_str());
```

Dynamically setting the volume of a member of the room

This API is used to set the volume of a member in the room. It takes effect only on the local.

API prototype

```
ITMGAudioCtrl virtual int SetSpeakerVolumeByOpenID(const char* openId, int vol)
= 0;
```

Parameter	Type	Description
openId	const char*	`OpenID` of the user whose volume level needs to be set
vol	int	Percentage. Recommended value range: 0–200. Default value: `100` .

Getting volume percentage

This API is used to get the volume level set by `SetSpeakerVolumeByOpenID` .

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerVolumeByOpenID(const char* openId) = 0;
```

Parameter	Type	Description
openId	const char*	`OpenID` of the user whose volume level needs to be set

Returned values

API returns volume percentage set by OpenID, where 100 is by default.

Setting the speaker volume

This API is used to set the speaker volume.

API prototype

```
ITMGAudioCtrl virtual int SetSpeakerVolume(int vol)
```

Parameter	Type	Description
vol	int	Volume level. Value range: 0–200. Default value: 100. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
int vol = 100;
ITMGContextGetInstance()->GetAudioCtrl()->SetSpeakerVolume(vol);
```

Getting the speaker volume

This API is used to get the speaker volume. An int-type value will be returned to indicate the volume. 101 indicates that the `SetSpeakerVolume` API has not been called.
"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

API prototype

```
ITMGAudioCtrl virtual int GetSpeakerVolume()
```

Sample code

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSpeakerVolume();
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

API prototype

```
ITMGAudioCtrl virtual int EnableLoopBack(bool enable)
```

Parameter	Type	Description
enable	bool	Specifies whether to enable

Sample code

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

Getting user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, please see the `EnterRoom` API.

API prototype

```
class ITMGRoom {
public:
    virtual ~ITMGRoom() {} ;
    virtual int GetRoomType() = 0;

};
```

Sample code

```
ITMGContext* context = ITMGContextGetInstance();
ITMGContextGetInstance()->GetRoom()->GetRoomType();
```

Getting the room ID

This API is used to get the voice chat room ID and can be called only after a successful room entry.

API prototype

```
ITMGRoom virtual int GetRoomID(char* pBuffer, int nLength) = 0;
```

Parameter	Type	Description
pBuffer	char*	It is used to receive the returned `roomid` .
nLength	int	`pBuffer` length. Value range: 128–256.

Modifying user's room audio type

This API is used to modify a user's room audio type. For the result, please see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` . The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

API prototype

```
IITMGContext TMGRoom public int ChangeRoomType((ITMG_ROOM_TYPE roomType)
```

Parameter	Type	Description
roomType	ITMG_ROOM_TYPE	Room type to be switched to the target type. For room audio types, please see the `EnterRoom` API.

Sample code

```
ITMGContext* context = ITMGContextGetInstance();
ITMGContextGetInstance()->GetRoom()->ChangeRoomType(ITMG_ROOM_TYPE_FLUENCY);
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result; error_info; new_room_type	{"error_info":"","new_room_type":0

Callback for modifying the room type

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result` , `error_info` , and `new_room_type` . The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description
ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	The existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	A user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).
ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	A user is already in the room, and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	A room member calls the `ChangeRoomType` API to request a change of room audio type.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE
== type)
        {
        // Process room type events
      }
}
```

The monitoring event of room call quality

This is the quality monitoring event, which is triggered once every two seconds after room entry, and its message is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY`. The returned parameters include `weight`, `loss`, and `delay`, which are as detailed below:

Parameter	Type	Description
weight	int	Value range: 1-50. 50 indicates excellent sound quality, 1 indicates very poor (barely usable) sound quality, and 0 represents an initial meaningless value. Generally, if the value is below 30, the business layer will remind users that the network is poor and recommend them to switch the network.
loss	double	Upstream packet loss rate
delay	int	Voice chat delay in ms

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
ITMGContext virtual const char* GetSDKVersion()
```

Sample code

```
ITMGContextGetInstance()->GetSDKVersion();
```

Checking mic status

Function prototype

```
ITMGContext virtual ITMG_CHECK_MIC_STATUS CheckMic() = 0;
```

Returned value handling

Returned Value	Description	Handling
ITMG_CHECK_MIC_STATUS_AVAILABLE = 0	Normally available	No handling required
ITMG_CHECK_MIC_STATUS_NO_GRANTED = 2	Access not obtained/denied	The access permission needs to be obtained before the mic is enabled
ITMG_CHECK_MIC_STATUS_INVALID_MIC = 3	No device available	Generally, this error will be reported on PCs when no mics are available. Prompt the user to insert a headset or mic
ITMG_CHECK_MIC_STATUS_NOT_INIT = 5	Not initialized	Call `EnableMic` after `Init`

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext int SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL
levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

`ITMG_LOG_LEVEL` is as detailed below:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContextGetInstance()->SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

OS	Path
Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents
Android	/sdcard/Android/data/xxx.xxx.xxx/files
Mac	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext virtual int SetLogPath(const char* logDir)
```

Parameter	Type	Description
logDir	const char*	Path

Sample code

```
cosnt char* logDir = ""// Set a path by yourself

ITMGContext* context = ITMGContextGetInstance();
context->SetLogPath(logDir);
```

Getting the printed log path

This API is used to get the log path. Its returned value is a string of `const char*` type.

API prototype

```
ITMGContext virtual const char* GetLogPath() = 0;
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

API prototype

```
ITMGRoom virtual const char* GetQualityTips()
```

Sample code

```
ITMGContextGetInstance()->GetRoom()->GetQualityTips();
```

Callback Messages

Message	Description	Parameter
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room	result; error

ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected for network or other reasons	result; error
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The room members were updated	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	Room reconnection started	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	Room reconnection succeeded	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status changed	result; error_info; sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker was changed	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A speaker was added	result; error

ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A mic was added	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic was changed	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	Room quality message	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Recording of a voice message was completed	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Upload of a voice message was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Download of a voice message was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Playback of a voice message was completed	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast recording-to-	result; text;file_id

	text conversion was completed	
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion was completed	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	A voice message is being converted into text in a streaming manner	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed	result; text;file_id

Speech-to-Text Service

Last updated：2024-01-18 15:11:45

This document describes how to integrate with and debug GME client APIs for the voice messaging and speech-to-text services for Windows.

Key Considerations for Using GME

GME provides the real-time voice and voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

Notes

You have created a GME application and obtained the SDK AppID and key. For more information, see Activating Services.

You have activated **GME real-time voice and voice messaging and speech-to-text services**. For more information, see Activating Services.

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error code, see Error Codes.

Note:

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.

Non-streaming speech-to-text API ***SpeechToText()***: There can be up to 10 concurrent requests per account.

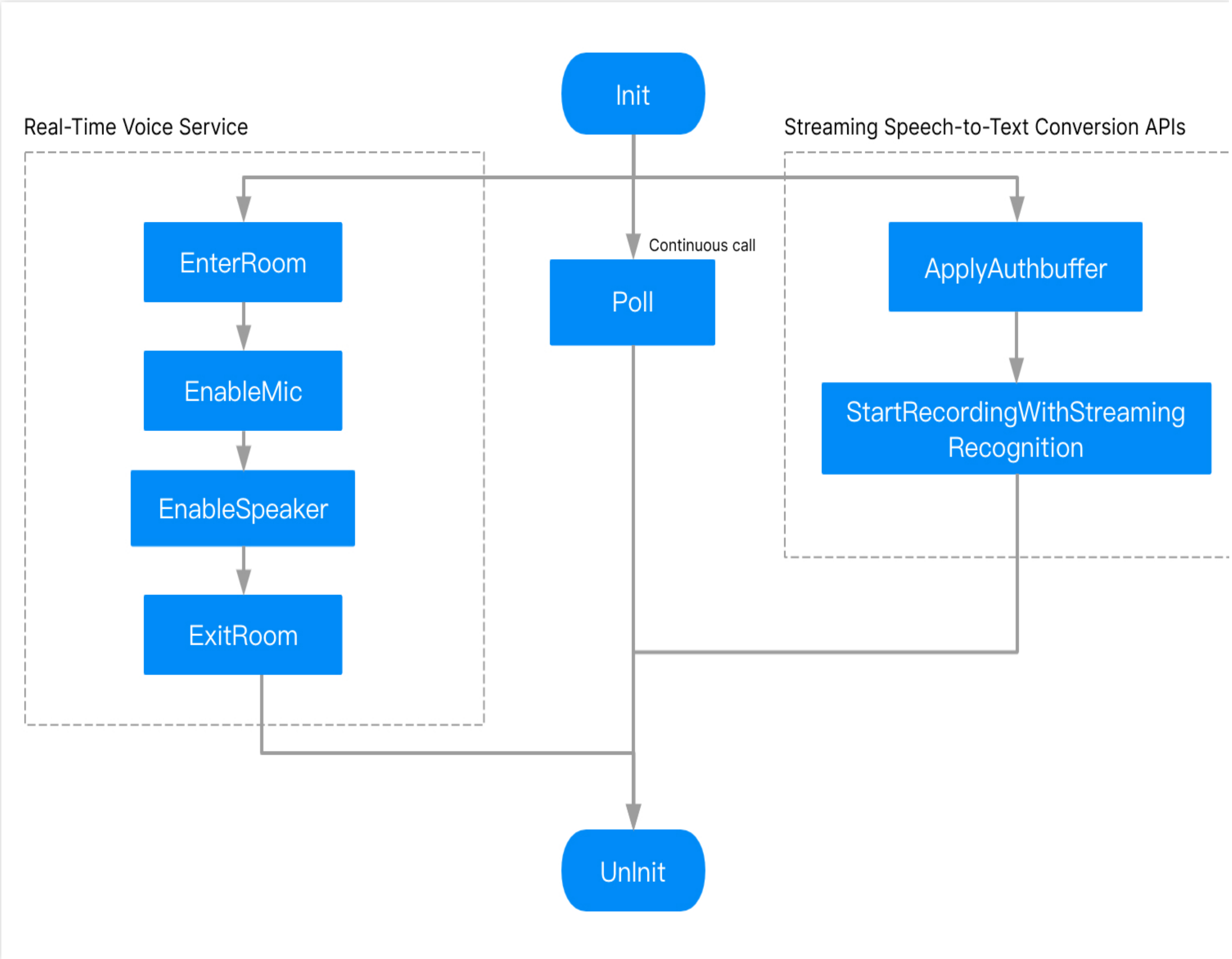
Streaming speech-to-text API ***StartRecordingWithStreamingRecognition()***: There can be up to 50 concurrent requests per account.

Real-time streaming speech-to-text API ***StartRealTimeASR()***: There can be up to 50 concurrent requests per account.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Initializing authentication, API: ApplyPTTAuthbuffer
4. Starting streaming speech recognition, API: StartRecordingWithStreamingRecognition
5. Stop recording, API: StopRecording
6. Uninitializing GME, API: UnInit

C++ classes

Class	Description
ITMGContext	Key APIs
ITMGPTT	Voice messaging and speech-to-text APIs

Key APIs

API	Description
Init	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME

Importing the header file

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

Callback

Setting callback sample

```
// When initializing the SDK
m_pTmgContext = ITMGContextGetInstance();
m_pTmgContext->SetTMGDelegate(this);

// In the destructor
CTMGSDK_For_AudioDlg::~CTMGSDK_For_AudioDlg()
{
            ITMGContextGetInstance()->SetTMGDelegate(NULL);
}
```

Message delivery

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.

```
// 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;
            }
}
```

Getting singleton

The GME SDK is provided in the form of a singleton, all calls begin with `ITMGContext` , and callbacks are passed to the application through `ITMGDelegate` , which should be configured first.

Sample code

```
ITMGContext* m_pTmgContext;
m_pTmgContext->Init(AppID, OpenID);
```

Initializing the 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

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

Parameter	Type	Description
sdkAppId	const char*	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	const char*	`openID` can only be in `Int64` type, which is passed in after being converted to a `const char*` . 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.

Returned values

Returned Value	Description
AV_OK = 0	Initialized SDK successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

Notes on 7015 error code

The 7015 error code is judged by md5. If this error is reported during integration, please check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party reinforcement, Unity packaging mechanism and other factors, the md5 of the library file will be affected, resulting in misjudgment. **Please ignore this error in the logic for official release**, and try to avoid displaying it in the UI.

Sample code

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

Triggering event callback

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

You can refer to the `EnginePollHelper.cpp` file in the demo.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
class ITMGContext {
protected:
    virtual ~ITMGContext() {}

public:
    virtual void Poll()= 0;
}
```

Sample code

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

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
ITMGContext int Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
ITMGContext int Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

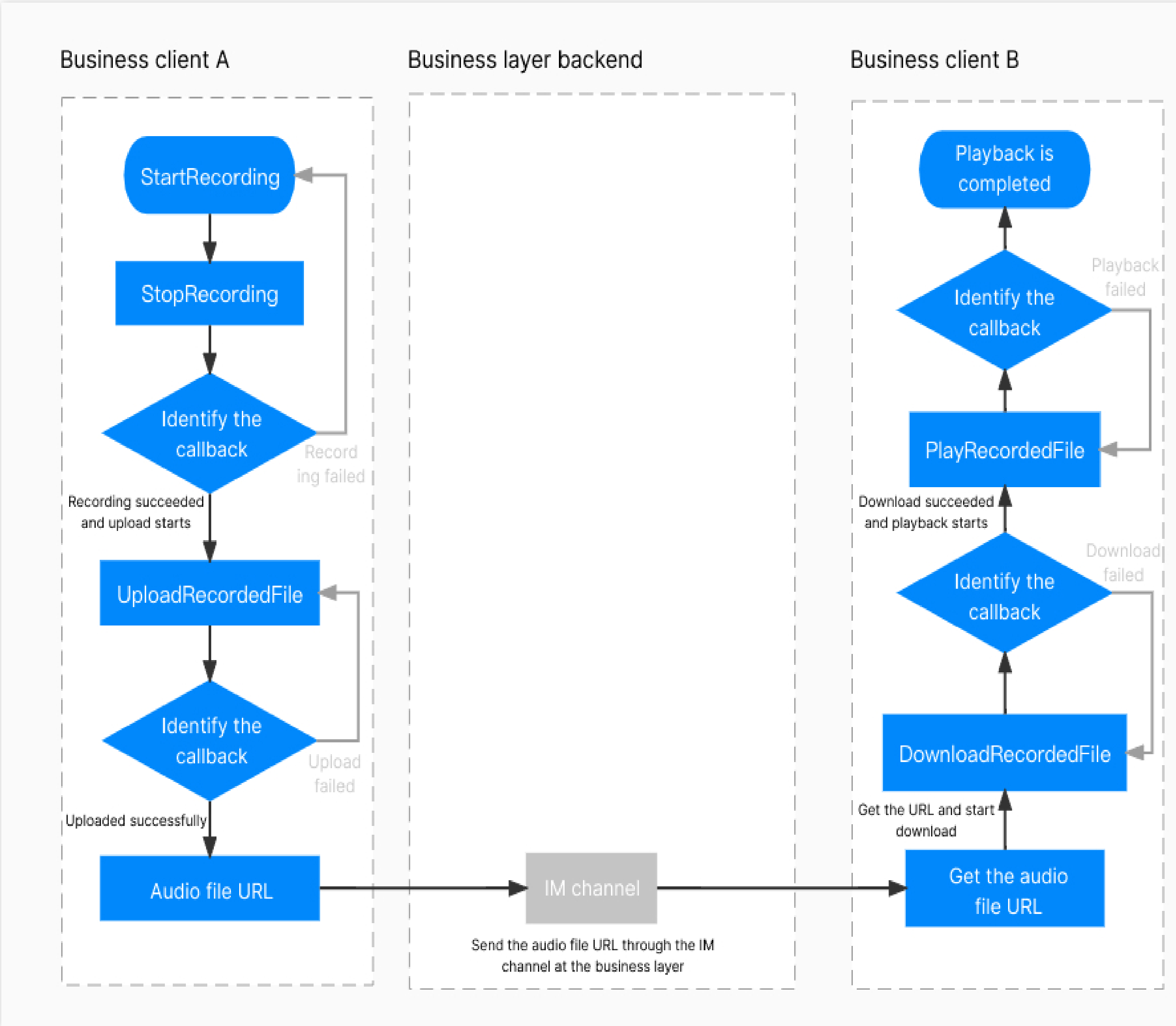
API prototype

```
ITMGContext int Uninit()
```

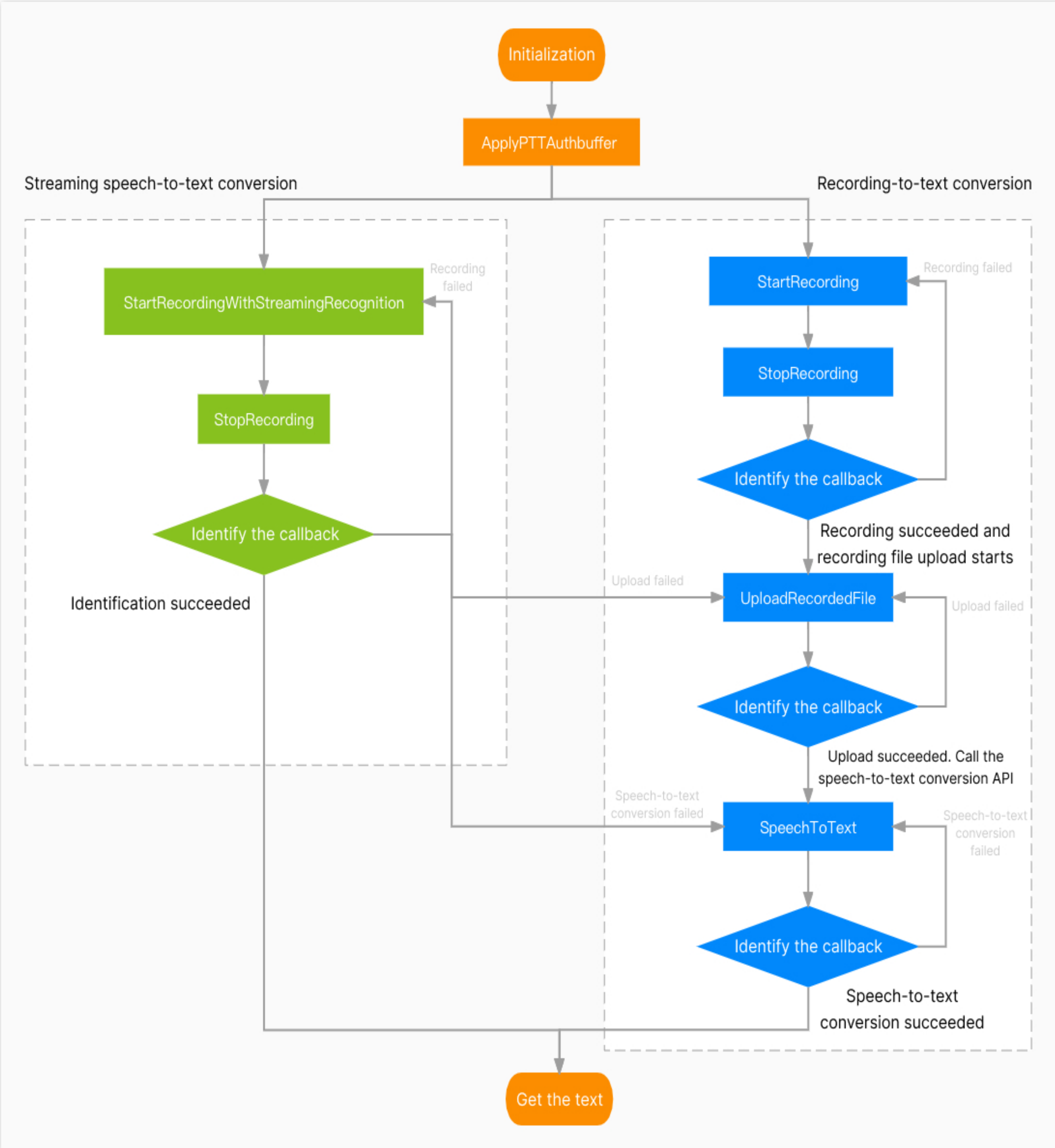
Voice Messaging and Speech-to-Text Services

Note:
The speech-to-text service consists of fast recording-to-text conversion and streaming speech-to-text conversion. You do not need to enter a voice chat room when using the voice message service.

The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum recording duration to 10 seconds, call the `SetMaxMessageLength` API to set it after initialization.



Flowchart for using the speech-to-text service



API	Description
GenAuthBuffer	Generates the local authentication key
ApplyPTTAuthbuffer	Initializes authentication
SetMaxMessageLength	Specifies the maximum length of voice message

Generating 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

```
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	`AppId` from the Tencent Cloud console.
strRoomID	const char*	Enter `null` or an empty string
strOpenID	const char*	User ID, which is the same as `openID` during initialization.
strKey	const char*	Permission key from the Tencent Cloud console.
strAuthBuffer	const char*	Returned `authbuff` .
bufferLength	int	Length of the `authbuff` passed in. 500 is recommended.

Application authentication

After the authentication information is generated, the authentication is assigned to the SDK.

API prototype

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

Parameter	Type	Description
authBuffer	const char*	Authentication
authBufferLen	int	Authentication length

Sample code

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

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

API prototype

```
ITMGPTT virtual int SetMaxMessageLength(int msTime)
```

Parameter	Type	Description
msTime	int	Audio duration in ms. Value range: 1000 < msTime <= 58000

Sample code

```
int msTime = 10000;
ITMGContextGetInstance()->GetPTT()->SetMaxMessageLength(msTime);
```

Streaming Speech Recognition

Voice message and speech-to-text APIs

API	Description
StartRecordingWithStreamingRecognition	Starts streaming recording
StopRecording	Stops recording

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** Stop recording.

API 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	const char*	Path of stored audio file
speechLanguage	const char*	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translateLanguage	const char*	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

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

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen on callback messages in the `OnEvent` notification, which is as detailed below:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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` notification based on the actual needs. The passed parameters include the following four messages.

Message Name	Description
result	Return code indicating whether 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

Note:

The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

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 conversion 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.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

If the error code 4098 is reported, see FAQs for solutions.

Sample code

```
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
isComplete)
{
                    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 result
Json error")).c_str());
                    }
                    else
                    {
                        if (isComplete) {
                            ::SetWindowText(m_EditUpload.GetSafeHwnd(),
MByteToWChar(root["file_id"].asString()).c_str());
                        }
                        else {
                            std::string isruning =
"STREAMINGRECOGNITION_IS_RUNNING";
                            ::SetWindowText(m_EditUpload.GetSafeHwnd(),
MByteToWChar(isruning).c_str());
                        }
                    }
}
```

Voice Message Recording

The recording process is as follows: start recording > stop recording > return recording callback > start the next recording.

Voice message and speech-to-text APIs

API	Description
StartRecording	Starts recording
PauseRecording	Pauses recording
ResumeRecording	Resumes recording

StopRecording	Stops recording
CancelRecording	Cancels recording

Starting recording

This API is used to start recording.

API prototype

```
ITMGPTT virtual int StartRecording(const char* fileDir)
```

Parameter	Type	Description
fileDir	const char*	Path of stored audio file

Sample code

```
char buffer[256]={0};
snprintf(buffer, sizeof(buffer), "%sunreal_ptt_local.file",
getFilePath().c_str());
ITMGContextGetInstance()->GetPTT()->StartRecording(buffer);
```

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 virtual int StopRecording()
```

Sample code

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

Callback for recording start

The recording start result will be returned through the callback.

To stop recording, call `StopRecording` . The callback for recording start will be returned after the recording is stopped.

Parameter	Type	Description

result	int32	0: recording is completed
filepath	FString	Path of stored recording file, which must be accessible and cannot be the `fileid`

Error codes

Error Code Value	Cause	Suggested Solution
4097	Parameter is empty.	Check whether the API parameters in the code are correct.
4098	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Ensure that the SDK recording feature is used at the right time.
4100	Audio data is not captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	Mic permission is required for using the SDK. To add the permission, please see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

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

```
                    // Process
                    break;
                    }
                }
}
```

Pausing recording

This API is used to pause recording. If you want to resume recording, please call the `ResumeRecording` API.

API prototype

```
ITMGPTT virtual int PauseRecording()
```

Sample code

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

Resuming recording

This API is used to resume recording.

API prototype

```
ITMGPTT virtual int ResumeRecording()
```

Sample code

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

Canceling recording

This API is used to cancel recording. **There is no callback after cancellation**.

API prototype

```
ITMGPTT virtual int CancelRecording()
```

Sample code

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

Voice Message Upload, Download, and Playback

API	Description
UploadRecordedFile	Uploads the audio file
DownloadRecordedFile	Downloads the audio file
PlayRecordedFile	Plays back the audio file
StopPlayFile	Stops playing back the audio file
GetFileSize	Gets audio file size
GetVoiceFileDuration	Gets the audio file duration

Uploading an audio file

This API is used to upload an audio file.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of uploaded audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->UploadRecordedFile(filePath);
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	int32	0: recording is completed
filepath	FString	Path of stored recording file
fileid	FString	File URL path

Error codes

Error Code Value	Cause	Suggested Solution
8193	An error occurred while accessing the file during upload.	Ensure the existence of the file and the validity of the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
8195	A network error occurred.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8200	No `appinfo` is set.	Check whether the `apply` API is called or whether the input parameters are empty.

Sample code

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

Downloading the audio file

This API is used to download an audio file.

API prototype

```
ITMGPTT virtual int DownloadRecordedFile(const char* fileId, const char*
filePath)
```

Parameter	Type	Description
fileId	const char*	URL path of file
filePath	const char*	Local path of saved file

Sample code

```
ITMGContextGetInstance()->GetPTT()->DownloadRecordedFile(fileID,filePath);
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	int32	0: recording is completed
filepath	FString	Path of stored recording file
fileid	FString	URL path of file, which will be retained on the server for 90 days

Error codes

Error Code Value	Cause	Suggested Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
12291	Network storage system exception.	The server failed to get the audio file. Check

		whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	Server file system error.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed during the process of getting the download parameters.	Check whether the device can access the internet.
12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.

Sample code

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

Playing back audio

This API is used to play back audio.

API prototype

```
ITMGPTT virtual int PlayRecordedFile(const char* filePath)
ITMGPTT virtual int PlayRecordedFile(const char* filePath, nt voiceType)
```

Parameter	Type	Description
filePath	const char*	Local audio file path

voicetype	int	Voice changer type. For more information, see Voice Changing Effects.

Error codes

Error Code Value	Cause	Suggested Solution
20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
ITMGContextGetInstance()->GetPTT()->PlayRecordedFile(filePath);
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameter includes `result` and `file_path` .

Parameter	Type	Description
code	int	0: playback is completed
filepath	FString	Path of stored recording file

Error codes

Error Code Value	Cause	Suggested Solution
20481	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	Parameter is empty.	Check whether the API parameters in the code are correct.
20484	Internal error.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
    switch (eventType) {
    ...
        else if (eventType == ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE) {
            int32 result = JsonObject->GetIntegerField(TEXT("result"));
            FString filepath = JsonObject->GetStringField(TEXT("file_path"));
            onPttPlayFileCompleted(result, filepath);
        }
    }
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

API prototype

```
ITMGPTT virtual int StopPlayFile()
```

Sample code

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

Getting audio file size

This API is used to get the size of an audio file.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->GetFileSize(filePath);
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

API prototype

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

Parameter	Type	Description
filePath	const char*	Path of audio file, which is a local path

Sample code

```
ITMGContextGetInstance()->GetPTT()->GetVoiceFileDuration(filePath);
```

Fast Recording-to-Text Conversion

API	Description
SpeechToText	Converts speech to text

Converting audio file to text

This API is used to convert a specified audio file to text.

API prototype

```
ITMGPTT virtual void SpeechToText(const char* fileID)
```

Parameter	Type	Description
fileID	const char*	Audio file URL

Sample code

```
ITMGContextGetInstance()->GetPTT()->SpeechToText(fileID);
```

Translating audio file into text in specified language

This API can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation.

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

API prototype

```
ITMGPTT virtual int SpeechToText(const char* fileID,const char* speechLanguage)
ITMGPTT virtual int SpeechToText(const char* fileID,const char*
speechLanguage,const char* translateLanguage)
```

Parameter	Type	Description
fileID	const char*	The URL of the audio file, which will be retained on the server for 90 days.
speechLanguage	const char*	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translatelanguage	const char*	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

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

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result` , `file_path` and `text` (recognized text).

Parameter	Type	Description
result	int32	0: recording is completed
fileid	FString	URL of recording file, which will be retained on the server for 90 days
text	FString	Converted text

Error codes

Error Code Value	Cause	Suggested Solution

32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	Network failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	Incorrect speech-to-text conversion parameter.	Check whether the API parameter `fileid` in the code is empty.
32785	Speech-to-text translation returned an error.	Error with the backend of voice messaging and speech-to-text feature. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

Sample code

```
void TMGTestScene::OnEvent(ITMG_MAIN_EVENT_TYPE eventType,const char* data){
        switch (eventType) {
            case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
            {
            // Process
            break;
                }
            ...
                    case ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE:
            {
```



```
                // Process
                break;
                }
            }
}
```

Voice Message Volume Level APIs

API	Description
GetMicLevel	Gets real-time mic volume level
SetMicVolume	Sets recording volume level
GetMicVolume	Gets recording volume level
GetSpeakerLevel	Gets real-time speaker volume
SetSpeakerVolume	Sets playback volume level
GetSpeakerVolume	Gets playback volume level

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetMicLevel()
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetMicLevel();
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT virtual int SetMicVolume(int vol)
```

Sample code

```
ITMGContextGetInstance()->GetPTT()->SetMicVolume(100);
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetMicVolume()
```

Sample code

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

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetSpeakerLevel()
```

Sample code

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

Setting the playback volume of voice message

This API is used to set the playback volume of voice message. Value range: 0-200.

API prototype

```
ITMGPTT virtual int SetSpeakerVolume(int vol)
```

Sample code

```
ITMGContextGetInstance()->GetPTT()->SetSpeakerVolume(100);
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice message. An int-type value will be returned. Value range: 0-200.

API prototype

```
ITMGPTT virtual int GetSpeakerVolume()
```

Sample code

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

Advanced APIs

Getting version number

This API is used to get the SDK version number for SDK usage analysis.

API prototype

```
ITMGContext virtual const char* GetSDKVersion()
```

Sample code

```
ITMGContextGetInstance()->GetSDKVersion();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
ITMGContext int SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL
levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	

		Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

`ITMG_LOG_LEVEL` description:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContextGetInstance()->SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

OS	Path
Windows	%appdata%\\Tencent\\GME\\ProcessName
iOS	Application/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/Documents
Android	/sdcard/Android/data/xxx.xxx.xxx/files
Mac	/Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents

API prototype

```
ITMGContext virtual int SetLogPath(const char* logDir)
```

Parameter	Type	Description
logDir	const char*	Path

Sample code

```
cosnt char* logDir = ""// Set a path by yourself
ITMGContext* context = ITMGContextGetInstance();
context->SetLogPath(logDir);
```

Callback Messages

Message	Description	Parameter
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room	result; error
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected for network or other reasons	result; error
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Room members were updated	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	The reconnection to the room started	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	The reconnection to the room succeeded	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status was	result; error_info;

	changed	sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker device was changed	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A new speaker device was added	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker device was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A new mic device was added	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic device was lost	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic device was changed	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	The room quality changed	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Voice message recording was completed	result; file_

ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Voice message upload was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Voice message download was completed	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Voice message playback was completed	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast speech-to-text conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion was completed	result; file_ text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	Streaming speech-to-text conversion is in progress	result; file_ text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed	result; text;file_id

SDK for iOS

Integrating SDK

Last updated：2024-01-18 15:11:45

This document describes how to integrate GME SDK into an iOS project so that the iOS developers can easily debug and integrate the APIs for Game Multimedia Engine (GME).

SDK preparations

1. Download the applicable demo and SDK. For more information, see SDK Download Guide.
2. Decompress the obtained SDK resources.
3. The folder contains:

GMESDK.framework: Native resource related to iOS development.

libGMESDK.a: Unity resource related to iOS development (you can ignore this file if you don't use the Unity engine for development).

Note:

You can run the SDK on iOS 9.0 or later.

Voice Chat

Last updated：2024-01-18 15:11:45

This document describes how to integrate with and debug GME client APIs for the voice chat feature for iOS.

Key Considerations for Using GME

GME provides the real-time voice chat service, voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

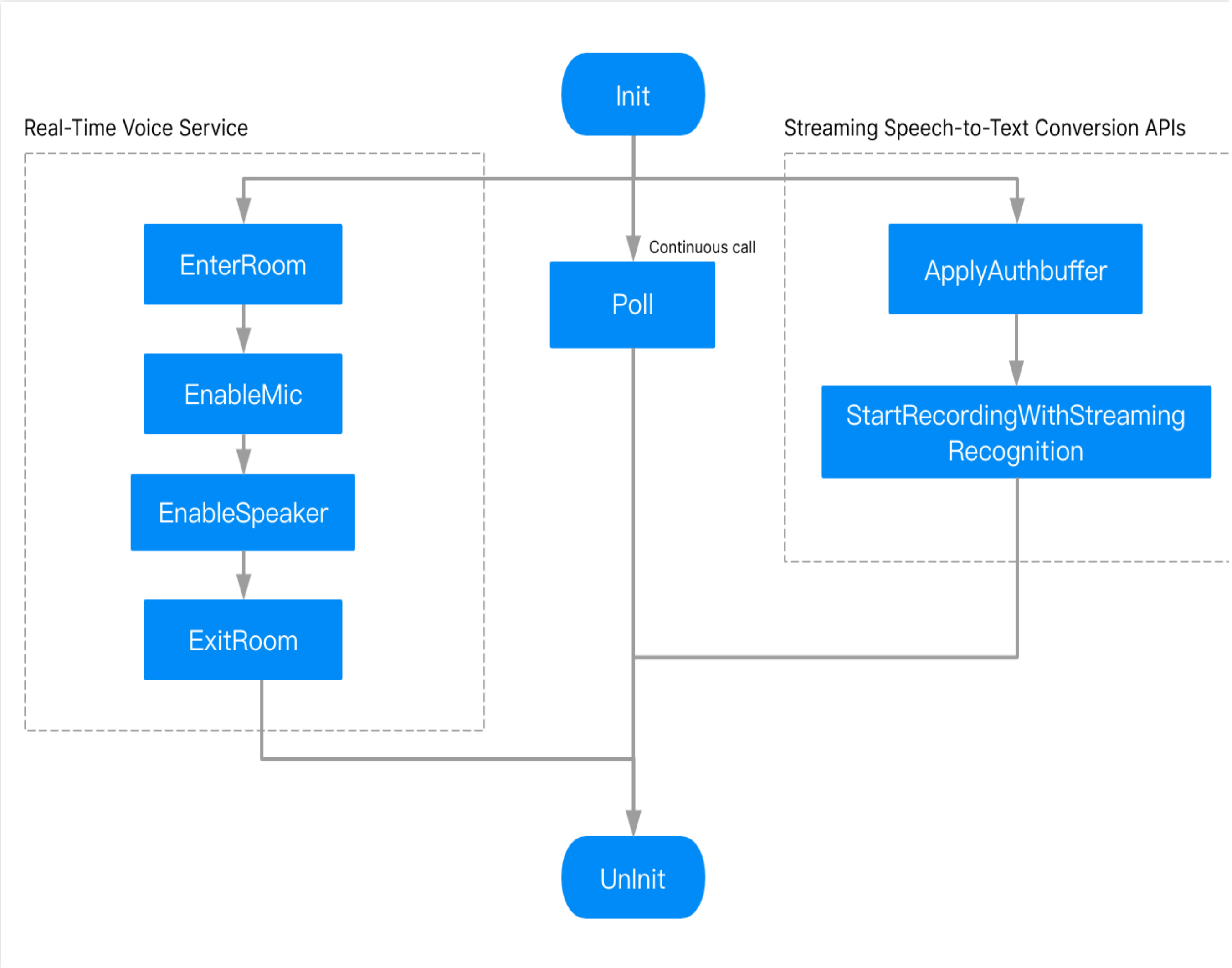
Notes

You have created a GME application and obtained the `AppID` and `Key` of the SDK as instructed in Activating Services.
You have **activated the voice chat, voice messaging, and speech-to-text services of GME**. For more information, see Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `QAVError.OK` will be returned with the value being `0` .
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error codes, see Error Codes.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initializing GME
2. Calling Poll periodically to trigger callbacks
3. Entering a voice chat room
4. Turning on the mic
5. Turning on the speaker
6. Exiting the voice chat room
7. Uninitializing GME

APIs

```
@class ITMGRoom;//Room APIs
@class ITMGAudioCtrl;//Audio APIs
@class ITMGAudioEffectCtrl;//Sound effect and accompaniment APIs
```

Core APIs

API	Description
InitEngine	Initializes GME.
Poll	Triggers an event callback.
Pause	Pauses the system.
Resume	Resumes the system.
Uninit	Uninitializes GME.
SetDefaultAudienceAudioCategory	Sets the audio of the iOS device.

Note:

If you need to switch the account, call `UnInit` to uninitialize the SDK. No fees are incurred for calling the `InitEngine` API.

Imported header files

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

Getting singleton

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

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

Sample code

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

Setting callbacks

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.

Sample code

`ITMGDelegate` is used for declaration.

```
@interface TMGDemoViewController ()<ITMGDelegate>{}
```

```
ITMGDelegate < NSObject >

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

The API callback messages is processed in `OnEvent` . For the message type, see `ITMG_MAIN_EVENT_TYPE` . The message content is a dictionary for parsing the API callback contents.

API prototype

```
- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary*)data;
```

Sample code

```
//TMGRealTimeViewController.m
TMGRealTimeViewController ()< ITMGDelegate >


- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data {
    NSString *log = [NSString stringWithFormat:@"OnEvent:%d,data:%@",
(int)eventType, data];
    [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:(%@)", result, error_info]];

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

// Refer to DispatchCenter.h and DispatchCenter.m
```

Initializing the SDK

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

API prototype

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

Parameter	Type	Description
sdkAppId	String	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
OpenId	String	`openID` can only be in Int64 type, which is passed in after being converted to a `const char*` . 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.

Returned values

Returned Value	Description
QAV_OK= 0	The SDK was initialized successfully.
QAV_ERR_SDK_NOT_FULL_UPDATE= 7015	Checks whether the SDK file is complete. We recommend that you delete it and then import it again.

Notes on 7015 error code

The error code 7015 is identified based on the MD5 value. If this error is reported during integration, check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party enhancement, the Unity packaging mechanism, and other factors, the MD5 value of the library file will be affected, resulting in misjudgment. Therefore, **ignore this error in the logic for official releases**, and avoid displaying it on the UI.

Sample code

```
_openId = _userIdText.text;
_appId = _appIdText.text;
int result = 0;
// After the user consents to the application's privacy policy, initialize the SDK at an appropriate time based on the application features
//result = 0: The user consents to the application's privacy policy
```

```
//result = 1: The user does not consent to the application's privacy policy
// If the user does not consent to the privacy policy, change `ret` to a value
other than 0
if (result == 0) {
     [[ITMGContext GetInstance] InitEngine:SDKAPPID openID:_openId];}
     else{
     log = [NSString stringWithFormat:@"The user does not consent to the
application's privacy policy"];
}
```

Triggering event callback

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

Refer to the `EnginePollHelper.m` file in SDK Download Guide.

Call the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
-(void)Poll;
```

Sample code

```
[[ITMGContext GetInstance] Poll];
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
-(QAVResult)Pause;
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
-(QAVResult)Resume;
```

Uninitializing SDK

This API is used to uninitialize the SDK. **If the game account is bound to** `openid` , switching game account requires uninitializing GME and then using the new `openid` to initialize again.

Note:

If the end user revokes the permission granted to the SDK to process the personal information, you can call the `Uninit` API to stop using the SDK features and stop collecting and close the user data used by the features.

API prototype

```
-(int)Uninit;
```

Sample code

```
[[ITMGContext GetInstance] Uninit];
```

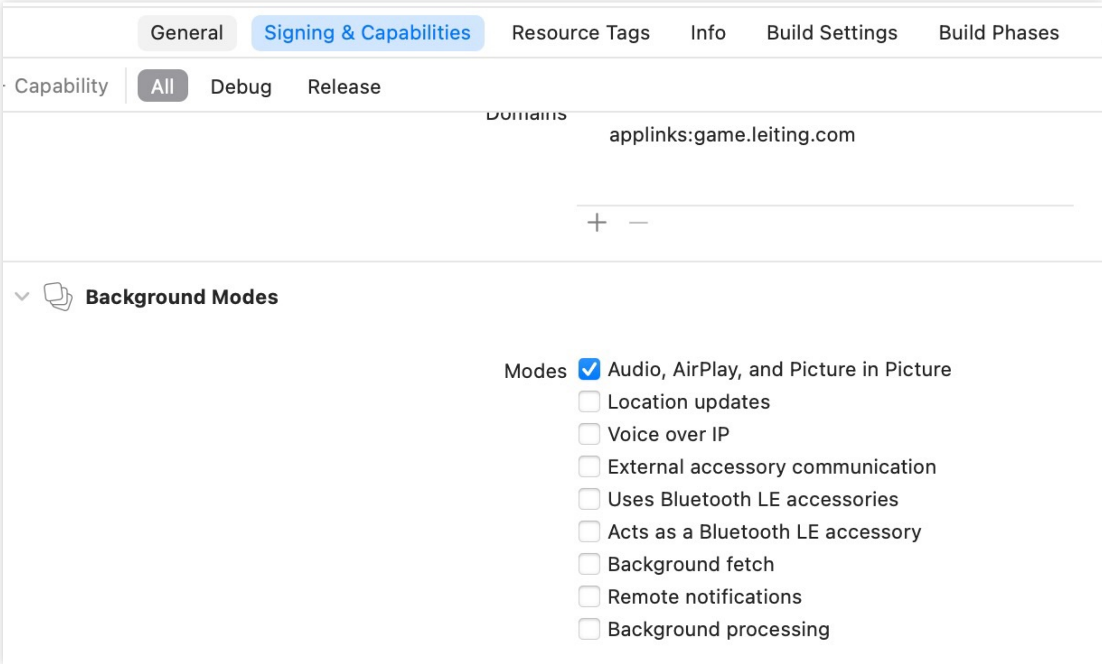
Audio settings for iOS device

This API is used to set the audio playback in the background, and the GME audio not to be affected by the mute switch or lock screen. For example, when the notification center or control center is opened, you can still receive and play back the GME audio. You need to call this API before room entry.

Meanwhile, you should pay attention to the following two points in the application:

Audio engine capture and playback are not paused when the application is switched to the background (i.e., `PauseAudio`).

You need to add at least `key:Required background modes` and `string:App plays audio or streams audio/video using AirPlay` to the `Info.plist` of the application.



Note:

It is recommended that developers call this API to set the audio.

Function prototype

```
-(QAVResult)SetDefaultAudienceAudioCategory:(ITMG_AUDIO_CATEGORY)audioCategory;
```

Type	Value	Description
ITMG_CATEGORY_AMBIENT	0	Audio is not played back in the background (default value).
ITMG_CATEGORY_PLAYBACK	1	Audio is played back in the background.

The specific implementation is to modify `kAudioSessionProperty_AudioCategory` . For more information, see Apple documentation.

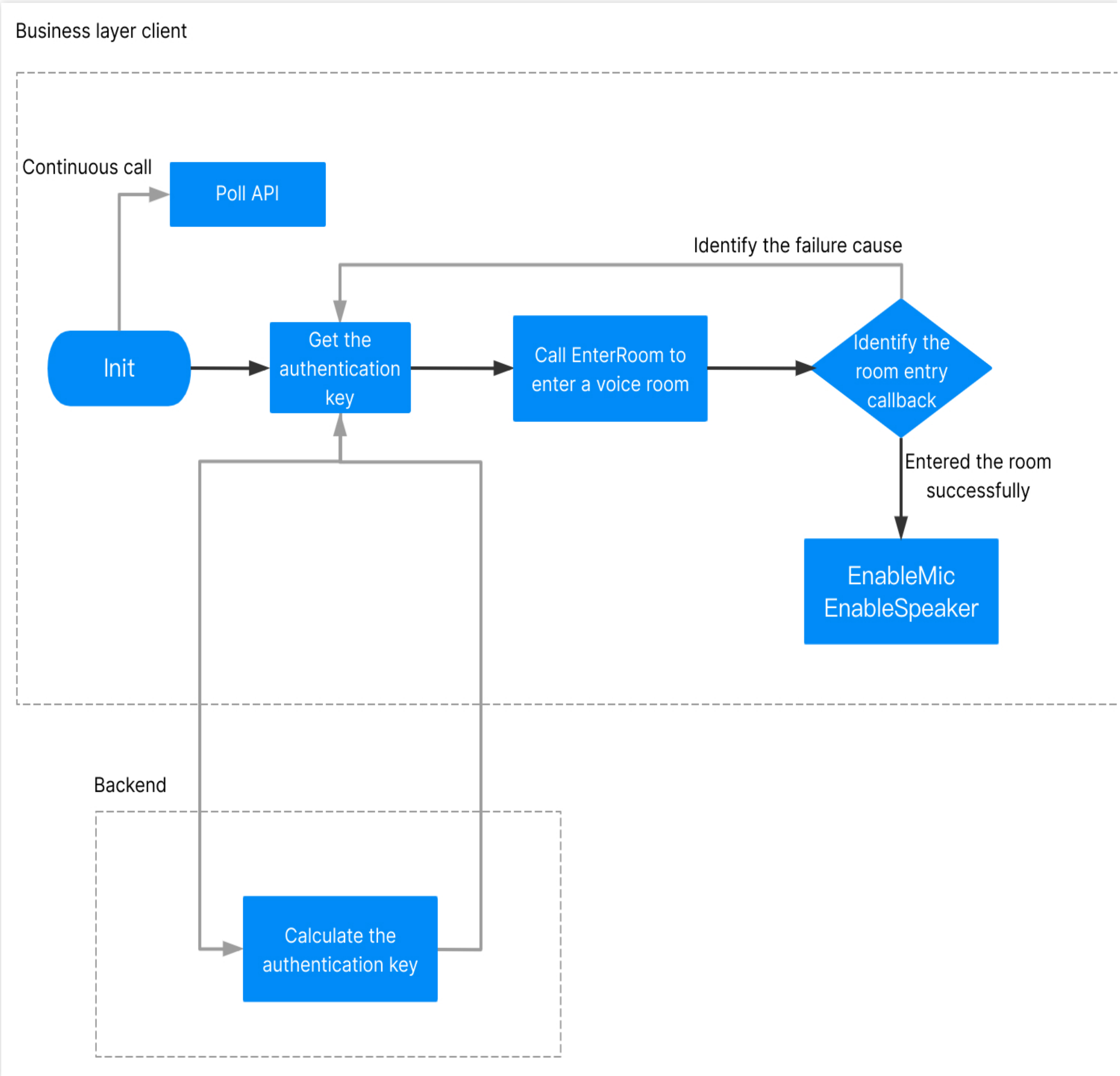
Sample code

```
[[ITMGContext
GetInstance]SetDefaultAudienceAudioCategory:ITMG_CATEGORY_AMBIENT];
```

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, see Sound and Audio.



API	Description
GenAuthBuffer	Calculates the local authentication key.
EnterRoom	Enters a room.
ExitRoom	Exits a room.
IsRoomEntered	Determines whether room entry is successful.
SwitchRoom	Switches the room quickly.

StartRoomSharing	Cross-room Co-anchoring

Local authentication key calculation

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

API prototype

```
@interface QAVAuthBuffer : NSObject
+ (NSData*) GenAuthBuffer:(unsigned int)appId roomId:(NSString*)roomId openID:
(NSString*)openID key:(NSString*)key;
+ @end
```

Parameter	Type	Description
appId	unsigned int	`AppId` from the Tencent Cloud console
roomId	NSString *	Room ID, which can contain up to 127 characters.
openID	NSString *	User ID, which is the same as `openID` during initialization.
key	NSString *	Permission key from the Tencent Cloud console.

Sample code

```
#import "GMESDK/QAVAuthBuffer.h"
NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:SDKAPPID3RD.intValue
roomId:_roomId openID:_openId key:AUTHKEY];
```

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.

Note:

If the room entry callback result is `0`, the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.

The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will become smooth sound quality. Only when a member in the room calls the `ChangeRoomType` API, the audio type of the room will be changed.

Function prototype

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

Parameter	Type	Description
roomId	NSString *	Room ID, which can contain up to 127 characters.
roomType	int	Room type. We recommend that you enter `ITMG_ROOM_TYPE_FLUENCY` . For more information on room audio types, see Sound Quality.
authBuffer	NSData *	Authentication key

Sample code

```
[[ITMGContext GetInstance] EnterRoom:_roomId roomType:_roomType
authBuffer:authBuffer];
```

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

Will the billing continue if the client is disconnected from the server when using the voice chat?

Sample code

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

```
-(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_ENTER_ROOM:
        {
            int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
            NSString* error_info = [data objectForKey:@"error_info"];
                //Receive the event of successful room entry
        }
            break;
    }
```

```
}
```

Data details

Message	Data	Example
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnected callback prompt `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code	Cause and Suggested Solution
7006	Authentication failed. Causes: `AppID` doesn't exist or is incorrect. An error occurred while authenticating `authbuff` . Authentication expired. `OpenId` is invalid.
7007	The user was already in another room.
1001	The user was already in the process of entering a room but repeated this operation. We recommend that you not call the room entry API until the room entry callback is returned.
1003	The user was already in the room and called the room entry 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.

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API; instead, you can directly call the `EnterRoom` API.

API prototype

```
-(int)ExitRoom
```

Sample code

```
[[ITMGContext GetInstance] 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` .

Sample code

```
-(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;
    }
}
```

Data details

Message	Data	Example
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"","result":0}

Determining whether user has entered room

This API is used to determine whether the user has entered a room. A value in bool type will be returned. Do not call this API during room entry.

API prototype

```
-(BOOL)IsRoomEntered;
```

Sample code

```
[[ITMGContext GetInstance] IsRoomEntered];
```

Switching room

User can call this API to quickly switch the voice chat room after entering the room. After the room is switched, the device is not reset, that is, if the microphone is already enabled in this room, the microphone will keep enabled after the room is switched.

The callback for quickly switching rooms is `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM` , and the fields are `error_info` and `result` .

API prototype

```
-(int) SwitchRoom:(NSString *)roomID  authBuffer:(NSData*)authBuffer;
```

Type descriptions

Parameter	Type	Description
targetRoomID	NSString *	ID of the room to enter
authBuffer	NSData*	Generates a new authentication key with the ID of the room to enter

Callback sample code

```
- (IBAction)swichRoom:(id)sender {
    NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:_appId.intValue
roomID:_roomIdText.text openID:_openId key:_key];
    [[[ITMGContext GetInstance]GetRoom]SwitchRoom:_roomIdText.text
authBuffer:authBuffer];
}

-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSString* log =[NSString stringWithFormat:@"OnEvent:%d,data:%@",
(int)eventType, data];
    [self showLog:log];
    NSLog(@"====%@====",log);
    switch (eventType) {
    case ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM:
        {
            int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
            NSString* log = nil;
            if (result == QAV_OK) {
```

```
                log = [NSString stringWithFormat:@"switch room success."];
            } else {
                log = [NSString stringWithFormat:@"switch room failed."];
            }
            [self showLog:log];
            break;
        }
    }
}
```

Cross-room mic connection

Call this API to connect the microphones across rooms after the room entry. After the call, the local user can communicate with the target OpenID user in the target room. The target room should be of the same type as the local room.

Example

User a is in room A, user b is in room B, and user a can talk with b through the cross-room API. When user c in room A speaks, users b and d in room B cannot hear. User c in room A can hear only the voice in room A and the voice of user b in room B but not other users in room B.

API prototype

```
-(int) StartRoomSharing:(NSString *)targetRoomID targetOpenID:(NSString *)targetOpenID authBuffer:(NSData*)authBuffer;

-(int) StopRoomSharing;
```

Type descriptions

Parameter	Type	Description
targetRoomID	NSString *	ID of the room to connect mic
targetOpenID	NSString *	Target `OpenID` to connect mic
authBuffer	NSData*	Reserved flag. You just need to enter NULL.

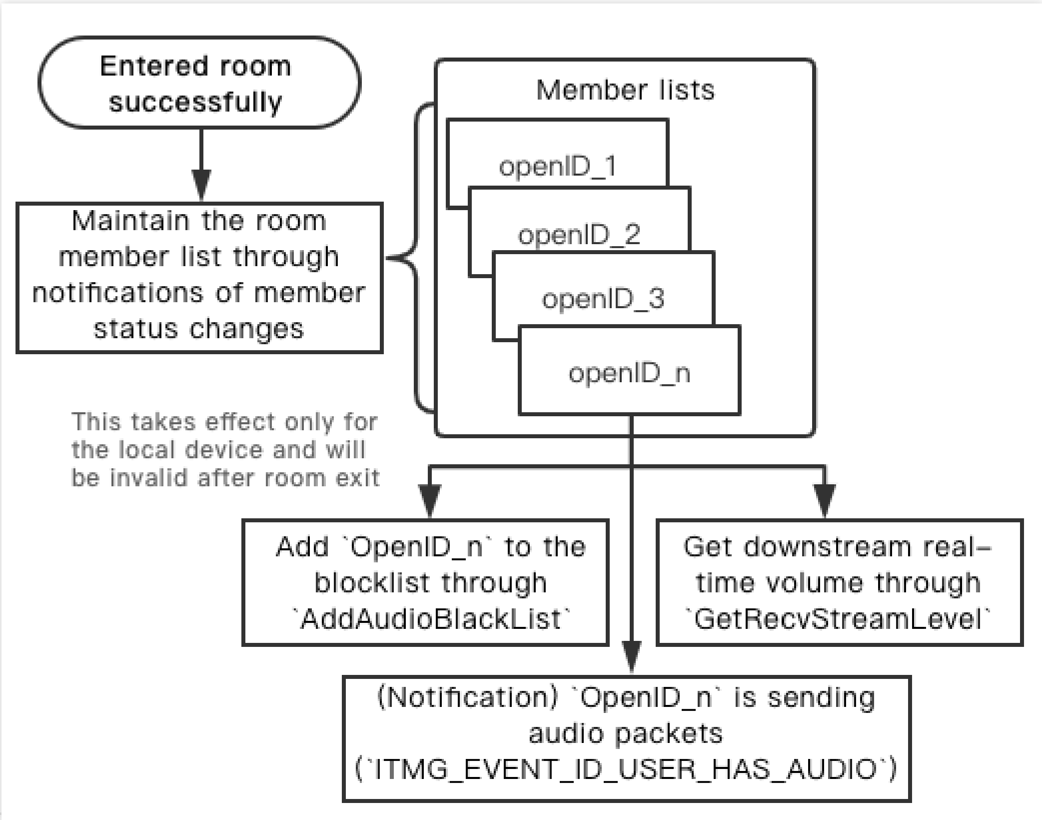
Sample code

```
- (IBAction)shareRoom:(id)sender {
    if(_shareRoomSwitch.isOn){
        [[[ITMGContext GetInstance]GetRoom]StartRoomSharing:_shareRoomID.text targetOpenID:_shareOpenID.text authBuffer:NULL];
```

```
    }else{
        [[[ITMGContext GetInstance]GetRoom]StopRoomSharing];
    }
}
```

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed.
AddAudioBlackList	Mutes a member in the room.
RemoveAudioBlackList	Unmutes a member.

Notifications of member room entry and speaking status

This API is used to obtain the user speaking in the room and display it in the UI, and to send a notification when someone entering or exiting the room.

Notification for this event will be sent only when the status changes. To get member status in real time, cache the notification when it is received at a higher layer. The event message is `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` , where the data contains `event_id` and `user_list` . The event message will be identified in the `OnEvent` function.
Notifications for audio events are subject to a threshold, and a notification will be sent only when this threshold is exceeded. The notification "A member has stopped sending audio packets" will be sent if no audio packets are received in more than two seconds.

event_id	Description	Maintenance
ITMG_EVENT_ID_USER_ENTER	Return the `openid` of the member entering the room.	Member list
ITMG_EVENT_ID_USER_EXIT	Return the `openid` of the member exiting the room.	Member list
ITMG_EVENT_ID_USER_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
ITMG_EVENT_ID_USER_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    ITMG_EVENT_ID_USER_UPDATE event_id=((NSNumber*)[data
objectForKey:@"event_id"]).intValue;
    NSMutableArray* uses = [NSMutableArray arrayWithArray: [data
objectForKey:@"user_list"]];
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_USER_UPDATE:
        {
        // Process
        //Parse the parameter to get `event_id` and `user_list`
            switch (eventID)
             {
             case ITMG_EVENT_ID_USER_ENTER:
                  // A member enters the room
                  break;
             case ITMG_EVENT_ID_USER_EXIT:
                  // A member exits the room
                break;
```

```
                case ITMG_EVENT_ID_USER_HAS_AUDIO:
                    // A member sends audio packets
                    break;
                case ITMG_EVENT_ID_USER_NO_AUDIO:
                    // A member stops sending audio packets
                    break;
             }
          break;
          }
      }
}
```

Data details

Message	Data	Example
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	event_id; user_list	{"event_id":0,"user_list":""}

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B.

If B blocks neither A nor C, B can hear both of them.

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
ITMGContext GetAudioCtrl -(QAVResult)AddAudioBlackList:(NSString*)openID;
```

Parameter	Type	Description
openId	NSString	`openid` of the user to be blocked

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ] AddAudioBlackList[id]];
```

Unmuting

This API is used to remove an ID from the audio data blacklist. A returned value of 0 indicates the call is successful.

API prototype

```
-(QAVResult)RemoveAudioBlackList:(NSString*)openID;
```

Parameter	Type	Description
openId	NSString	User `openid` to be unblocked

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ] RemoveAudioBlackList[openId]];
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.
When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend that you call the `EnableMic` or `EnableSpeaker` API.
When the user enters a voice chat room, enabling/disabling a capturing device will restart both capturing and playback devices. If the application is playing back background music, it will also be interrupted. Playback won't be interrupted if the mic is enabled/disabled through control of upstreaming/downstreaming. **Calling method: Call** `EnableAudioCaptureDevice(true)` and `EnableAudioPlayDevice(true)` once after room entry, and call `EnableAudioSend/Recv` to send/receive audio streams when Enable/Disable Mic is clicked.

API	Description
EnableMic	Enables/Disables the mic.
GetMicState	Gets the mic status.
EnableAudioCaptureDevice	Enables/Disables the capturing device.
IsAudioCaptureDeviceEnabled	Gets the capturing device status.
EnableAudioSend	Enables/Disables audio upstreaming.
IsAudioSendEnabled	Gets the audio upstreaming status.
GetMicLevel	Gets the real-time mic volume level.
GetSendStreamLevel	Gets the real-time audio upstreaming volume level.
SetMicVolume	Sets the mic volume level.

GetMicVolume	Gets the mic volume level.

Enabling or disabling mic

This API is used to enable/disable the mic. The mic and speaker are not enabled by default after room entry. `EnableMic` is equivalent to using `EnableAudioCaptureDevice` and `EnableAudioSend` together. **If accompaniment is used, call this API as instructed in** Accompaniment in Voice Chat.

API prototype

```
-(QAVResult)EnableMic:(BOOL)enable;
```

Parameter	Type	Description
enable	boolean	To enable the mic, set this parameter to `YES` . To disable the mic, set this parameter to `NO` .

Sample code

```
// Turn on mic
[[[ITMGContext GetInstance] GetAudioCtrl] EnableMic:YES];
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
-(int)GetMicState;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetMicState];
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.
This API can only be called after room entry. The device will be disabled automatically after room exit.
Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
-(QAVResult)EnableAudioCaptureDevice:(BOOL)enabled;
```

Parameter	Type	Description
enabled	BOOL	To enable the capturing device, set this parameter to `YES` . To disable the capturing device, set this parameter to `NO` .

Sample code

```
// Enable capturing device
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioCaptureDevice:enabled];
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
-(BOOL)IsAudioCaptureDeviceEnabled;
```

Sample code

```
BOOL IsAudioCaptureDevice = [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioCaptureDeviceEnabled];
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain muted. For more information on how to enable/disable the capturing device, see the `EnableAudioCaptureDevice` API.

API prototype

```
-(QAVResult)EnableAudioSend:(BOOL)enable;
```

Parameter	Type	Description
enable	BOOL	To enable audio upstreaming, set this parameter to `YES` . To disable audio upstreaming, set this parameter to `NO` .

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioSend:enabled];
```

Getting audio upstreaming status

This API is used to get the status of audio upstreaming.

API prototype

```
-(BOOL)IsAudioSendEnabled;
```

Sample code

```
BOOL IsAudioSend = [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioSendEnabled];
```

Getting the real-time mic volume

This API is used to get the real-time mic volume. An int-type value in the range of 0-100 will be returned. It is recommended to call this API once every 20 ms.

API prototype

```
-(int)GetMicLevel;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetMicLevel];
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume. An int-type value in the range of 0-100 will be returned.

API prototype

```
-(int)GetSendStreamLevel();
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSendStreamLevel];
```

Setting the mic software volume

This API is used to set the mic volume level. The corresponding parameter is `volume` , which is equivalent to attenuating or gaining the captured sound.

API prototype

```
-(QAVResult)SetMicVolume:(int) volume;
```

Parameter	Type	Description
volume	int	Value range: 0-200. Default value: `100` . `0` indicates that the audio is muted, while `100` indicates that the volume level remains unchanged.

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] SetMicVolume:100];
```

Getting the mic software volume

This API is used to obtain the microphone volume. An "int" value is returned. Value 101 represents API SetMicVolume has not been called.

API prototype

```
-(int) GetMicVolume;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetMicVolume];
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker.
GetSpeakerState	Gets the speaker status.
EnableAudioPlayDevice	Enables/Disables the playback device.
IsAudioPlayDeviceEnabled	Gets the playback device status.

EnableAudioRecv	Enables/Disables audio downstreaming.
IsAudioRecvEnabled	Gets the audio downstreaming status.
GetSpeakerLevel	Gets the real-time speaker volume level.
GetRecvStreamLevel	Gets the real-time downstreaming audio levels of other members in the room.
SetSpeakerVolume	Sets the speaker volume level.
GetSpeakerVolume	Gets the speaker volume level.

Enabling or disabling speaker

This API is used to enable/disable the speaker. `EnableSpeaker` is equivalent to using `EnableAudioPlayDevice` and `EnableAudioRecv` together. **If accompaniment is used, call this API as instructed in** Accompaniment in Voice Chat.

API prototype

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

Parameter	Type	Description
enable	boolean	To disable the speaker, set this parameter to `NO` . To enable the speaker, set this parameter to `YES` .

Sample code

```
// Turn on the speaker
[[[ITMGContext GetInstance] GetAudioCtrl] EnableSpeaker:YES];
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

Function prototype

```
-(int)GetSpeakerState;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSpeakerState];
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

API prototype

```
-(QAVResult)EnableAudioPlayDevice:(BOOL)enabled;
```

Parameter	Type	Description
enabled	BOOL	To disable the playback device, set this parameter to `NO` . To enable the playback device, set this parameter to `YES` .

Sample code

```
// Enable the playback device
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioPlayDevice:enabled];
```

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
-(BOOL)IsAudioPlayDeviceEnabled;
```

Sample code

```
BOOL IsAudioPlayDevice =  [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioPlayDeviceEnabled];
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain muted. For more information on how to enable/disable the playback device, see the `EnableAudioPlayDevice` API.

API prototype

```
-(QAVResult)EnableAudioRecv:(BOOL)enabled;
```

Parameter	Type	Description

enabled	BOOL	To enable audio downstreaming, set this parameter to `YES` . To disable audio downstreaming, set this parameter to `NO` .

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioRecv:enabled];
```

Getting audio downstreaming status

This API is used to get the status of audio downstreaming.

API prototype

```
-(BOOL)IsAudioRecvEnabled;
```

Sample code

```
BOOL IsAudioRecv = [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioRecvEnabled];
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume. An int-type value will be returned to indicate the volume. It is recommended to call this API once every 20 ms.

API prototype

```
-(int)GetSpeakerLevel;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSpeakerLevel];
```

Getting the real-time downstreaming audio levels of other members in room

This API is used to get the real-time audio downstreaming volume of other members in the room. An int-type value will be returned. Value range: 0-200.

API prototype

```
-(int)GetRecvStreamLevel:(NSString*) openID;
```

Parameter	Type	Description
openID	NSString	`openId` of another member in the room

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetRecvStreamLevel:(NSString*)
openId];
```

Dynamically setting the volume of a member of the room

This API is used to set the volume of a member in the room. It takes effect only on the local.

API prototype

```
-(int) SetSpeakerVolumeByOpenID:(NSString *)openId volume:(int)volume;
```

Parameter	Type	Description
openId	String *	`OpenID` of the target user
volume	int	Percentage. Recommended value range: 0-200. Default value: `100` .

Getting volume percentage

Call this API to get the volume set by SetSpeakerVolumeByOpenID

API prototype

```
-(int) GetSpeakerVolumeByOpenID:(NSString *)openId;
```

Returned values

API returns volume percentage set by OpenID, where 100 is by default.

Setting the speaker volume

This API is used to set the speaker volume.
The corresponding parameter is volume. 0 indicates that the audio is muted, while 100 indicates that the volume remains unchanged. The default value is 100.

Function prototype

```
-(QAVResult)SetSpeakerVolume:(int)vol;
```

Parameter	Type	Description
vol	int	Volume level. Value range: 0–200.

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] SetSpeakerVolume:100];
```

Getting the speaker volume

This API is used to get the speaker volume. An int-type value will be returned to indicate the volume. 101 indicates that the `SetSpeakerVolume` API has not been called.
"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

Function prototype

```
-(int)GetSpeakerVolume;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSpeakerVolume];
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

Function prototype

```
-(QAVResult)EnableLoopBack:(BOOL)enable;
```

Parameter	Type	Description
enable	boolean	Specifies whether to enable the in-ear monitoring.

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] EnableLoopBack:YES];
```

Modifying user's room audio type

This API is used to modify a user's room audio type. For the result, please see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE`. The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

Function prototype

```
-(int)ChangeRoomType:(int)nRoomType;
```

Parameter	Type	Description
nRoomType	int	Room type to be switched to. For room audio types, see the `EnterRoom` API.

Sample code

```
[[[ITMGContext GetInstance]GetRoom ]ChangeRoomType:_roomType];
```

Getting user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, please see the `EnterRoom` API.

Function prototype

```
-(int)GetRoomType;
```

Sample code

```
[[[ITMGContext GetInstance]GetRoom ]GetRoomType];
```

Callback for modifying the room type

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result`, `error_info`, and `new_room_type`. The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description

ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	The existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	A user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).
ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	A user is already in the room, and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	A room member calls the `ChangeRoomType` API to request a change of the room audio type.

Sample code

```
-(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_CHANGE_ROOM_TYPE:
            NSLog(@"ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE:%@ ",data);
            int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
            int newRoomType = ((NSNumber*) [data
objectForKey:@"new_room_type"]).intValue;
            int subEventType = ((NSNumber*) [data
objectForKey:@"sub_event_type"]).intValue;
     }
}
```

Data details

Message	Data	Exa
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result;error_info;new_room_type;subEventType	{"er

The monitoring event of room call quality

The message for quality monitoring event triggered once every two seconds after room entry is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which represent the following information.

This API is used to monitor the network quality. If the user's network is poor, the business layer will remind the user to switch to a better network through the UI.

Parameter	Type	Description

weight	int	Value range: 1–50. `50` indicates excellent sound quality, `1` indicates very poor (barely usable) sound quality, and `0` represents an initial meaningless value. Generally, if the value is below 30, you can remind users that the network is poor and recommend them to switch the network.
Loss	double	Upstream packet loss rate
Delay	int	Voice chat delay in ms

Getting version number

This API is used to get the SDK version number for analysis.

Function prototype

```
-(NSString*)GetSDKVersion;
```

Sample code

```
[[ITMGContext GetInstance] GetSDKVersion];
```

Checking mic permission

This API is used to return the mic permission status.

Function prototype

```
-(ITMG_RECORD_PERMISSION)CheckMicPermission;
```

Parameter description

Parameter	Value	Description
ITMG_PERMISSION_GRANTED	0	The mic permission is granted.
ITMG_PERMISSION_Denied	1	Microphone disabled.
ITMG_PERMISSION_NotDetermined	2	No authorization box has been popped up to request the permission.
ITMG_PERMISSION_ERROR	3	An error occurred while calling the API.

Sample code

```
[[ITMGContext GetInstance] CheckMicPermission];
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

Function prototype

```
-(void)SetLogLevel:(ITMG_LOG_LEVEL)levelWrite (ITMG_LOG_LEVEL)levelPrint;
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to log. Default value: `TMG_LOG_LEVEL_INFO` .
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: `TMG_LOG_LEVEL_ERROR` .

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
[[ITMGContext GetInstance] SetLogLevel:TMG_LOG_LEVEL_INFO TMG_LOG_LEVEL_INFO];
```

Setting the log printing path

This API is used to set the log printing path, and needs to be called before initialization. The default path is `Application/********-****-****-****-************/Documents` .

Function prototype

```
-(void)SetLogPath:(NSString*)logDir;
```

Parameter	Type	Description
logDir	NSString	Path

Sample code

```
[[ITMGContext GetInstance] SetLogPath:Path];
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

Function prototype

```
-(NSString*)GetQualityTips;
```

Sample code

```
[[[ITMGContext GetInstance]GetRoom ] GetQualityTips];
```

Callback Messages

Message list

Message	Description
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room.
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room.
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected due to a network or another issue.
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room type changed.
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Room members were updated.
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	The room quality changed.

Data list

Message	Data	Example
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"",
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"",
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"w
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result; error_info; sub_event_type; new_room_type	{"error_info":"",
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	result; error_info	{"deviceID":"{0 dd00542b47ae Audio)","error_
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	result; error_info	{"deviceID":"{0 dd00542b47ae Audio)","error_
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	result; error_info	{"deviceID":"{0 7e454093f229 Audio)","error_
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	result; error_info	{"deviceID":"{0 7e454093f229 Audio)","error_
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	user_list; event_id	{"event_id":1,"u
ITMG_MAIN_EVENT_TYPE_NUMBER_OF_USERS_UPDATE	AllUser; AccUser; ProxyUser	{"AllUser":3,"A
ITMG_MAIN_EVENT_TYPE_NUMBER_OF_AUDIOSTREAMS_UPDATE	AudioStreams	{"AudioStream
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	weight; loss; delay	{"weight":5,"los

Speech-to-Text Service

Last updated：2024-01-18 15:11:45

This document describes how to integrate with and debug the GME APIs for iOS.

Note:

This document applies to GME SDK version 2.9.

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:

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.

Non-streaming speech-to-text API ***SpeechToText()***: There can be up to 10 concurrent requests per account.

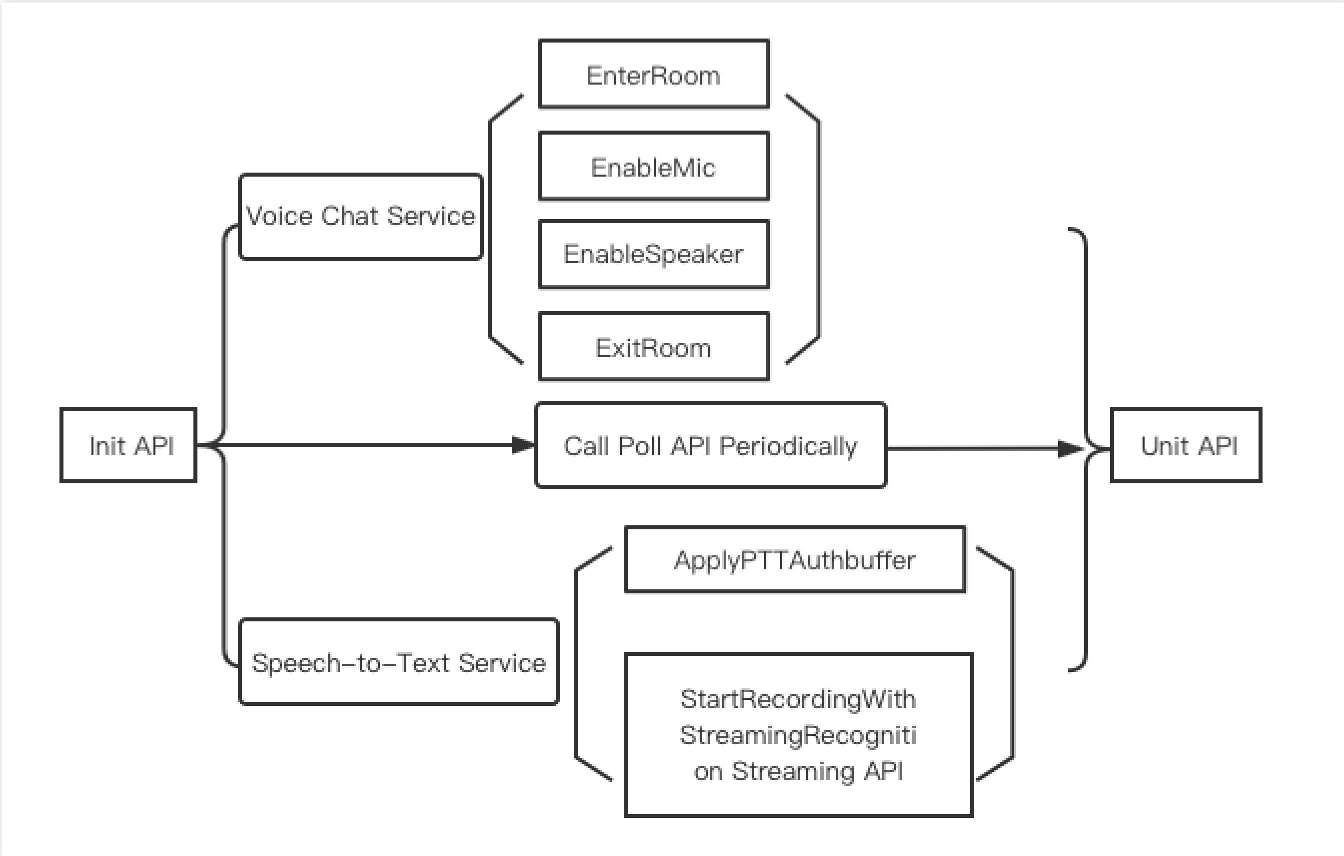
Streaming speech-to-text API ***StartRecordingWithStreamingRecognition()***: There can be up to 50 concurrent requests per account.

Real-time streaming speech-to-text API ***StartRealTimeASR()***: There can be up to 50 concurrent requests per account.

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**.

The billing will not start after initialization. Receiving or sending a voice message in speech-to-text service is counted as a voice message DAU.



Directions

1. Initializing GMEAPI: Init
2. Calling Poll periodically to trigger event callbacksAPI: Poll
3. Initializing authenticationAPI: ApplyPTTAuthbuffer
4. Starting streaming speech recognitionAPI: StartRecordingWithStreamingRecognition
5. Stop recordingAPI: StopRecording
6. Uninitializing GMEAPI: UnInit

Important notes

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error code, see Error Codes.

Core APIs

Before the initialization, the SDK is in the uninitialized status, and **you need to initialize it through the `Init` API** before you can use the voice chat and speech-to-text services.

Call the `Init` API before calling any APIs of GME.

If you have any questions when using the service, see General.

API	Description
InitEngine	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME
SetDefaultAudienceAudioCategory	Sets audio playback in background on device

Imported header files

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

Getting singleton

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

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

Sample code

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

Setting callbacks

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.

Sample code

`ITMGDelegate` is used for declaration.

```
@interface TMGDemoViewController ()<ITMGDelegate>{}
```

```
ITMGDelegate < NSObject >

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

The API callback messages is processed in `OnEvent` . For the message type, see `ITMG_MAIN_EVENT_TYPE` . The message content is a dictionary for parsing the API callback contents.

Function prototype

```
- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary*)data;
```

Sample code

```
//TMGRealTimeViewController.m
TMGRealTimeViewController ()< ITMGDelegate >


- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data {
    NSString *log = [NSString stringWithFormat:@"OnEvent:%d,data:%@",
(int)eventType, data];
    [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:(%@)", result, error_info]];

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

// Refer to DispatchCenter.h and DispatchCenter.m
```

Initializing SDK

This API is used to initialize the GME service. It is recommended to call it when initializing the application. No fee is incurred for calling this API.

For more information on how to get the `sdkAppID` parameter, see Activating Services.

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

Note:

The Init API must be called in the same thread with other APIs. It is recommended to call all APIs in the main thread.

Function prototype

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

Parameter	Type	Description
sdkAppId	String	`AppId` provided by the GME service from the Tencent Cloud console
OpenId	String	`OpenId` can only be in Int64 type, which is passed after being converted to a string.

Returned Value	Description
QAV_OK= 0	Initialized SDK successfully.
QAV_ERR_SDK_NOT_FULL_UPDATE= 7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is only a reminder but will not cause an initialization failure.
If this error is reported during integration, please check the integrity and version of the SDK file as prompted.
If this error is returned after executable file export, please ignore it and try to avoid displaying it in the UI.

Sample code

```
_openId = _userIdText.text;
_appId = _appIdText.text;
[[ITMGContext GetInstance] InitEngine:SDKAPPID openID:_openId];
```

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 EnginePollHelper.m file in Demo.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

Function prototype

```
-(void)Poll;
```

Sample code

```
[[ITMGContext GetInstance] Poll];
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause.
If you need to pause the audio when switching to the background, you can call the `Pause` API in the listening code used to switch to the background, and call the `Resume` API in the listening event used to resume the foreground.

Function prototype

```
-(QAVResult)Pause;
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

Function prototype

```
-(QAVResult)Resume;
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **Switching accounts requires uninitialization**.

Function prototype

```
-(int)Uninit;
```

Sample code

```
[[ITMGContext GetInstance] Uninit];
```

Audio settings for iOS device

This API is used to set the audio playback in the background, and the GME audio not to be affected by the mute switch or lock screen. For example, when the notification center or control center is opened, you can still receive and play back the GME audio. You need to call this API before room entry.

Meanwhile, you should pay attention to the following two points in the application:

Audio engine capture and playback are not paused when the application is switched to the background (i.e., `PauseAudio`).

You need to add at least `key:Required background modes` and `string:App plays audio or streams audio/video using AirPlay` to the `Info.plist` of the application.

Note:

It is recommended that developers call this API to set the audio.

Function prototype

```
-(QAVResult)SetDefaultAudienceAudioCategory:(ITMG_AUDIO_CATEGORY)audioCategory;
```

Type	Parameter	Description
ITMG_CATEGORY_AMBIENT	0	Audio is not played back in the background (default value)
ITMG_CATEGORY_PLAYBACK	1	Audio is played back in the background

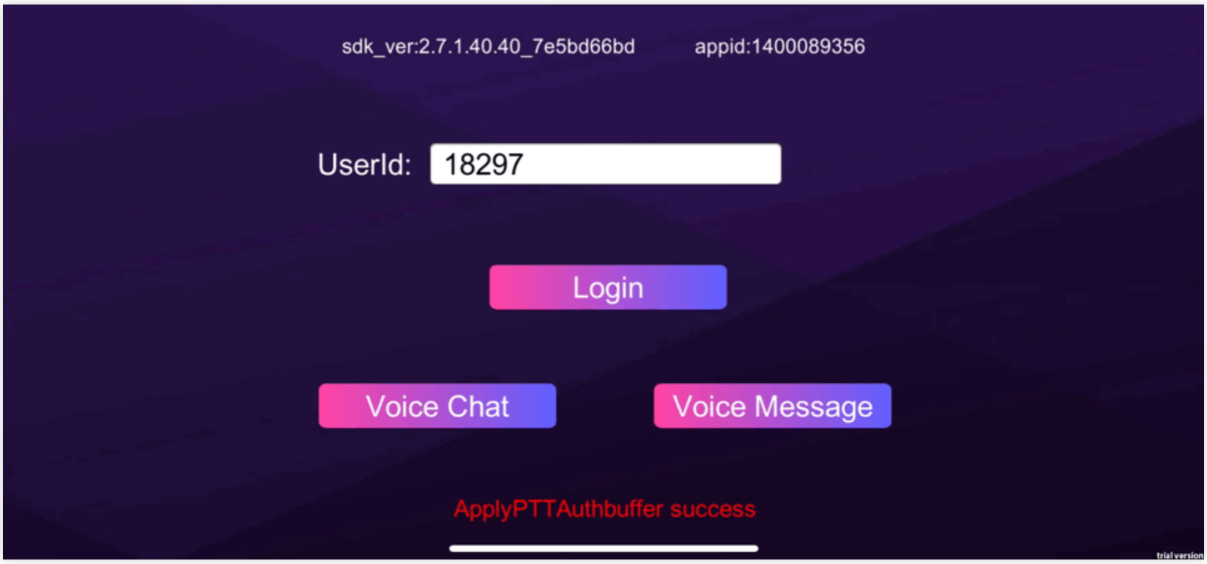
This can be achieved by modifying kAudioSessionProperty_AudioCategory. For more information, see Apple official documentation.

Sample code

```
[[ITMGContext
GetInstance]SetDefaultAudienceAudioCategory:ITMG_CATEGORY_AMBIENT];
```

Speech-to-Text

Voice messaging refers to recording and sending a voice message. At the same time, the voice message can be converted to text and translated, as shown below:

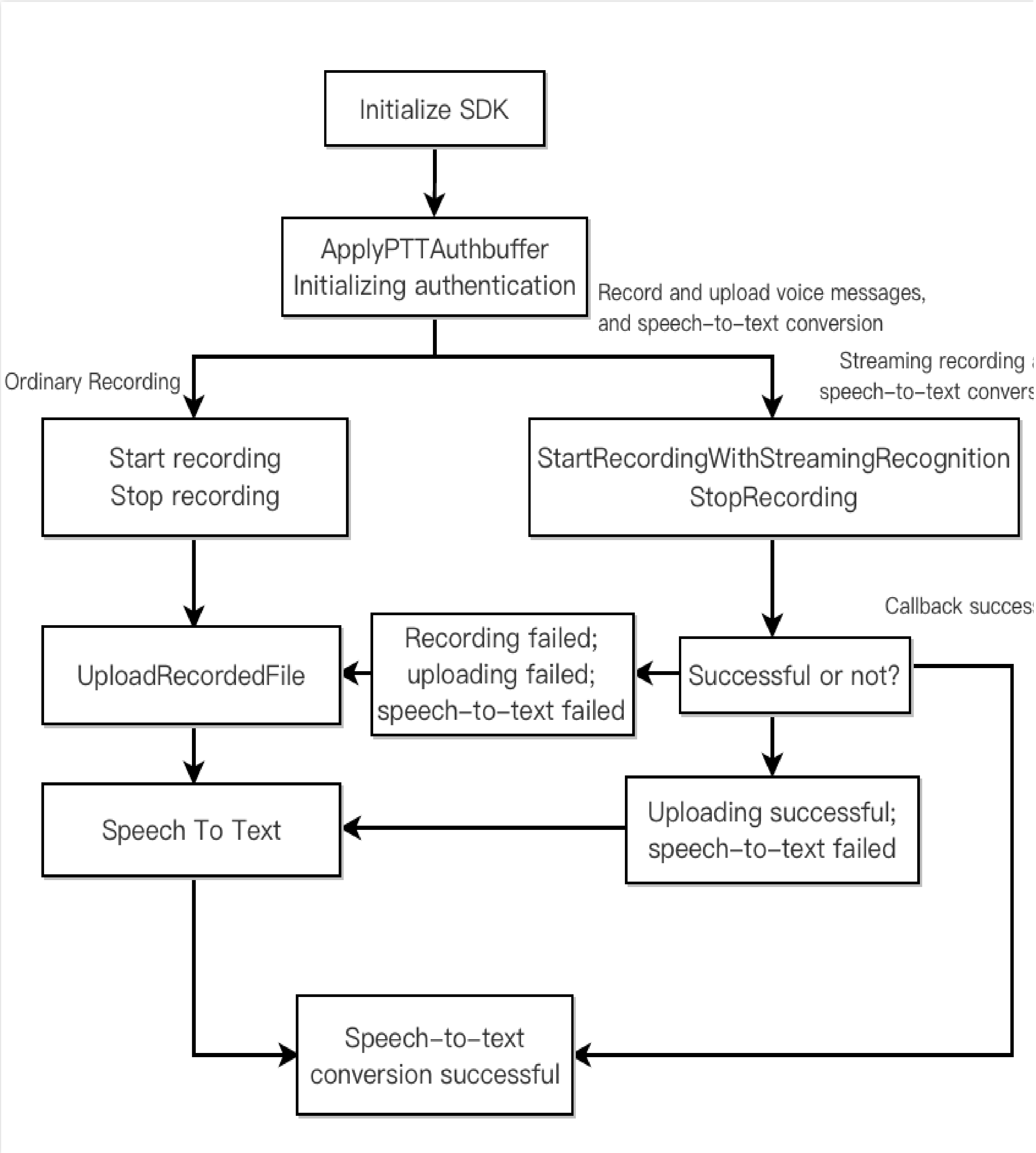


Note:

It is recommended to use the streaming speech-to-text service.

You do not need to enter a voice chat room when using the voice messaging service.

Voice message and speech-to-text conversion flowchart



Integrating Voice Messaging and Speech-to-Text Service

Voice messaging and speech-to-text APIs

API	Description

ApplyPTTAuthbuffer	Initializes authentication
SetMaxMessageLength	Specifies the maximum length of voice message
StartRecording	Starts recording
StartRecordingWithStreamingRecognition	Starts streaming recording
PauseRecording	Pauses recording
ResumeRecording	Resumes recording
StopRecording	Stops recording
CancelRecording	Cancels recording
GetMicLevel	Gets the real-time mic volume
SetMicVolume	Sets the recording volume
GetMicVolume	Gets the recording volume
GetSpeakerLevel	Gets the real-time speaker volume
SetSpeakerVolume	Sets the playback volume
GetSpeakerVolume	Gets the playback volume
UploadRecordedFile	Uploads the audio file
DownloadRecordedFile	Downloads the audio file
PlayRecordedFile	Plays back audio
StopPlayFile	Stops playing back audio
GetFileSize	Gets the audio file size
GetVoiceFileDuration	Gets the audio file duration
SpeechToText	Converts speech to text

Maximum recording duration

The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum recording duration to 10 seconds, please call the `SetMaxMessageLength` API to set it after initialization.

Initializing the SDK

Before the initialization, the SDK is in the uninitialized status, and you need to initialize it through the `Init` API before you can use the voice chat and voice message services.
If you have any questions when using the service, see Speech-to-text Conversion.

Authentication information

Generate `AuthBuffer` for encryption and authentication of relevant features. For release in the production environment, use the backend deployment key as detailed in Authentication Key.
To get authentication for voice message and speech-to-text, the room ID parameter must be set to `null` .

Function prototype

```
@interface QAVAuthBuffer : NSObject
+ (NSData*) GenAuthBuffer:(unsigned int)appId roomId:(NSString*)roomId openID:
(NSString*)openID key:(NSString*)key;
+ @end
```

Parameter	Type	Description
appId	int	`AppId` from the Tencent Cloud console.
roomId	NSString	Enter `null` .
openID	NSString	User ID, which is the same as `openID` during initialization.
key	NSString	Permission key from the Tencent Cloud console.

Sample code

```
#import "GMESDK/QAVAuthBuffer.h"
NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:SDKAPPID3RD.intValue
roomId:_roomId openID:_openId key:AUTHKEY];
```

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

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

Parameter	Type	Description
authBuffer	NSData*	Authentication

Sample code

```
[[[ITMGContext GetInstance]GetPTT]ApplyPTTAuthbuffer:(NSData *)authBuffer];
```

Streaming Speech Recognition

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

```
-(int)StartRecordingWithStreamingRecognition:(NSString *)filePath;
-(int)StartRecordingWithStreamingRecognition:(NSString *)filePath language:
(NSString*)speechLanguage translatelanguage:(NSString*)translateLanguage;
```

Parameter	Type	Description
filePath	String	Path of stored audio file
speechLanguage	String	The language in which the audio file is to be converted to text. For parameters, please see Language Parameter Reference List
translateLanguage	String	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 `speechLanguage`)

Sample code

```
recordfilePath = [docDir stringByAppendingFormat:@"/test_%d.ptt",index++];
[[[ITMGContext GetInstance] GetPTT]
StartRecordingWithStreamingRecognition:recordfilePath language:@"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` . Event messages are divided into:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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. `fileid` is fixed at `http://gme-v2-`

Note:
The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

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 conversion 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.

Sample code

```
- (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)
            {
                donwLoadUrlPath = data[@"file_id"];

                recordfilePath = [data objectForKey:@"file_path"];
                _localFileField.text = recordfilePath;

                _donwloadUrlField.text = [data objectForKey:@"file_id"] ;

                UITextField *_audiotoTextField =
(UITextField*)objc_getAssociatedObject(self, [PTT_AUDIO_TO_TEXT UTF8String]);
                _audiotoTextField.text = [data objectForKey:@"text"] ;
            }

        }
            break;
    }
}
```

Voice Message Recording

The recording process is as follows: start recording -> stop recording -> return recording callback -> start the next recording.

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

Function prototype

```
-(QAVResult)SetMaxMessageLength:(int)msTime
```

Parameter	Type	Description

msTime	int	Audio duration in ms. Value range: 1000 < msTime <= 58000

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SetMaxMessageLength:(int)msTime];
```

Starting recording

This API is used to start recording. The recording file must be uploaded first before you can perform operations such as speech-to-text conversion. **To stop recording, call** `StopRecording` .

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of stored audio file

Sample code

```
recordfilePath =[docDir stringByAppendingFormat:@"/test_%d.ptt",index++];
[[[ITMGContext GetInstance]GetPTT]StartRecording:recordfilePath];
```

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

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

Sample code

```
[[[ITMGContext GetInstance]GetPTT]StopRecording];
```

Callback for recording start

A callback will be executed through a delegate function to pass a message when recording is completed.

To stop recording, call `StopRecording`. The callback for recording start will be returned after the recording is stopped.

The callback function `OnEvent` will be called after recording is started. The event message `ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameter includes `result` and `file_path`.

Error codes

Error Code Value	Cause	Suggested Solution
4097	Parameter is empty.	Check whether the API parameters in the code are correct.
4098	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Ensure that the SDK recording feature is used at the right time.
4100	Audio data is not captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	Mic permission is required for using the SDK. To add the permission, please see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE:
```

```
            {
            //Recording callback
            }
                break;
        }
}
```

Pausing recording

This API is used to pause recording. If you want to resume recording, please call the `ResumeRecording` API.

Function prototype

```
-(int)PauseRecording;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]PauseRecording];
```

Resuming recording

This API is used to resume recording.

Function prototype

```
-(int)ResumeRecording;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]ResumeRecording];
```

Canceling recording

This API is used to cancel recording. There is no callback after cancellation.

Function prototype

```
-(QAVResult)CancelRecording;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]CancelRecording];
```

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)GetMicLevel;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetMicLevel];
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)SetMicVolume:(int) volume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SetMicVolume:100];
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(int)GetMicVolume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetMicVolume];
```

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)GetSpeakerLevel;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetSpeakerLevel];
```

Setting the playback volume of voice message

This API is used to set the playback volume of voice messaging. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)SetSpeakerVolume:(int)volume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SetSpeakerVolume:100];
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice messaging. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT`.

Function prototype

```
-(int)GetSpeakerVolume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetSpeakerVolume];
```

Voice Message Playback

Playing back audio

This API is used to play back audio.

Function prototype

```
-(int)PlayRecordedFile:(NSString*)filePath;
-(int)PlayRecordedFile:(NSString*)filePath VoiceType:(ITMG_VOICE_TYPE) type;
```

Parameter	Type	Description
downloadFilePath	NSString	Local audio file path
type	ITMG_VOICE_TYPE	Voice changer type. For more information, see Voice Changing Effects.

Error codes

Error Code Value	Cause	Suggested Solution

20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]PlayRecordedFile:path];
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameter includes `result` and `file_path` .

Error codes

Error Code Value	Cause	Suggested Solution
20481	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	Parameter is empty.	Check whether the API parameters in the code are correct.
20484	Internal error.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE:
        {
        // Callback for audio playback
        }
            break;
```

```
    }
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

Function prototype

```
-(int)StopPlayFile;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]StopPlayFile];
```

Getting audio file size

This API is used to get the size of an audio file.

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of audio file, which is a local path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetFileSize:path];
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of audio file, which is a local path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetVoiceFileDuration:path];
```

Voice Message Upload and Download

Uploading an audio file

This API is used to upload an audio file.

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of uploaded audio file, which is a local path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]UploadRecordedFile:path];
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Error codes

Error Code Value	Cause	Suggested Solution
8193	An error occurred while accessing the	Ensure the existence of the file and the validity of

	file during upload.	the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
8195	A network error occurred.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8200	No `appinfo` is set.	Check whether the `apply` API is called or whether the input parameters are empty.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE:
        {
            if (data != NULL &&[[data objectForKey:@"result"] intValue]== 0)
            {
                _donwloadUrlField.text = [data objectForKey:@"file_id"] ;
                donwLoadUrlPath = [data objectForKey:@"file_id"] ;
            }
        }
            break;
    }
}
```

Downloading the audio file

This API is used to download an audio file.

Function prototype

```
-(void)DownloadRecordedFile:(NSString*)fileId downloadFilePath:
(NSString*)downloadFilePath
```

Parameter	Type	Description
fileID	NSString	File URL path
downloadFilePath	NSString	Local path of saved file

Sample code

```
[[[ITMGContext GetInstance]GetPTT]DownloadRecordedFile:fileIdpath
downloadFilePath:path];
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Error codes

Error Code Value	Cause	Suggested Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
12291	Network storage system exception	The server failed to get the audio file. Check whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	Server file system error.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed during the process of getting the download parameters.	Check whether the device can access the internet.

12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE:
        {
            if (data != NULL &&[[data objectForKey:@"result"] intValue]== 0)
            {
                _audiofileToPlayField.text = [data objectForKey:@"file_path"] ;
                donwLoadLocalPath = [data objectForKey:@"file_path"];
            }
            else
            {
                donwLoadLocalPath = NULL;
            }
        }
            break;
    }
}
```

Speech-to-Text Service

Converting audio file to text

This API is used to convert a specified audio file to text.

Function prototype

```
-(void)SpeechToText:(NSString*)fileID;
```


Parameter	Type	Description
fileID	NSString	URL of audio file

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SpeechToText:fileID];
```

Translating audio file into text in specified language

This API can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation.

Function prototype

```
-(void)SpeechToText:(NSString*)fileID (NSString*)speechLanguage
(NSString*)translateLanguage;
```

Parameter	Type	Description
fileID	NSString*	URL of audio file, which will be retained on the server for 90 days
speechLanguage	NSString*	The language in which the audio file is to be converted to text. For parameters, please see Language Parameter Reference List.
translateLanguage	NSString*	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 `speechLanguage` .

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SpeechToText:fileID speechLanguage:"cmn-Hans-
CN" translateLanguage:"cmn-Hans-CN"];
```

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the

`OnEvent` function.

The passed parameters include `result` , `file_path` and `text` (recognized text).

Error codes

Error Code Value	Cause	Suggested Solution
32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	Network failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	Incorrect speech-to-text conversion parameter.	Check whether the API parameter `fileid` in the code is empty.
32785	Speech-to-text translation returned an error.	Error with the backend of voice message and speech-to-text feature. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE:
        {
            if (data != NULL &&[[data objectForKey:@"result"] intValue]== 0)
            {
                UITextField *_audiotoTextField =
(UITextField*)objc_getAssociatedObject(self, [PTT_AUDIO_TO_TEXT UTF8String]);
                _audiotoTextField.text = [data objectForKey:@"text"] ;
            }
        }
```

```
                break;
        }
}
```

Advanced APIs

Getting version number

This API is used to get the SDK version number for analysis.

Function prototype

```
-(NSString*)GetSDKVersion;
```

Sample code

```
[[ITMGContext GetInstance] GetSDKVersion];
```

Checking mic permission

This API is used to return the mic permission status.

Function prototype

```
-(ITMG_RECORD_PERMISSION)CheckMicPermission;
```

Parameter description

Parameter	Value	Description
ITMG_PERMISSION_GRANTED	0	Mic permission is granted.
ITMG_PERMISSION_Denied	1	Mic is disabled.
ITMG_PERMISSION_NotDetermined	2	No authorization box has been popped up to request the permission.
ITMG_PERMISSION_ERROR	3	An error occurred while calling the API.

Sample code

```
[[ITMGContext GetInstance] CheckMicPermission];
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

Function prototype

```
-(void)SetLogLevel:(ITMG_LOG_LEVEL)levelWrite (ITMG_LOG_LEVEL)levelPrint;
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

ITMG_LOG_LEVEL

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
[[ITMGContext GetInstance] SetLogLevel:TMG_LOG_LEVEL_INFO TMG_LOG_LEVEL_INFO];
```

Setting the log printing path

This API is used to set the log printing path, and needs to be called before initialization. The default path is `Application/********-****-****-************/Documents` .

Function prototype

```
-(void)SetLogPath:(NSString*)logDir;
```

Parameter	Type	Description
logDir	NSString	Path

Sample code

```
[[ITMGContext GetInstance] SetLogPath:Path];
```

Callback Messages

Message list

Message	Description
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Indicates that PTT recording is completed.
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Indicates that PTT upload is completed.
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Indicates that PTT download is completed.
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Indicates that PTT playback is completed.
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Indicates that speech-to-text conversion is completed.

Data list

Message	Data	Sample
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	result;	{"file_path":

	file_path	
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	result; file_path;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	result; file_path;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	result; file_path	{"file_path":
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	result; text;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	result; file_path; text;file_id	{"file_id":"",'

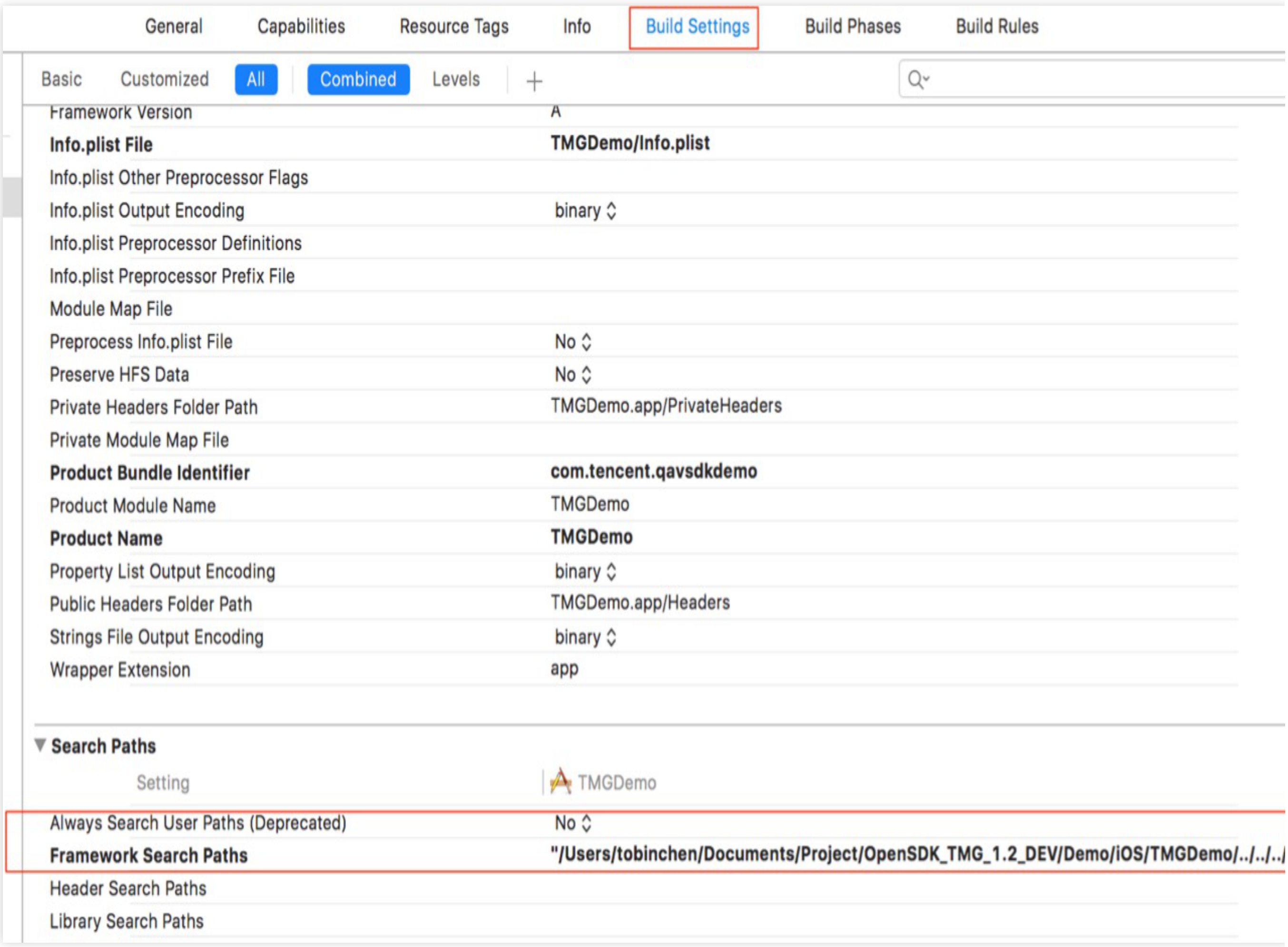
Project Export

Last updated：2024-01-18 15:11:45

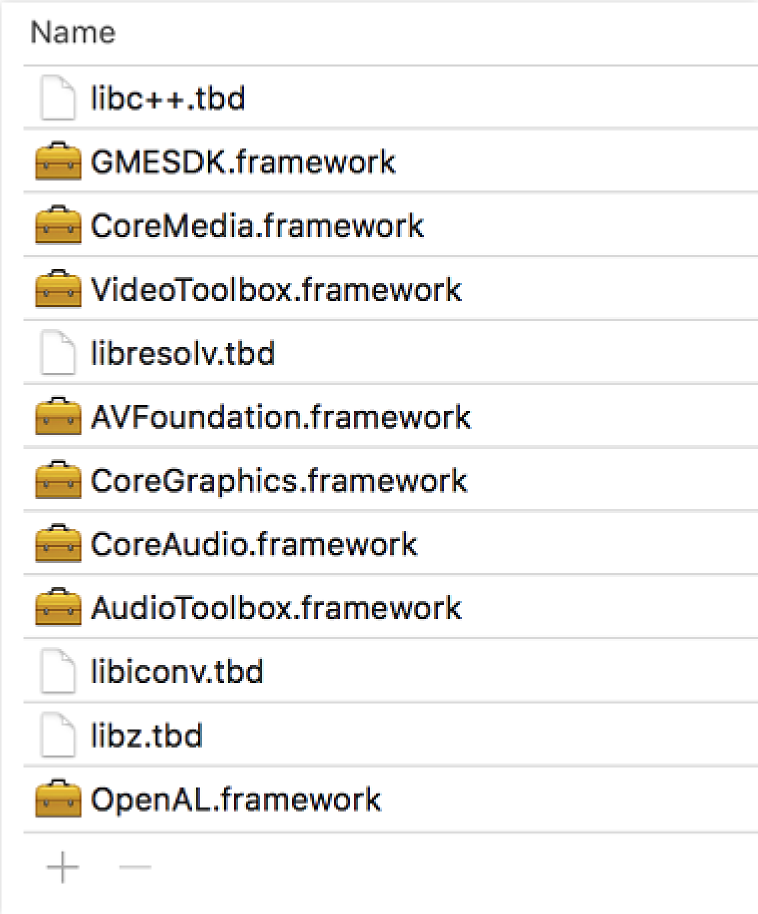
This document mainly describes the notes on exporting the iOS project so that the iOS developers can easily debug and integrate the APIs for Game Multimedia Engine (GME).

Export Configuration

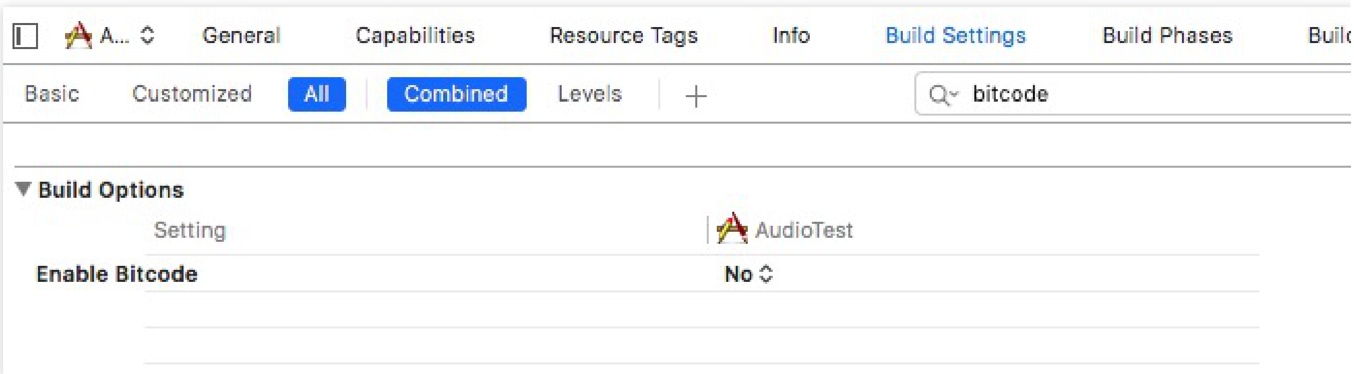
1. Add the following dependent library to **Xcode** > **Link Binary With Libraries** > **Build Setting** as needed, and set Framework Search Paths to point to the directory where the SDK resides, as shown below:



2. Add dependent libraries as shown below:



3. Bitcode should be supported by all class libraries that the project depends on. Bitcode is not supported by the SDK, so it can be disabled. To disable Bitcode, you only need to search for Bitcode under **Targets** > **Build Settings** and set the corresponding option to `NO` .



4. The GME SDK for iOS requires the following permissions:

Required background modes: Allows running in the background (optional).

Microphone Usage Description: Allows access to microphone.

GME 2.9 or Later

If the accessed SDK is on v2.9 or later, you need to configure it as instructed in iOS Project Upgrade Guide.

iOS Project Upgrade Guide

Last updated：2024-01-18 15:11:45

Overview

The GME SDK has been upgraded to v2.9. To implement this upgrade, perform the following steps in your Xcode project:

Upgrade Directions

1. Download the SDK

In the new version, the dynamic libraries of the SDK are split into the following files:

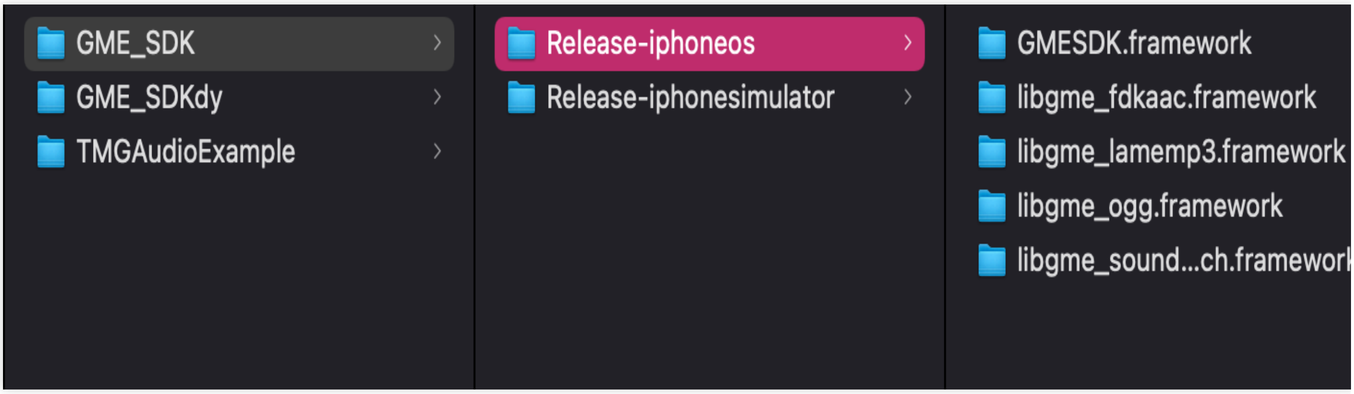
libgmefdkaac.framework

libgmeogg.framework

libgmelamemp3.framework

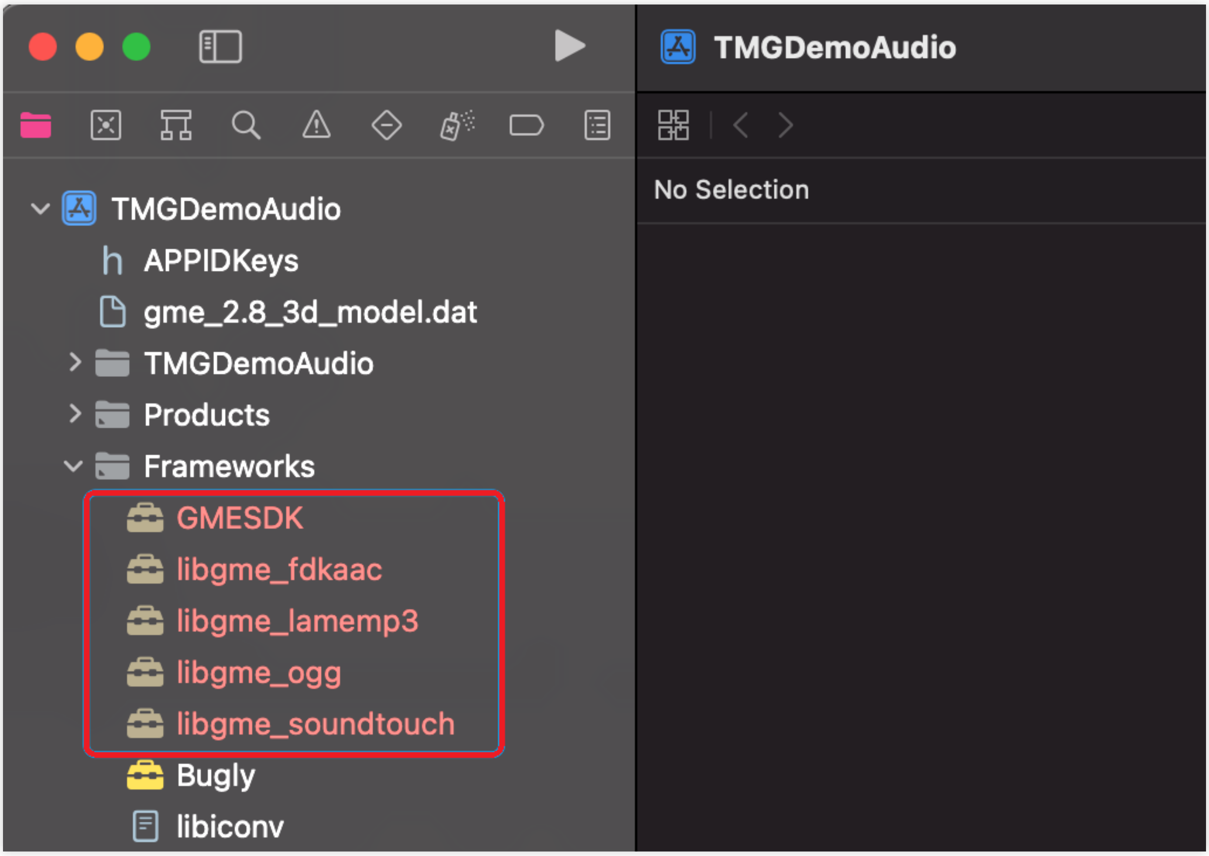
libgmesoundtouch.framework

Make sure that the downloaded SDK contains these files. After downloading, put them together with `GMESDK.framework` in the project directory. `Release-iphoneos` is the SDK file used for real devices, while `Release-iphonesimulator` is the SDK file used for simulators.



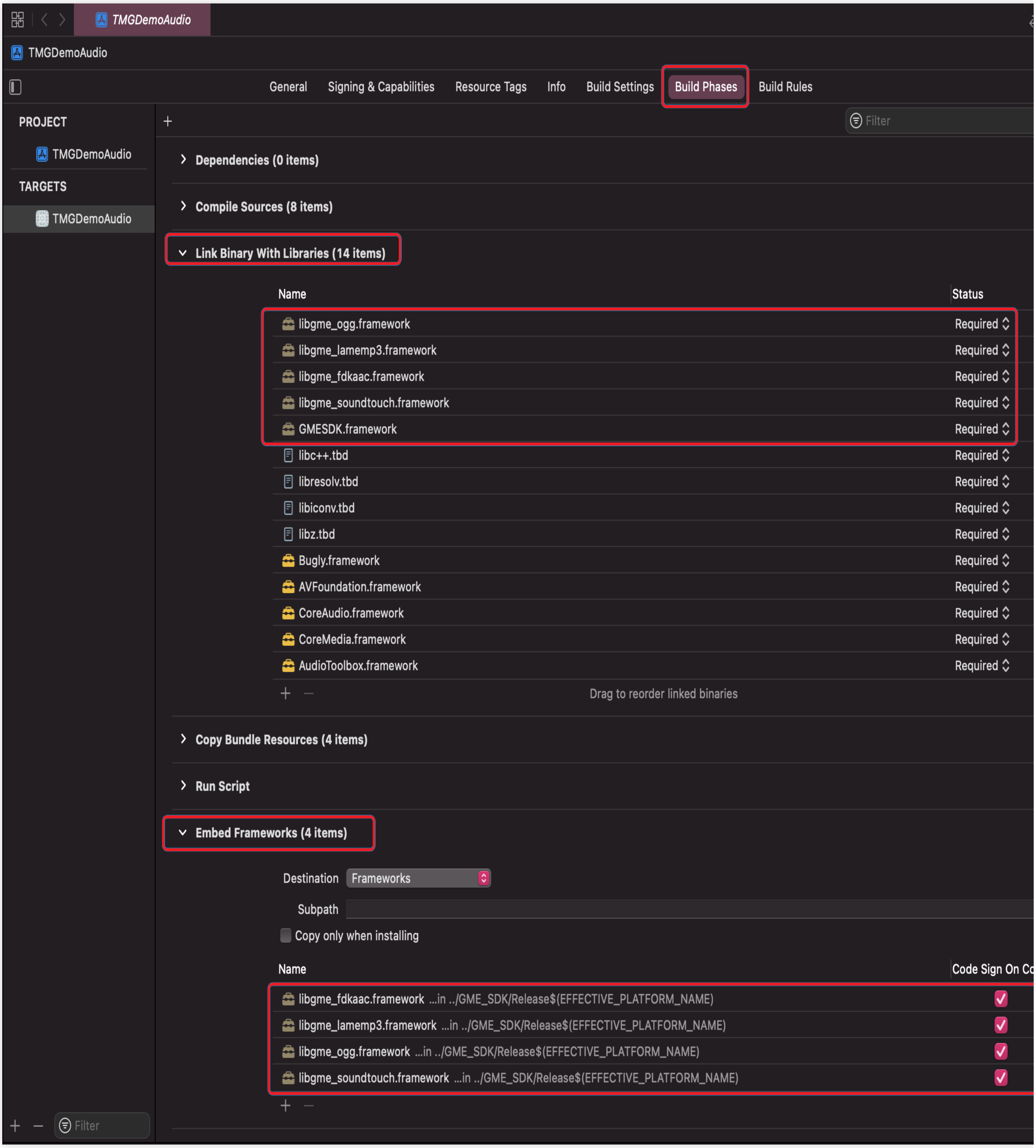
2. Import the SDK into the project

Import all frameworks into the project as shown below:



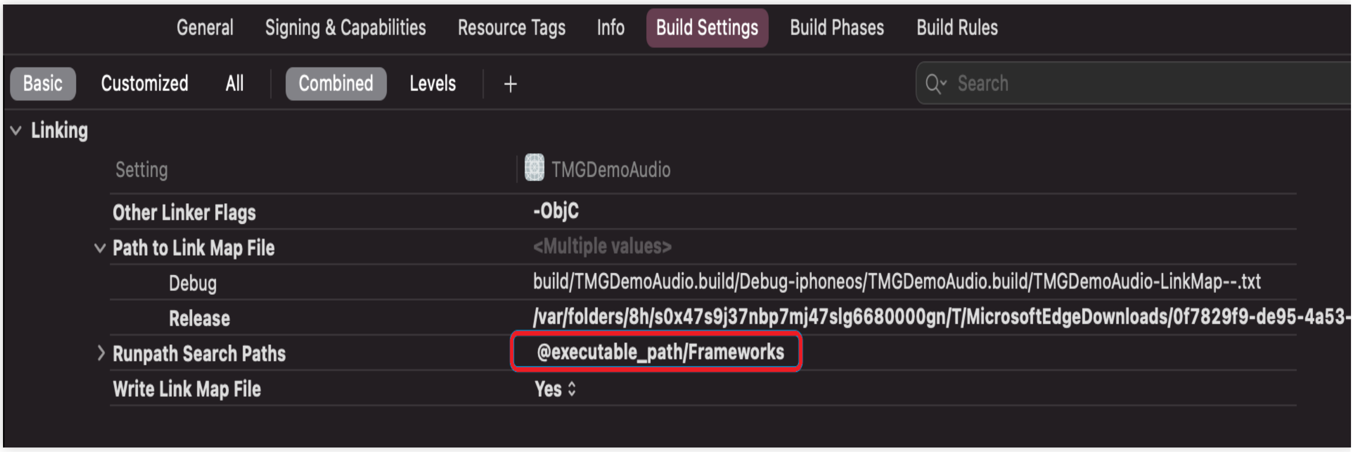
3. Configure frameworks and sign

1. In the Xcode project, click **Build Phases**, expand **Link Binary With Libraries**, and import all GME frameworks.
2. Expand **Embed Framework**, import all GME frameworks, and select **Code Sign On Copy**.



4. Modify the rpath

You need to add `@executable_path/Frameworks` in the `rpath` . If it has already been added, there is no need for modification.


General
Signing & Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Basic
Customized
All
Combined
Levels
Search
Linking
Setting
TMGDemoAudio
Other Linker Flags
-ObjC
Path to Link Map File
<Multiple values>
Debug
build/TMGDemoAudio.build/Debug-iphoneos/TMGDemoAudio.build/TMGDemoAudio-LinkMap--.txt
Release
/var/folders/8h/s0x47s9j37nbp7mj47slg6680000gn/T/MicrosoftEdgeDownloads/0f7829f9-de95-4a53-
Runpath Search Paths
@executable_path/Frameworks
Write Link Map File
Yes

SDK for Android

Integrating SDK

Last updated：2024-01-18 15:13:51

This document describes how to integrate GME SDK into an Android project so that the Android developers can easily debug and integrate the APIs for Game Multimedia Engine (GME).

Preparing SDK

1. Download the applicable demo and SDK. For more information, see SDK Download Guide.
2. Decompress the obtained SDK resources.
3. The SDK development resources are in the `libs` folder.

Note:

You can run the SDK on Android 5.0 or later.

Configuration Guide

Method 1

1. Copy the `gmesdk.jar` file in the `libs` directory to the `libs` directory of the Android project.
2. **Copy the library files of the corresponding architecture based on the project requirements**. For example, if the project requires the armeabi-v7a architecture, you need to copy the library files in the `armeabi-v7a` directory to the `armeabi-v7a` directory in the project. If there is no `armeabi-v7a` directory in the project, create one.

Method 2

Place .so and .jar files in any folder in the project and specify the folder in `surceSets` .

Configuring the project

Add the code that imports the library to build.gradle under the App directory of the project.

```
sourceSets {
        main {
            jniLibs.srcDirs = ['libs']
        }
}
```

Voice Chat

Last updated：2024-01-18 15:13:51

This document describes how to integrate with and debug GME client APIs for the voice chat feature for Android.

Key Considerations for Using GME

GME provides the real-time voice chat service, voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

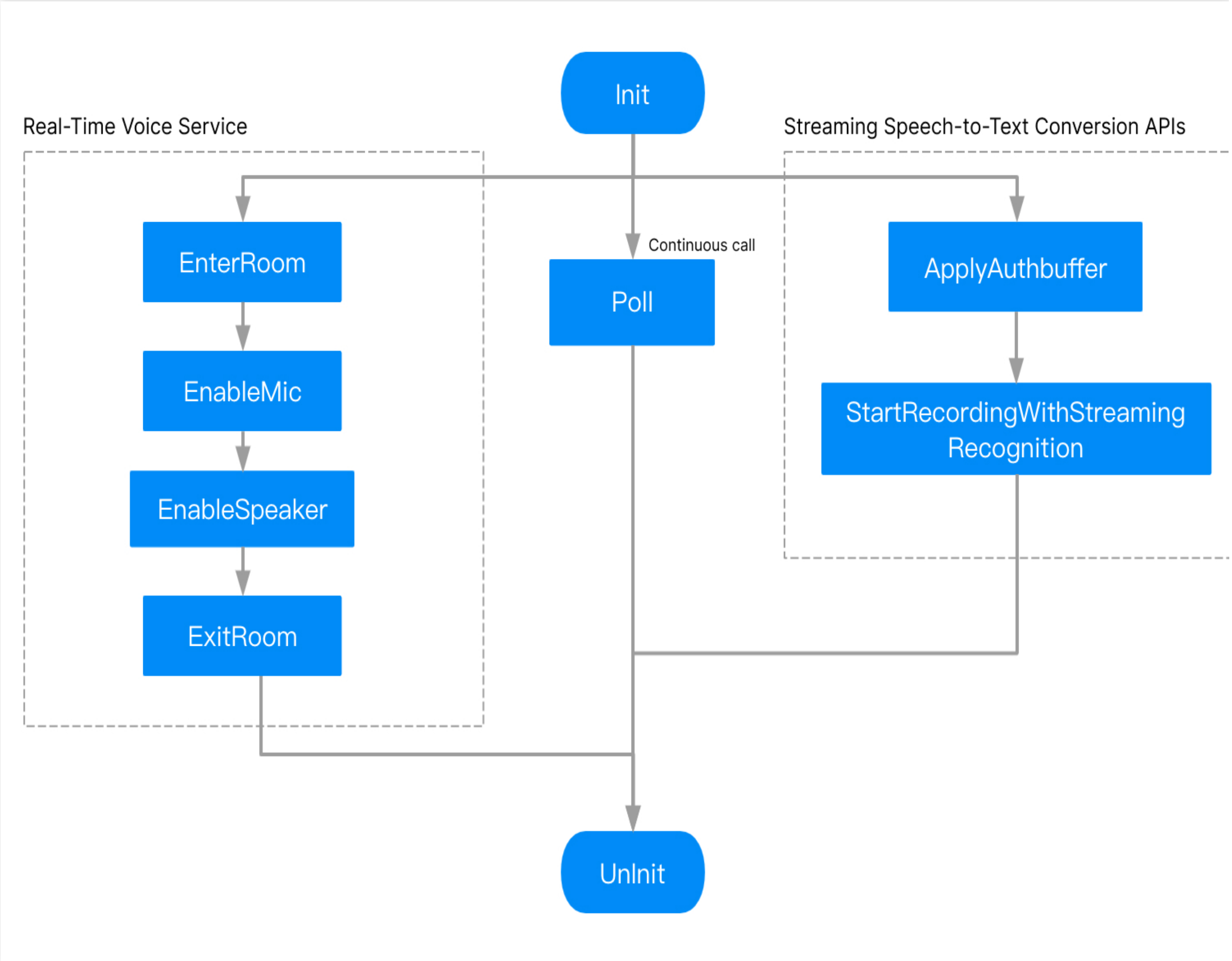
Notes

You have created a GME application and obtained the `AppID` and `Key` of the SDK as instructed in Activating Services.
You have **activated the voice chat, voice messaging, and speech-to-text services of GME**. For more information, see Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `QAVError.OK` will be returned with the value being `0` .
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error codes, see Error Codes.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initializing GME
2. Calling Poll periodically to trigger callbacks
3. Entering a voice chat room
4. Turning on the mic
5. Turning on the speaker
6. Exiting the voice chat room
7. Uninitializing GME

Voice chat for Android class

Class	Description
ITMGContext	Core APIs
ITMGRoom	Room APIs
ITMGRoomManager	Room management APIs

ITMGAudioCtrl	Audio APIs
ITMGAudioEffectCtrl	Sound effect and accompaniment APIs

Core APIs

API	Description
Init	Initializes GME.
Poll	Triggers an event callback.
Pause	Pauses the system.
Resume	Resumes the system.
Uninit	Uninitializes GME.

Note:

If you need to switch the account, please call `UnInit` to uninitialize the SDK. No fee is incurred for calling Init API.

Getting singleton

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

Sample code

```
import com.tencent.TMG.ITMGContext;
ITMGContext.getInstance(this);
```

Registering 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.

API prototype

```
static public abstract class ITMGDelegate {
    public void OnEvent(ITMG_MAIN_EVENT_TYPE type, Intent data){}
}
```

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

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

Sample code

```
private ITMGContext.ITMGDelegate itmgDelegate = null;
itmgDelegate = new ITMGContext.ITMGDelegate() {
    @Override
    public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
        if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM == typ
        {
            // Analyze the returned data
            int nErrCode = data.getIntExtra("result" , -1);
            String strErrMsg = data.getStringExtra("error_info");
                }
        }
}
```

Register the callback function to the SDK before room entry.

API prototype

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

Parameter	Type	Description
delegate	ITMGDelegate	SDK callback function

Sample code

```
ITMGContext.GetInstance(this).SetTMGDelegate(itmgDelegate);
```

Initializing the SDK

You need to initialize the SDK through the `Init` API before you can use the real-time voice chat, 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

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

Parameter	Type	Description
sdkAppId	String	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openId	String	`openID` can only be in Int64 type, which is passed in after being converted to a `const char*` . 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.

Returned values

Returned Value	Description
QAVError.OK= 0	SDK was initialized successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Check whether the SDK file is complete. We recommend that you delete it and then import the SDK again.

Notes on 7015 error code

The error code 7015 is identified based on the MD5 value. If this error is reported during integration, check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party enhancement, the Unity packaging mechanism, and other factors, the library file MD5 will be affected, resulting in misjudgment. Therefore, **ignore this error in the logic for official releases**, and avoid displaying it on the UI.

Sample code

```
String sdkAppID = "14000xxxxx";
String openID = "100";
int ret = 0;
// After the user agrees to the application's privacy policy, initialize the SDK at
//ret = 0: The user agrees to the application's privacy policy
//ret = 1: The user does not agree to the application's privacy policy
// If the user does not agree to the privacy policy, change `ret` to a value other
if(ret != 0){
    Log.e(TAG,"The user does not agree to the application's privacy policy");
}else{
        ITMGContext.GetInstance(this).Init(sdkAppId, openId);
}
```

Triggering event callback

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

Refer to the `EnginePollHelper.java` file in SDK Download Guide.

Call the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
public abstract int Poll();
```

Sample code

```
private Handler mhandler = new Handler();private Runnable mRunnable = new Runnable(
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. If you do not need the audio played back in the background in the room, call `Pause` API to pause the GME service.

API prototype

```
public abstract int Pause();
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
public abstract int Resume();
```

Uninitializing SDK

This API is used to uninitialize the SDK. **If the game account is bound to** `openid` , switching game account requires uninitializing GME and then using the new `openid` to initialize again.

Note:

If the end user revokes the permission granted to the SDK to process the personal information, you can call the `Uninit` API to stop using the SDK features and stop collecting and close the user data used by the features.

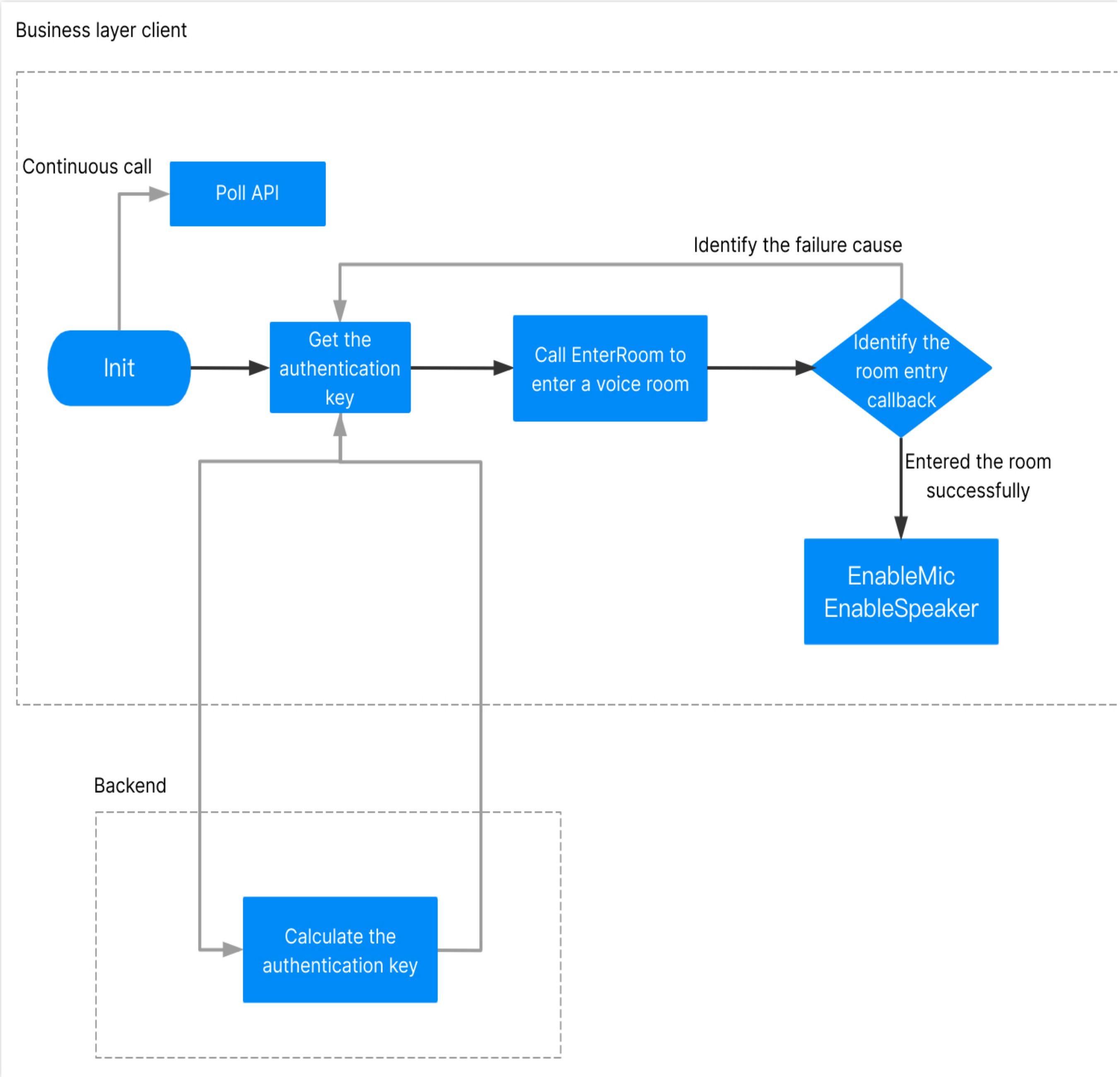
API prototype

```
public abstract int Uninit();
```

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, see Sound and Audio.



API	Description
GenAuthBuffer	Calculates the local authentication key.
EnterRoom	Enters a room.
ExitRoom	Exits a room.
IsRoomEntered	Determines whether room entry is successful.
SwitchRoom	Switches the room quickly.
StartRoomSharing	Cross-room Co-anchoring

Local authentication key calculation

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

API prototype

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

Parameter	Type	Description
appId	int	`AppId` from the Tencent Cloud console
roomId	String	Room ID, which can contain up to 127 characters.
openId	String	User ID, which is the same as `OpenId` during initialization.
key	String	Permission key from the Tencent Cloud console.

Sample code

```
import com.tencent.av.sig.AuthBuffer;// Header file
byte[] authBuffer = AuthBuffer.getInstance().genAuthBuffer(Integer.parseInt(sdkAppI
```

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.

Note:

If the room entry callback result is `0`, the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.

The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will become smooth sound quality. Only when a member in the room calls the `ChangeRoomType` API, the audio type of the room will be changed.

API prototype

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

Parameter	Type	Description

roomId	String	Room ID, which can contain up to 127 characters.
roomType	int	Room type. We recommend that you enter `ITMG_ROOM_TYPE_FLUENCY` . For more information on room audio types, see Sound Quality.
authBuffer	byte[]	Authentication key

Sample code

```
ITMGContext.GetInstance(this).EnterRoom(roomId,roomType, authBuffer);
```

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

Will the billing continue if the client is disconnected from the server when using the voice chat?

Function prototype

```
private ITMGContext.ITMGDelegate itmgDelegate = null;
itmgDelegate= new ITMGContext.ITMGDelegate() {
            @Override
             public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data
                }
};
```

Sample code

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

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM == type)
        {
               // Analyze the returned data
            int nErrCode = data.getIntExtra("result" , -1);
            String strErrMsg = data.getStringExtra("error_info");

            if (nErrCode == AVError.AV_OK)
            {
```

```
                //Entered room successfully, and you can proceed with your operatio
                ScrollView_ShowLog("EnterRoom success");
                Log.i(TAG,"EnterRoom success!");
            }
            else
            {
                //If you fail to enter the room, you need to analyze the returned e
                ScrollView_ShowLog("EnterRoom fail :" + strErrMsg);
                Log.i(TAG,"EnterRoom fail!");
            }
        }
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT == ty
        {
            //waiting timeout, please check your network
        }
    }
```

Data details

Message	Data	Example
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnected callback prompt `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code	Cause and Suggested Solution
7006	Authentication failed. Causes: `AppID` doesn't exist or is incorrect. An error occurred while authenticating `authbuff` . Authentication expired. `OpenId` is invalid.
7007	The user was already in another room.
1001	The user was already in the process of entering a room but repeated this operation. We

	recommend that you not call the room entry API until the room entry callback is returned.
1003	The user was already in the room and called the room entry 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.

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API; instead, you can directly call the `EnterRoom` API.

API prototype

```
public abstract int ExitRoom();
```

Sample code

```
ITMGContext.GetInstance(this).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`.

Sample code

```
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
        }
}
```

Data details

Message	Data	Example
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"","result":0}

Determining whether user has entered room

This API is used to determine whether the user has entered a room. A value in bool type will be returned. Do not call this API during room entry.

API prototype

```
public abstract boolean IsRoomEntered();
```

Sample code

```
ITMGContext.GetInstance(this).IsRoomEntered();
```

Switching room

User can call this API to quickly switch the voice chat room after entering the room. After the room is switched, the device is not reset, that is, if the microphone is already enabled in this room, the microphone will keep enabled after the room is switched.

The callback for quickly switching rooms is `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM` , and the fields are `error_info` and `result` .

API prototype

```
public abstract int SwitchRoom(String targetRoomID, byte[] authBuffer);
```

Type descriptions

Parameter	Type	Description
targetRoomID	String	ID of the room to enter
authBuffer	byte[]	Generates a new authentication key with the ID of the room to enter

Callback sample code

```
if(ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM == type) {
int result = data.getIntExtra("result", 1);
String errorInfo = data.getStringExtra("error_info");
if (result == 0) {
            Toast.makeText(getActivity(), "switch room success.", Toast.LENGTH_SHOR
            }
else {
            Toast.makeText(getActivity(), "switch room failed.. error info=" + erro
            }
```

```
}
```

Cross-room mic connection

Call this API to connect the microphones across rooms after the room entry. After the call, the local user can communicate with the target OpenID user in the target room. The target room should be of the same type as the local room.

Example

User a is in room A, user b is in room B, and user a can talk with b through the cross-room API. When user c in room A speaks, users b and d in room B cannot hear. User c in room A can hear only the voice in room A and the voice of user b in room B but not other users in room B.

API prototype

```
/// <summary> Enable the room sharing, and connect the mic of the OpenID in another
public abstract int StartRoomSharing(String targetRoomID, String targetOpenID, byte
/// <summary> Stop the enabled room sharing.</summary>
public abstract int StopRoomSharing();
```

Type descriptions

Parameter	Type	Description
targetRoomID	String	ID of the room to connect mic
targetOpenID	String	Target `OpenID` to connect mic
authBuffer	byte[]	Reserved flag. You just need to enter NULL.

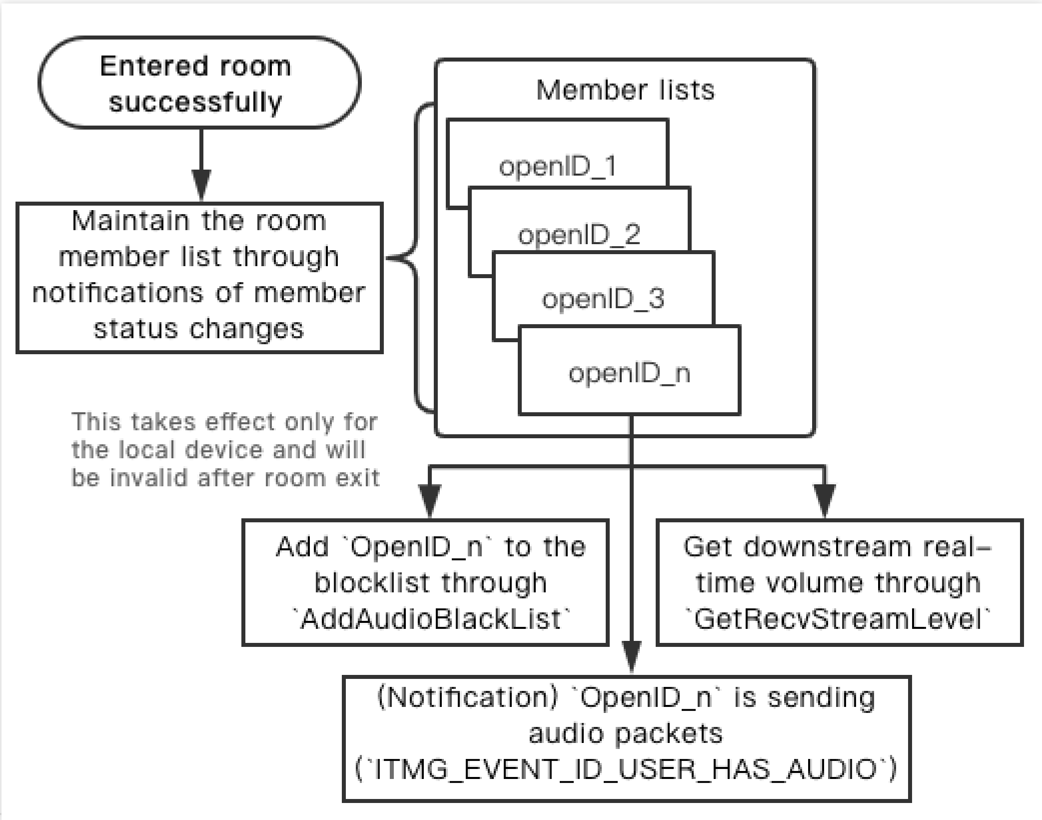
Sample code

```
if (mSwtichRoomShareStart.isChecked())
    {
        String strRoomID = mEditRoomShareRoomID.getText().toString();
        String strOpenID = mEditRoomShareOpenID.getText().toString();
        int nRet = ITMGContext.GetInstance(getActivity()).GetRoom().StartRoomSharin
        if (nRet != 0)
            {
                Toast.makeText(getActivity(), String.format("StartRoomSharing faile
            }else
            {
                int nRet = ITMGContext.GetInstance(getActivity()).GetRoom().StopRoo
                    if (nRet != 0)
```

```
                                {
                                        Toast.makeText(getActivity(), String.format("StopRoomSh
                                }
                        }
}
```

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed.
AddAudioBlackList	Mutes a member in the room.
RemoveAudioBlackList	Unmutes a member.

Notifications of member room entry and speaking status

This API is used to obtain the user speaking in the room and display it in the UI, and to send a notification when someone entering or exiting the room.

Notification for this event will be sent only when the status changes. To get member status in real time, cache the notification when it is received at a higher layer. The event message is `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` , where the data contains `event_id` and `user_list` . The event message will be identified in the `OnEvent` function.
Notifications for audio events are subject to a threshold, and a notification will be sent only when this threshold is exceeded. The notification "A member has stopped sending audio packets" will be sent if no audio packets are received in more than two seconds.

event_id	Description	Maintenance
ITMG_EVENT_ID_USER_ENTER	Return the `openid` of the member entering the room.	Member list
ITMG_EVENT_ID_USER_EXIT	Return the `openid` of the member exiting the room.	Member list
ITMG_EVENT_ID_USER_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
ITMG_EVENT_ID_USER_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_USER_UPDATE == type)
        {
        // Update member status
        int nEventID = data.getIntExtra("event_id", 0);
        String[] openIdList =data.getStringArrayExtra("user_list");
         switch (nEventID)
             {
                   case ITMG_EVENT_ID_USER_ENTER:
                 // A member enters the room
                 break;
                   case ITMG_EVENT_ID_USER_EXIT:
                 // A member exits the room
                       break;
                   case ITMG_EVENT_ID_USER_HAS_AUDIO:
                       // A member sends audio packets
                       break;
                   case ITMG_EVENT_ID_USER_NO_AUDIO:
                       // A member stops sending audio packets
```

```
                        break;
                    default:
                        break;
                }
            }
}
```

Data details

Message	Data	Example
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	event_id; user_list	{"event_id":0,"user_list":""}

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B.

If B blocks neither A nor C, B can hear both of them.

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
public abstract int AddAudioBlackList(String openId);
```

Parameter	Type	Description
openId	String	`openid` of the user to be blocked

Sample code

```
ITMGContext.GetInstance(this).GetAudioCtrl().AddAudioBlackList(openId);
```

Unmuting

This API is used to remove an ID from the audio data blacklist. A returned value of 0 indicates the call is successful.

API prototype

```
public abstract int RemoveAudioBlackList(String openId);
```

Parameter	Type	Description
openId	String	User `openid` to be unblocked

Sample code

```
ITMGContext.GetInstance(this).GetAudioCtrl().RemoveAudioBlackList(openId);
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.
When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend that you call the `EnableMic` or `EnableSpeaker` API.
When the user enters a voice chat room, enabling/disabling a capturing device will restart both capturing and playback devices. If the application is playing back background music, it will also be interrupted. Playback won't be interrupted if the mic is enabled/disabled through control of upstreaming/downstreaming. **Calling method: Call** `EnableAudioCaptureDevice(true)` and `EnableAudioPlayDevice(true)` once after room entry, and call `EnableAudioSend/Recv` to send/receive audio streams when Enable/Disable Mic is clicked.

API	Description
EnableMic	Enables/Disables the mic.
GetMicState	Gets the mic status.
EnableAudioCaptureDevice	Enables/Disables the capturing device.
IsAudioCaptureDeviceEnabled	Gets the capturing device status.
EnableAudioSend	Enables/Disables audio upstreaming.
IsAudioSendEnabled	Gets the audio upstreaming status.
GetMicLevel	Gets the real-time mic volume level.
GetSendStreamLevel	Gets the real-time audio upstreaming volume level.
SetMicVolume	Sets the mic volume level.
GetMicVolume	Gets the mic volume level.

Enabling or disabling mic

This API is used to enable/disable the mic. The mic and speaker are not enabled by default after room entry. `EnableMic` is equivalent to using `EnableAudioCaptureDevice` and `EnableAudioSend` together. **If accompaniment is used, call this API as instructed in** Accompaniment in Voice Chat.

API prototype

```
public abstract int EnableMic(boolean isEnabled);
```

Parameter	Type	Description
isEnabled	boolean	To enable the mic, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
// Turn on mic
ITMGContext.GetInstance(this).GetAudioCtrl().EnableMic(true);
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
public abstract int GetMicState();
```

Sample code

```
int micState = ITMGContext.GetInstance(this).GetAudioCtrl().GetMicState();
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.
This API can only be called after room entry. The device will be disabled automatically after room exit.
Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
public abstract int EnableAudioCaptureDevice(boolean isEnabled);
```

Parameter	Type	Description

isEnabled	boolean	To enable the capturing device, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable capturing device
ITMGContext.GetInstance(this).GetAudioCtrl().EnableAudioCaptureDevice(true);
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
public abstract boolean IsAudioCaptureDeviceEnabled();
```

Sample code

```
bool IsAudioCaptureDevice =
ITMGContext.GetInstance(this).GetAudioCtrl().IsAudioCaptureDeviceEnabled();
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain mute. For more information on how to enable/disable the capturing device, see the `EnableAudioCaptureDevice` API.

API prototype

```
public abstract int EnableAudioSend(boolean isEnabled);
```

Parameter	Type	Description
isEnabled	boolean	To enable audio upstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
ITMGContext.GetInstance(this).GetAudioCtrl().EnableAudioSend(true);
```

Getting audio upstreaming status

This API is used to get the status of audio upstreaming.

API prototype

```
public abstract boolean IsAudioSendEnabled();
```

Sample code

```
bool IsAudioSend =
ITMGContext.GetInstance(this).GetAudioCtrl().IsAudioSendEnabled();
```

Getting the real-time mic volume

This API is used to get the real-time mic volume. An int-type value in the range of 0-100 will be returned. It is recommended to call this API once every 20 ms.

API prototype

```
public abstract int GetMicLevel();
```

Sample code

```
int micLevel = ITMGContext.GetInstance(this).GetAudioCtrl().GetMicLevel();
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume. An int-type value in the range of 0-100 will be returned.

API prototype

```
ITMGContext TMGAudioCtrl int GetSendStreamLevel()
```

Sample code

```
int Level = ITMGContext.GetInstance(this).GetAudioCtrl().GetSendStreamLevel();
```

Setting the mic software volume

This API is used to set the mic volume level. The corresponding parameter is `volume` , which is equivalent to attenuating or gaining the captured sound.

API prototype

```
public abstract int SetMicVolume(int volume);
```

Parameter	Type	Description
volume	int	Value range: 0-200. Default value: `100` . `0` indicates that the audio is muted, while `100` indicates that the volume level remains unchanged.

Sample code

```
ITMGContext.GetInstance(this).GetAudioCtrl().SetMicVolume(volume);
```

Getting the mic software volume

This API is used to obtain the microphone volume. An "int" value is returned. Value 101 represents API SetMicVolume has not been called.

API prototype

```
public abstract int GetMicVolume();
```

Sample code

```
ITMGContext.GetInstance(this).GetAudioCtrl().GetMicVolume();
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker.
GetSpeakerState	Gets the speaker status.
EnableAudioPlayDevice	Enables/Disables the playback device.
IsAudioPlayDeviceEnabled	Gets the playback device status.
EnableAudioRecv	Enables/Disables audio downstreaming.
IsAudioRecvEnabled	Gets the audio downstreaming status.

GetSpeakerLevel	Gets the real-time speaker volume level.
GetRecvStreamLevel	Gets the real-time downstreaming audio levels of other members in the room.
SetSpeakerVolume	Sets the speaker volume level.
GetSpeakerVolume	Gets the speaker volume level.

Enabling or disabling speaker

This API is used to enable/disable the speaker. `EnableSpeaker` is equivalent to using `EnableAudioPlayDevice` and `EnableAudioRecv` together. **If accompaniment is used, call this API as instructed in** Accompaniment in Voice Chat.

API prototype

```
public abstract int EnableSpeaker(boolean isEnabled);
```

Parameter	Type	Description
isEnabled	boolean	To disable the speaker, set this parameter to `false`; otherwise, set it to `true`.

Sample code

```
// Turn on the speaker
ITMGContext.GetInstance(this).GetAudioCtrl().EnableSpeaker(true);
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

API prototype

```
public abstract int GetSpeakerState();
```

Sample code

```
int micState = ITMGContext.GetInstance(this).GetAudioCtrl().GetSpeakerState();
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

API prototype

```
public abstract int EnableAudioPlayDevice(boolean isEnabled);
```

Parameter	Type	Description
isEnabled	boolean	To disable the playback device, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

```
// Enable the playback device
ITMGContext.GetInstance(this).GetAudioCtrl().EnableAudioPlayDevice(true);
```

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
public abstract boolean IsAudioPlayDeviceEnabled();
```

Sample code

```
bool IsAudioPlayDevice =
ITMGContext.GetInstance(this).GetAudioCtrl().IsAudioPlayDeviceEnabled();
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain muted. For more information on how to enable/disable the playback device, see the `EnableAudioPlayDevice` API.

API prototype

```
public abstract int EnableAudioRecv(boolean isEnabled);
```

Parameter	Type	Description
isEnabled	boolean	To enable audio downstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code


```
ITMGContext.GetInstance(this).GetAudioCtrl().EnableAudioRecv(true);
```

Getting audio downstreaming status

This API is used to get the status of audio downstreaming.

API prototype

```
public abstract boolean IsAudioRecvEnabled();
```

Sample code

```
bool IsAudioRecv =
ITMGContext.GetInstance(this).GetAudioCtrl().IsAudioRecvEnabled();
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume. An int-type value will be returned to indicate the volume. It is recommended to call this API once every 20 ms.

API prototype

```
public abstract int GetSpeakerLevel();
```

Sample code

```
int SpeakLevel =
ITMGContext.GetInstance(this).GetAudioCtrl().GetSpeakerLevel();
```

Getting the real-time downstreaming audio levels of other members in room

This API is used to get the real-time audio downstreaming volume of other members in the room. An int-type value will be returned. Value range: 0-200.

API prototype

```
public abstract int GetRecvStreamLevel(String openId);
```

Parameter	Type	Description
openId	String	`openId` of another member in the room

Sample code

```
int Level =
ITMGContext.GetInstance(this).GetAudioCtrl().GetRecvStreamLevel(openId);
```

Dynamically setting the volume of a member of the room

This API is used to set the volume of a member in the room. It takes effect only on the local.

API prototype

```
public abstract int SetSpeakerVolumeByOpenID(String openId, int volume);
```

Parameter	Type	Description
openId	String	`OpenID` of the target user
volume	int	Percentage. Recommended value range: 0-200. Default value: `100` .

Sample code

Executed statements

```
// Lower the volume of 123333 to 80%
String strOpenID = "1233333";
int nOpenVolume = Integer.valueOf(80);
int nRet = ITMGContext.GetInstance(getActivity()).GetAudioCtrl().SetSpeakerVolumeBy
if (nRet != 0)
{
  // Toast error occured
}
else
{
  // Toast set successfully
}
```

Getting volume percentage

Call this API to get the volume set by SetSpeakerVolumeByOpenID

API prototype

```
public abstract int GetSpeakerVolumeByOpenID(String openId);
```

Parameter	Type	Description

openId	String	`OpenID` of the target user

Returned values

API returns volume percentage set by OpenID, where 100 is by default.

Setting the speaker volume

This API is used to set the speaker volume.

API prototype

```
public abstract int SetSpeakerVolume(int volume);
```

Parameter	Type	Description
volume	int	Volume level. Value range: 0-200. Default value: `100` . `0` indicates that the audio is muted, while `100` indicates that the volume level remains unchanged.

Sample code

```
int speVol = (int)(value * 
100);ITMGContext.GetInstance(this).GetAudioCtrl().SetSpeakerVolume(volume);
```

Getting the speaker volume

This API is used to get the speaker volume. An int-type value will be returned to indicate the volume. 101 indicates that the `SetSpeakerVolume` API has not been called.
"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

API prototype

```
public abstract int GetSpeakerVolume();
```

Sample code

```
ITMGContext.GetInstance(this).GetAudioCtrl().GetSpeakerVolume();
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

API prototype

```
public abstract int EnableLoopBack(boolean enable);
```

Parameter	Type	Description
enable	boolean	Specifies whether to enable in-ear monitoring.

Sample code

```
ITMGContext.GetInstance(this).GetAudioCtrl().EnableLoopBack(true);
```

Getting user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, see the `EnterRoom` API.

API prototype

```
public abstract int GetRoomType();
```

Sample code

```
ITMGContext.GetInstance(this).GetRoom().GetRoomType();
```

Getting the room ID

This API is used to get the voice chat room ID and can be called only after a successful room entry. A string will be returned.

API prototype

```
public abstract String GetRoomID();
```

Modifying user's room audio type

This API is used to modify a user's room audio type. For the result, please see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` . The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

API prototype

```
public abstract int ChangeRoomType(int nRoomType);
```

Parameter	Type	Description
nRoomType	int	Room type to be switched to. For room audio types, see the `EnterRoom` API.

Sample code

```
ITMGContext.GetInstance(this).GetRoom().ChangeRoomType(nRoomType);
```

Callback for modifying the room type

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result` , `error_info` , and `new_room_type` . The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description
ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	The existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	A user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).
ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	A user is already in the room, and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	A room member calls the `ChangeRoomType` API to request a change of the room audio type.

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE ==
type)        {           // Process the room type events  }}
```

Data details

Message	Data	Exa
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result;error_info;new_room_type;subEventType	{"e

The monitoring event of room call quality

This is the quality monitoring event, which is triggered once every two seconds after room entry, and its message is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which are as detailed below:

Parameter	Type	Description
weight	int	Value range: 1-50. `50` indicates excellent sound quality, `1` indicates very poor (barely usable) sound quality, and `0` represents an initial meaningless value. Generally, if the value is below 30, you can remind users that the network is poor and recommend them to switch the network.
Loss	double	Upstream packet loss rate
Delay	int	Voice chat delay in ms

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
public abstract String GetSDKVersion();
```

Sample code

```
ITMGContext.GetInstance(this).GetSDKVersion();
```

Checking mic permission

This API is used to return the mic permission status.

Function prototype

```
public abstract ITMG_RECORD_PERMISSION  CheckMicPermission();
```

Parameter description

Parameter	Value	Description
ITMG_PERMISSION_GRANTED	0	The mic permission is granted.
ITMG_PERMISSION_Denied	1	Microphone disabled.
ITMG_PERMISSION_NotDetermined	2	No authorization box has been popped up to request the permission.
ITMG_PERMISSION_ERROR	3	An error occurred while calling the API.

Sample code

```
ITMGContext.GetInstance(this).CheckMicPermission();
```

Checking mic status

Function prototype

```
public abstract ITMG_CHECK_MIC_STATUS CheckMic();
```

Returned value handling

Returned Value	Description	Handling
ITMG_CHECK_MIC_STATUS_AVAILABLE = 0	Normally available	No handling required
ITMG_CHECK_MIC_STATUS_NO_GRANTED = 2	Access not obtained/denied	The access permission needs to be obtained before the mic is enabled.
ITMG_CHECK_MIC_STATUS_INVALID_MIC = 3	No device available	Generally, this error will be reported on PCs when no mics are available. Prompt the user to insert a headset or mic.
ITMG_CHECK_MIC_STATUS_NOT_INIT = 5	Not initialized	Call `EnableMic` after `Init` .

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
public abstract int SetLogLevel(int levelWrite, int levelPrint);
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to log. Default value: `TMG_LOG_LEVEL_INFO` .
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: `TMG_LOG_LEVEL_ERROR` .

`ITMG_LOG_LEVEL` description:

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContext.GetInstance(this).SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO
);
```

Setting the log printing path

This API is used to set the log printing path, and needs to be called before initialization. The default path is /sdcard/Android/data/xxx.xxx.xxx/files.

API prototype

```
public abstract int SetLogPath(String logDir);
```

Parameter	Type	Description
logDir	String	Path

Sample code

```
ITMGContext.GetInstance(this).SetLogPath(path);
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

API prototype

```
public abstract String GetQualityTips();
```

Sample code

```
ITMGContext.GetInstance(this).GetRoom().GetQualityTips();
```

Callback Messages

Message	Description	Data
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room.	result; error
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room.	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected due to a network or another issue.	result; error
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Room members were updated.	user_list; event_id

ITMG_MAIN_EVENT_TYPE_RECONNECT_START	The reconnection to the room started.	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	The reconnection to the room succeeded.	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched.	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status was changed.	result; error_info; sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started.	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped.	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker was changed.	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A new speaker was added.	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker was lost.	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A new mic was added.	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic was lost.	result; error

ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic was changed.	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	The room quality changed.	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Voice message recording was completed.	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Voice message upload was completed.	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Voice message download was completed.	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Voice message playback was completed.	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast speech-to-text conversion was completed.	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion was completed.	result; file_ text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	Streaming speech-to-text	result; file_ text;file_id

	conversion is in progress.	
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed.	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed.	result; text;file_id

Speech-to-Text Service

Last updated：2024-01-18 15:13:51

This document describes how to integrate with and debug the GME APIs to implement speech-to-text service for Android.

Note:

This document applies to GME SDK version 2.9.

Key Considerations for Using GME

GME provides voice chat service, voice messaging and speech-to-text services and they rely on core APIs such as Init and Poll.

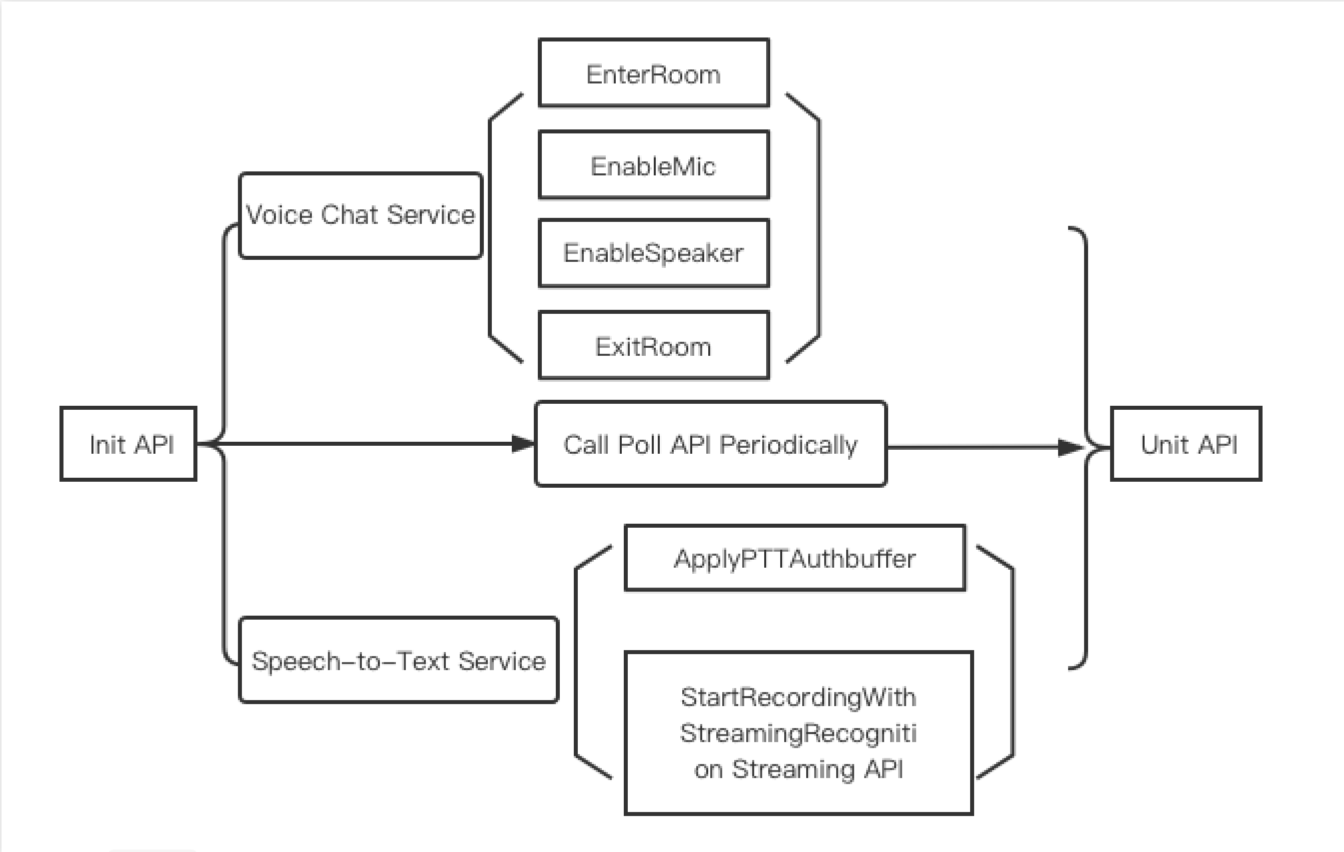
Note:

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.

Non-streaming speech-to-text API **SpeechToText()**: There can be up to 10 concurrent requests per account.

Streaming speech-to-text API **StartRecordingWithStreamingRecognition()**: There can be up to 50 concurrent requests per account.

Voice chat streaming speech-to-text API **StartRealTimeASR()**: There can be up to 50 concurrent requests per account.



Notes on the `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.

The billing will not start after initialization. Receiving or sending a voice message in speech-to-text service is counted as a voice message DAU.

Directions

1. Initializing GME
2. Calling Poll periodically to trigger callbacks
3. Initializing authentication
4. Starting streaming speech-to-text conversion
5. Stopping recording
6. Uninitializing GME

Important notes

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `QAVError.OK` will be returned with the value being `0` .

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error codes, see Error Codes.

Voice message for Android class

Class	Description
ITMGContext	Core APIs
ITMGPTT	Voice messaging and speech-to-text APIs

Core APIs

Before the initialization, the SDK is in the uninitialized status, and **you need to initialize it through the** `Init` API before you can use the voice chat and speech-to-text services.
Call the `Init` API before calling any APIs of GME.
If you have any questions when using the service, see General.

API	Description
Init	Initializes GME.
Poll	Triggers an event callback.
Pause	Pauses the system.
Resume	Resumes the system.
Uninit	Uninitializes GME.

Note:
If you need to switch the account, please call `UnInit` to uninitialize the SDK. No fee is incurred for calling Init API.

Getting singleton

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

Sample code

```
import com.tencent.TMG.ITMGContext;
ITMGContext.getInstance(this);
```

Registering 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.

Function prototype

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

```
static public abstract class ITMGDelegate {
    public void OnEvent(ITMG_MAIN_EVENT_TYPE type, Intent data){}
}
```

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

Sample code

Register the callback function to the SDK before room entry.

```
private ITMGContext.ITMGDelegate itmgDelegate = null;
itmgDelegate = new ITMGContext.ITMGDelegate() {
    @Override
    public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
        if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM == typ
        {
            // Analyze the returned data
            int nErrCode = data.getIntExtra("result" , -1);
            String strErrMsg = data.getStringExtra("error_info");
                }
        }
}
```

Function prototype

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

Parameter	Type	Description
delegate	ITMGDelegate	SDK callback function

Sample code

```
ITMGContext.GetInstance(this).SetTMGDelegate(itmgDelegate);
```

Initializing the SDK

This API is used to initialize the GME service. We recommend that you call it when initializing the application. No fees are incurred for calling this API.

For more information on how to get the `sdkAppID` parameter, see Activating Services.

`openID` uniquely identifies a user with the rules stipulated by you. It must be unique in the application and can only be in Int64 type.

Note:

The Init API must be called in the same thread with other APIs. It is recommended to call all APIs in the main thread.

Function prototype

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

Parameter	Type	Description
sdkAppId	String	`AppId` from the GME console
openId	String	`OpenId` can only be in Int64 type, which is passed in after being converted to a string.

Returned Value	Description
QAVError.OK= 0	The SDK was initialized successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. We recommend that you delete it and then import it again.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is only a reminder but will not cause an initialization failure.

If this error is reported during integration, check the integrity and version of the SDK file as prompted.

If this error is returned after executable file export, ignore it and avoid displaying it on the UI.

Sample code

```
String sdkAppID = "14000*****";
String openID = "100";
int ret = ITMGContext.GetInstance(this).Init(sdkAppId, openId);
if(ret != 0){
                Log.e(TAG,"SDK initialization failed");
}
```

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.
You can refer to the `EnginePollHelper.java` file in the demo.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

Function prototype

```
public abstract int Poll();
```

Sample code

```
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);
    }
};
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause.
If you need to pause the audio when switching to the background, you can call the `Pause` API in the listening code used to switch to the background, and call the `Resume` API in the listening event used to resume the foreground.

Function prototype

```
public abstract int Pause();
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

Function prototype

```
public abstract int Resume();
```

Uninitializing SDK

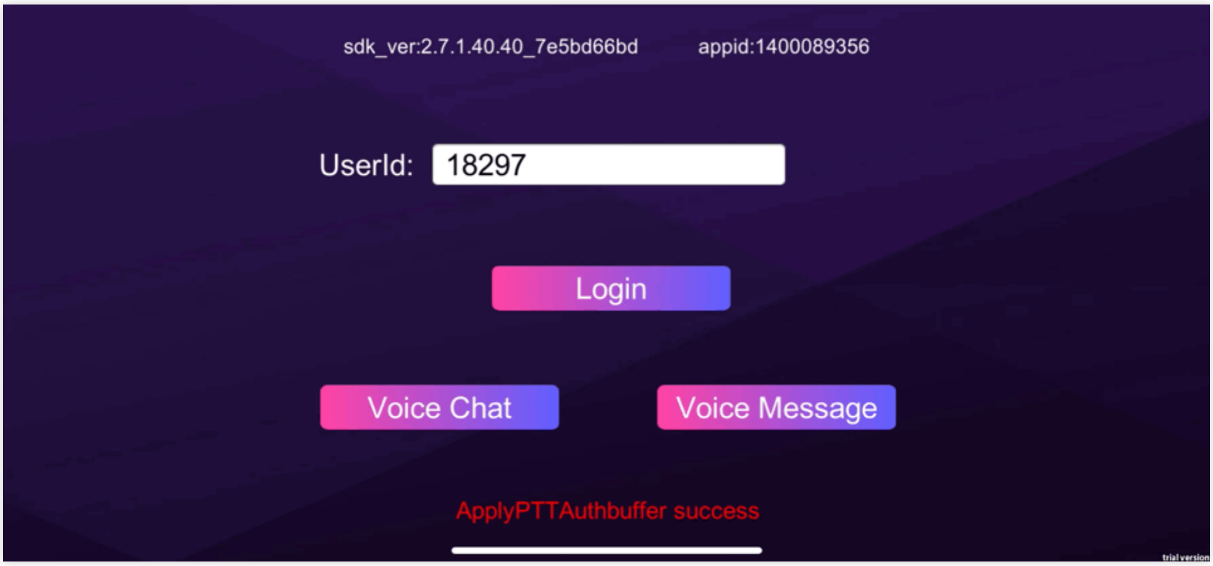
This API is used to uninitialize the SDK to make it uninitialized. **Switching accounts requires uninitialization**.

Function prototype

```
public abstract int Uninit();
```

Voice Messaging and Speech-to-Text

Voice messaging refers to recording and sending a voice message. At the same time, the voice message can be converted to text and translated, as shown below:

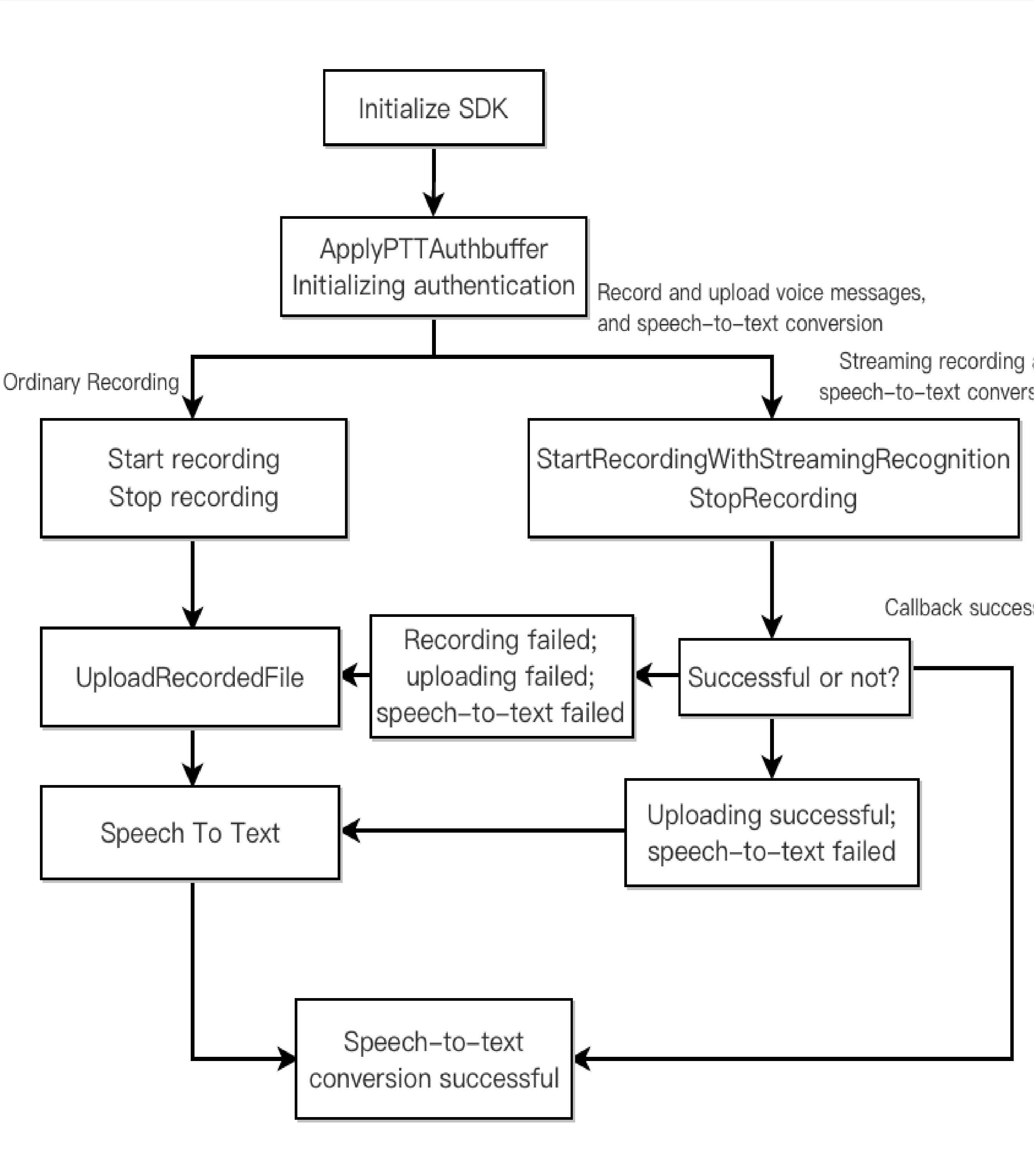


Note:

We recommend that you use the streaming speech-to-text service.

You do not need to enter a voice chat room when using the voice messaging service.

Voice messaging and speech-to-text conversion flowchart



Integrating Voice Messaging and Speech-to-Text Service

Voice messaging and speech-to-text APIs

API	Description

ApplyPTTAuthbuffer	Initializes authentication.
SetMaxMessageLength	Specifies the maximum duration of a voice message.
StartRecording	Starts recording.
StartRecordingWithStreamingRecognition	Starts streaming recording.
PauseRecording	Pauses recording.
ResumeRecording	Resumes recording.
StopRecording	This API is used to stop audio recording.
CancelRecording	Cancels recording.
GetMicLevel	Gets the real-time mic volume level.
SetMicVolume	Sets the recording volume level.
GetMicVolume	Gets the recording volume level.
GetSpeakerLevel	Gets the real-time speaker volume level.
SetSpeakerVolume	Sets the playback volume level.
GetSpeakerVolume	Gets the playback volume level.
UploadRecordedFile	Uploads an audio file.
DownloadRecordedFile	Downloads an audio file.
PlayRecordedFile	Plays back an audio file.
StopPlayFile	Stops playing back an audio file.
GetFileSize	Gets the audio file size.
GetVoiceFileDuration	Gets the audio file duration.
SpeechToText	Converts speech to text.

Maximum recording duration

The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum recording duration to 10 seconds, please call the `SetMaxMessageLength` API to set it after initialization.

Initializing the SDK

Before the initialization, the SDK is in the uninitialized status, and you need to initialize it through the `Init` API before you can use the voice chat and voice message services.
If you have any questions when using the service, see Speech-to-text Conversion.

Authentication information

Generate `AuthBuffer` for encryption and authentication of relevant features. For release in the production environment, use the backend deployment key as detailed in Authentication Key. To get the authentication key for voice messaging and speech-to-text services, the room ID parameter must be set to null.

Function prototype

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

Parameter	Type	Description
appId	int	`AppId` from the Tencent Cloud console
roomId	string	Room ID, which must be set to null.
openId	string	User ID, which is the same as `openId` during initialization.
key	string	Permission key from the Tencent Cloud console.

Sample code

```
import com.tencent.av.sig.AuthBuffer;// Header file
byte[] authBuffer = AuthBuffer.getInstance().genAuthBuffer(Integer.parseInt(sdkAppI
```

Authentication initialization

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

Function prototype

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

Parameter	Type	Description
authBuffer	String	Authentication

Sample code

```
byte[] authBuffer =  AuthBuffer.getInstance().genAuthBuffer(Integer.parseInt(sdkApp
ITMGContext.GetInstance(this).GetPTT().ApplyPTTAuthbuffer(authBuffer);
```

Streaming Speech Recognition

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

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

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

Sample code

```
String temple = getActivity().getExternalFilesDir(null).getAbsolutePath() + "/test_
ITMGContext.GetInstance(getActivity()).GetPTT().StartRecordingWithStreamingRecognit
```

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen for callback messages in the callback function `onEvent`. Event messages are divided into:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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 delivered event message contains the following four parameters.

Parameter	Description
result	Return code indicating whether streaming speech-to-text conversion is successful
text	Text converted from speech
file_path	Local path of the stored recording file
file_id	Backend URL address of the recording file, which will be retained for 90 days. This field is fixed at http://gme-v2-.

Note:
The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

Error Code	Description	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 conversion 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.

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNIT
            {
                /Callback for streaming voice message and speech-to-text
                Handler mainHander = new Handler(Looper.getMainLooper());
                mainHander.post(new Runnable() {
                    @Override
```

```
                    public void run() {
                if (nErrCode ==0) {

                    String textString = templeData.getStringExtra("text");
                    EditText _editText = (EditText) root.findViewById(R.id.edit_aud
                    _editText.setText(textString);

                    recordfilePath = templeData.getStringExtra("file_path");
                    mEditTextfiletoupload.setText(recordfilePath);

                    donwLoadUrlPath = templeData.getStringExtra("file_id");
                    mEditTextDownloadurl.setText(donwLoadUrlPath);

                    Log.e(TARGET, "STREAMINGRECOGNITION" + "nErrCode=" + nErrCode +
                }
                else
                {

                    Toast.makeText(getActivity(), String.format("Streaming speech-t
                    Log.e(TARGET, "Streaming speech-to-text conversion failed. Erro
                }
                    }
                });
            }
}
```

Voice Message Recording

The recording process is as follows: start recording > stop recording > return recording callback > start the next recording.

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

Function prototype

```
public abstract int SetMaxMessageLength(int msTime);
```

Parameter	Type	Description
msTime	int	Audio duration in ms. Value range: 1000 < `msTime` < 58000.

Sample code

```
ITMGContext.GetInstance(this).GetPTT().SetMaxMessageLength(msTime);
```

Starting recording

This API is used to start recording. The recording file must be uploaded first before you can perform operations such as speech-to-text conversion. **To stop recording, call `StopRecording`** .

Function prototype

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

Parameter	Type	Description
filePath	String	Path of the stored audio file

Sample code

```
ITMGContext.GetInstance(this).GetPTT().StartRecording(filePath);
```

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

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

Sample code

```
ITMGContext.GetInstance(this).GetPTT().StopRecording();
```

Callback for recording start

A callback will be executed through a delegate function to pass a message when recording is completed.
To stop recording, call `StopRecording` . The callback for recording start will be returned after the recording is stopped.
The callback function `OnEvent` will be called after recording is started. The event message `ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE` will be returned, which will be identified in the `OnEvent`

function.

The passed parameter includes `result` and `file_path` .

Error codes

Error Code	Reasons	Suggested Solution
4097	Empty parameters.	Check whether the API parameters in the code are correct.
4098	An initialization error occurred.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Make sure that the SDK recording feature is used at the right time.
4100	No audio data is captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	The mic permission is required for using the SDK. To add the permission, see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE =
            {
                    // Callback for recording start
                     if (nErrCode ==0)
                        {
                            recordfilePath = templeData.getStringExtra("file_path")
                            mEditTextfiletoupload.setText(recordfilePath);
                        }
            }
}
```

Pausing recording

This API is used to pause recording. If you want to resume recording, call the `ResumeRecording` API.

Function prototype

```
public abstract int PauseRecording();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().PauseRecording();
```

Resuming recording

This API is used to resume recording.

Function prototype

```
public abstract  int ResumeRecording();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().ResumeRecording();
```

Canceling recording

This API is used to cancel recording. There is no callback after cancellation.

Function prototype

```
public abstract int CancelRecording();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().CancelRecording();
```

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. An int-type value will be returned. Value range: 0-200.

This API is different from the voice chat API and is in `ITMGPTT.java` .

Function prototype

```
public abstract int GetMicLevel();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetMicLevel();
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

This API is different from the voice chat API and is in `ITMGPTT.java`.

Function prototype

```
public abstract int SetMicVolume(int volume);
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().SetMicVolume(100);
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. An int-type value will be returned. Value range: 0-200.

This API is different from the voice chat API and is in `ITMGPTT.java`.

Function prototype

```
public abstract int GetMicVolume();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetMicVolume();
```

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. An int-type value will be returned. Value range: 0-200.

This API is different from the voice chat API and is in `ITMGPTT.java`.

Function prototype

```
public abstract int GetSpeakerLevel();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetSpeakerLevel();
```

Setting the playback volume of voice message

This API is used to set the playback volume of voice message. Value range: 0-200.

This API is different from the voice chat API and is in `ITMGPTT.java`.

Function prototype

```
public abstract int SetSpeakerVolume(int volume);
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().SetSpeakerVolume(100);
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice message. An int-type value will be returned. Value range: 0-200.

This API is different from the voice chat API and is in `ITMGPTT.java`.

Function prototype

```
public abstract int GetSpeakerVolume();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetSpeakerVolume();
```

Voice Message Playback

Playing back audio

This API is used to play back audio.

Function prototype

```
public abstract int PlayRecordedFile(String filePath);public abstract int PlayRecor
```

Parameter	Type	Description

downloadFilePath	String	Local audio file path
voicetype	int	Voice changing type. For more information, see Voice Changing.

Error codes

Error Code	Reasons	Suggested Solution
20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
ITMGContext.GetInstance(this).GetPTT().PlayRecordedFile(downloadFilePath);
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameter includes `result` and `file_path` .

Error codes

Error Code	Reasons	Suggested Solution
20481	An initialization error occurred.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	Empty parameters.	Check whether the API parameters in the code are correct.
20484	An internal error occurred.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if(ITMGContext.ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE== type)
```

```
                {
            // Callback for audio playback
        }
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

Function prototype

```
public abstract int StopPlayFile();
```

Sample code

```
ITMGContext.GetInstance(this).GetPTT().StopPlayFile();
```

Getting audio file size

This API is used to get the size of an audio file.

Function prototype

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

Parameter	Type	Description
filePath	String	Path of the audio file, which is a local path.

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetFileSize(path);
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

Function prototype

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

Parameter	Type	Description

filePath	String	Path of the audio file, which is a local path.

Sample code

```
ITMGContext.GetInstance(this).GetPTT().GetVoiceFileDuration(path);
```

Voice Message Upload and Download

Uploading an audio file

This API is used to upload an audio file.

Function prototype

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

Parameter	Type	Description
filePath	String	Path of the uploaded audio file, which is a local path.

Sample code

```
ITMGContext.GetInstance(this).GetPTT().UploadRecordedFile(filePath);
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Error codes

Error Code	Reasons	Suggested Solution
8193	An error occurred while accessing the file during upload.	Ensure the existence of the file and the validity of the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.

8195	Network error	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device network can normally access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device network can normally access the internet.
8200	`appinfo` is not set.	Check whether the `apply` API is called or whether the input parameter is not specified or null.

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if(ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE==
            {
               // Callback for audio file upload completion
            }
}
```

Downloading the audio file

This API is used to download an audio file.

Function prototype

```
public abstract int DownloadRecordedFile(String fileID, String filePath);
```

Parameter	Type	Description
fileID	String	File URL
downloadFilePath	String	Local path of the saved file

Sample code

```
ITMGContext.GetInstance(this).GetPTT().DownloadRecordedFile(url,path);
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function. The passed parameters include `result` 、 `file_path` and `file_id` .

Error codes

Error Code	Reasons	Suggested Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
12291	A network storage system exception occurred.	The server failed to get the audio file. Check whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	A server file system error occurred.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed while getting the download parameters.	Check whether the device can access the internet.
12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	`appinfo` is not set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if(ITMGContext.ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE== type)
        {
            //Download succeeded
    }
}
```

Speech-to-Text Service

Converting audio file to text

This API is used to convert a specified audio file to text.

Function prototype

```
public abstract int SpeechToText(String fileID);
```

Parameter	Type	Description
fileID	String	Audio file URL

Sample code

```
ITMGContext.GetInstance(this).GetPTT().SpeechToText(fileID);
```

Translating audio file into text in specified language

This API can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation.

Function prototype

```
public abstract int SpeechToText(String fileID, String speechLanguage,String transl
```

Parameter	Type	Description
fileID	String	URL of the audio file, which will be retained on the server for 90 days.
speechLanguage	String	The language in which the audio file is to be converted into text. For parameters, see Language Parameter Reference List.
translatelanguage	String	The language into which the audio file is to be translated into text. For parameters, see Language Parameter Reference List. (This parameter is currently unavailable. Enter the same value as that of `speechLanguage` .)

Sample code

```
ITMGContext.GetInstance(this).GetPTT().SpeechToText(fileID,"cmn-Hans-CN","cmn-Hans-
```

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result`, `file_path` and `text` (recognized text).

Error codes

Error Code	Reasons	Suggested Solution
32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	Network connection failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	`appinfo` is not set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	The speech-to-text conversion parameter is incorrect.	Check whether the API parameter `fileid` in the code is empty.
32785	A speech-to-text translation error occurred.	An error occurred in the voice messaging and speech-to-text feature on the backend. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.

Sample code

```
public void OnEvent(ITMGContext.ITMG_MAIN_EVENT_TYPE type, Intent data) {
    if(ITMGContext.ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE == type)
            {
            //Recognized audio file successfully
     }
}
```


Advanced APIs

Getting version number

This API is used to get the SDK version number for analysis.

Function prototype

```
public abstract String GetSDKVersion();
```

Sample code

```
ITMGContext.GetInstance(this).GetSDKVersion();
```

Checking mic permission

This API is used to return the mic permission status.

Function prototype

```
public abstract ITMG_RECORD_PERMISSION  CheckMicPermission();
```

Parameter description

Parameter	Value	Description
ITMG_PERMISSION_GRANTED	0	The mic permission is granted.
ITMG_PERMISSION_Denied	1	Mic disabled.
ITMG_PERMISSION_NotDetermined	2	No authorization box has been popped up to request the permission.
ITMG_PERMISSION_ERROR	3	An error occurred while calling the API.

Sample code

```
ITMGContext.GetInstance(this).CheckMicPermission();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

Function prototype

```
public abstract int SetLogLevel(int levelWrite, int levelPrint);
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to log. Default value: `TMG_LOG_LEVEL_INFO` .
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: `TMG_LOG_LEVEL_ERROR` .

ITMG_LOG_LEVEL

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContext.GetInstance(this).SetLogLevel(TMG_LOG_LEVEL_INFO,TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path, which is /sdcard/Android/data/xxx.xxx.xxx/files by default.

Function prototype

```
public abstract int SetLogPath(String logDir);
```

Parameter	Type	Description

logDir	String	Path

Sample code

```
ITMGContext.GetInstance(this).SetLogPath(path);
```

Callback Messages

Message list

Message	Description
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	PTT recording is completed.
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	PTT upload is completed.
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	PTT download is completed.
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	PTT playback is completed.
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Speech-to-text conversion is completed.

Data list

Message	Data	Example
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	result; file_path	{"file_path":
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	result; file_path;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	result; file_path;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	result; file_path	{"file_path":
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	result; text;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	result; file_path; text;file_id	{"file_id":"",'

Project Export

Last updated：2024-01-18 15:13:51

This document mainly describes the notes on exporting the Android project so that the Android developers can easily debug and integrate the APIs for Game Multimedia Engine (GME).

Project Export

The GME SDK provides lib files for v7a, v8a, x86, and x86_64 by default. Please delete unnecessary files as needed.

Warning

If the .so file of the corresponding architecture is missing during the running of the Android system device, the system will crash.

Configuring Permissions

Required permissions

Add the following permissions in the `AndroidManifest.xml` file of the project:

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

Adding permissions as needed

Add the following permissions in the `AndroidManifest.xml` file of the project as needed:

Read/Write

Bluetooth permission

The read/write permission is not required. Determine whether to add it according to the following rules:

If you use the default log path (`/sdcard/Android/data/xxx.xxx.xxx/files`), it means that you do not call `SetLogPath` and do not need the `WRITE_EXTERNAL_STORAGE` permission.

If you call the `SetLogPath` API to set the log path to an external storage device, and the storage path of the voice message recording is an external storage device, you need to apply for the `WRITE_EXTERNAL_STORAGE` permission to the user and get the user's approval.

You don't need to add this permission for devices on Android 6 or later.


```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Add the Bluetooth permission according to the following rules:

If `targetSDKVersion` in the project is v30 or earlier:

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
```

If `targetSDKVersion` in the project is v31 or later and GME is earlier than v2.9.6:

```
<uses-permission android:name="android.permission.BLUETOOTH"
android:maxSdkVersion="30" />
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
```

App obfuscation

If you want to obfuscate the code, configure the following:

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

Note that after **v2.9.0**, obfuscation is required with the following configurations.

```
-dontwarn com.gme.**
-keep class com.gme.** { *;}
-keepclassmembers class com.gme.**{*;}
```

Advanced Android Configuration

According to Behavior changes: all apps for Android 9 on the Android Developers platform, Android 9 limits background apps' access to user inputs and sensor data, that is, apps running in the background cannot access the mic or camera.

If Android 9 users need to continue capturing audio or video after locking the screen, a service can be initiated before the screen is locked or the app is brought to the background and terminated before the screen is unlocked or the app is brought to the foreground.

Android Project Export FAQs

Project problems occurred during or after the export of the executable files:

After the application is exported to an Android phone, when I open the application, an error message pops up indicating that the application is not supported by the device. What should I do?

What should I do if the screen goes black when I try to open an application after integrating the GME SDK and exporting an APK file?

SDK for macOS

Project Configuration

Last updated：2024-01-18 15:13:51

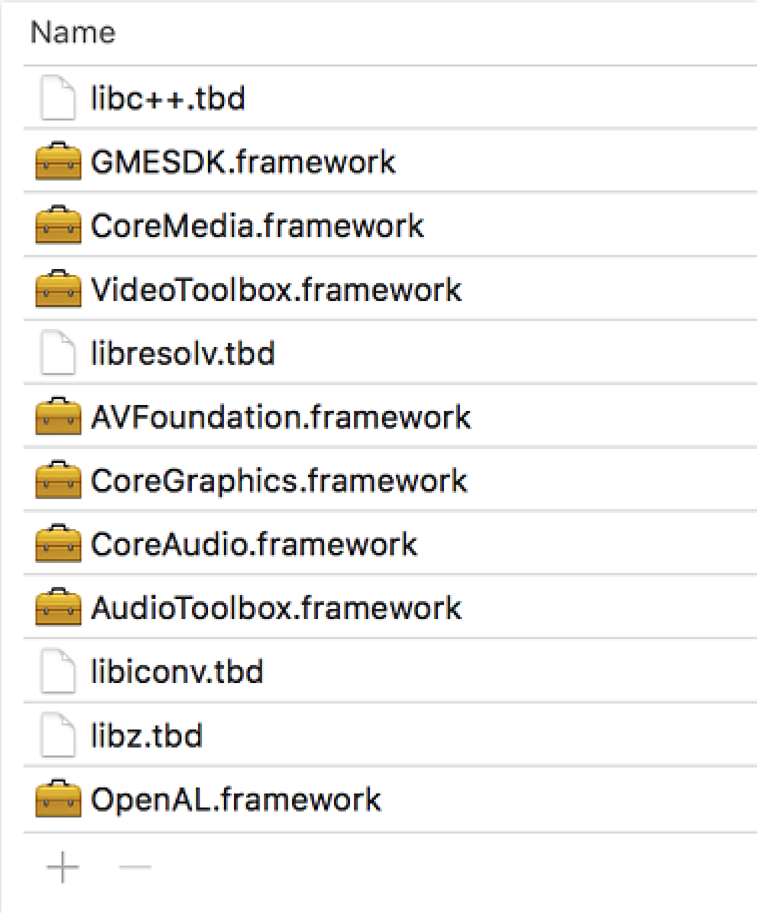
This document describes how to configure a macOS project for the GME APIs for macOS.

SDK Preparations

1. Download the applicable demo and SDK. For more information, please see Download Guide.
2. Decompress the obtained SDK resources.
3. The extracted `GMESDK.framework` is the resource related to GME.

Configuration Guide

Add the following dependent libraries to `Link Binary With Libraries` in Xcode as needed and configure `Framework Search Paths` to point to the directory where the SDK is located as shown below:


Name
libc++.tbd
GMESDK.framework
CoreMedia.framework
VideoToolbox.framework
libresolv.tbd
AVFoundation.framework
CoreGraphics.framework
CoreAudio.framework
AudioToolbox.framework
libiconv.tbd
libz.tbd
OpenAL.framework
+ —

Voice Chat API

Last updated：2024-01-18 15:13:51

This document describes how to integrate with Game Multimedia Engine (GME) on macOS.

Note:

This document applies to GME SDK version 2.9.

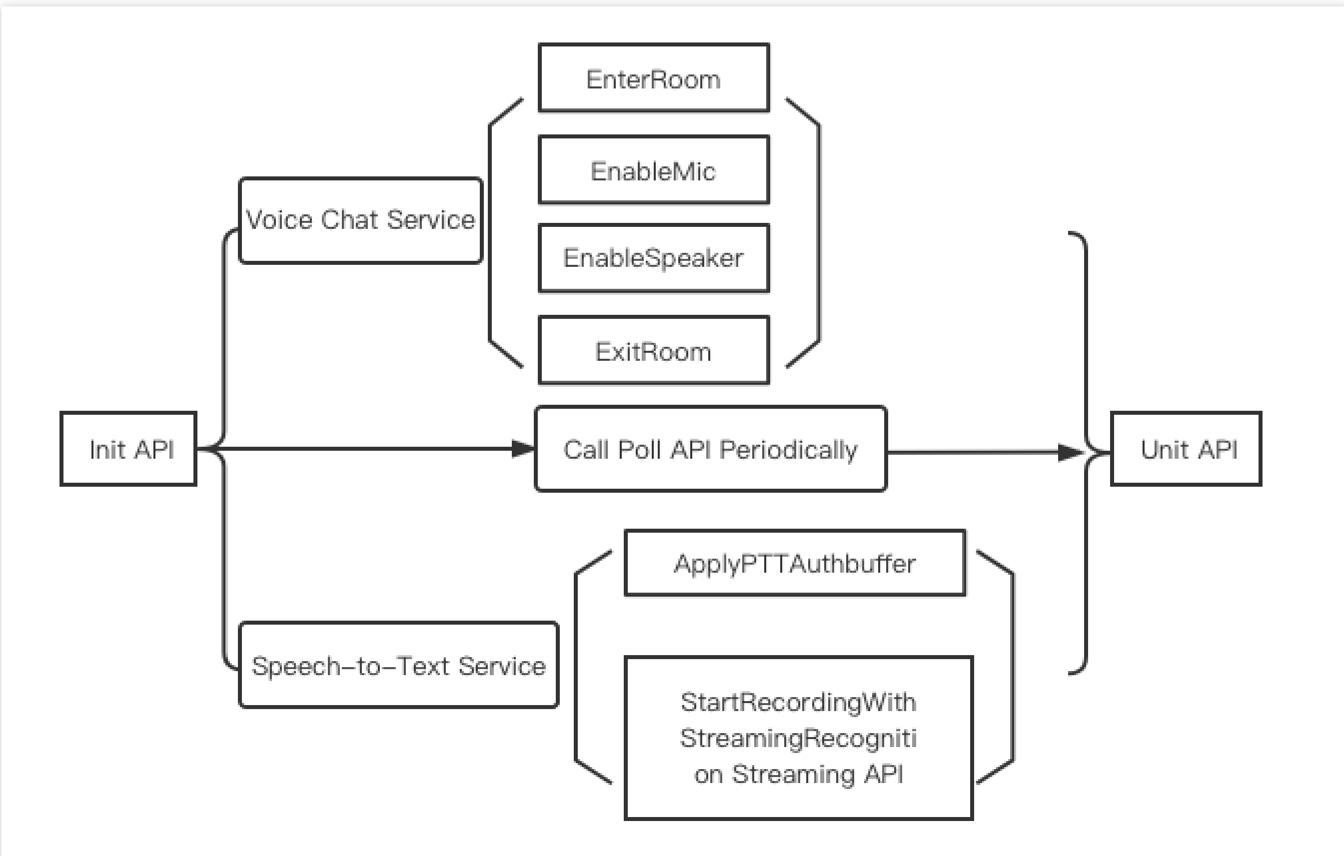
Considerations

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**.

Billing will not start after initialization. After you call Entering a voice chat room to enter the room successfully, the billing will start.



Directions

1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Entering a voice chat room, API: EnterRoom
4. Enabling the microphone, API: EnableMic
5. Enabling the speaker, API: EnableSpeaker
6. Exiting a voice room, API: ExitRoom
7. Uninitializing GME, API: UnInit

Important notes

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error code, please see Error Codes.

APIs

```
@class ITMGRoom;//Room APIs
@class ITMGAudioCtrl;//Audio APIs
@class ITMGAudioEffectCtrl;//Sound effect, accompaniment APIs
```

Key APIs

Before the initialization, the SDK is in the uninitialized status, and **you need to initialize it through the `Init` API** before you can use the voice chat and speech-to-text services.

You need to call the `Init` API before calling any APIs of GME.

If you have any questions when using the service, please see General FAQs.

API	Description
InitEngine	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME

Getting singleton

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

Function prototype

```
ITMGContext ITMGDelegate <NSObject>
```

Sample code

```
ITMGContext* _context = [ITMGContext GetInstance];
_context.TMGDelegate =self;
```

Message delivery

The API callback messages is processed in `OnEvent` . For the message type, please see `ITMG_MAIN_EVENT_TYPE` . The message content is a dictionary for parsing the API callback contents.

Function prototype

```
- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary*)data;
```

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
        NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
        switch (eventType) {
            //Identify `eventType`
            }
    }
```

Initializing SDK

This API is used to initialize the GME service. It is recommended to call it when initializing the application. No fee is incurred for calling this API.

For more information on how to get the `sdkAppID` parameter, see Voice Service Activation Guide.

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

Note:

The SDK must be initialized before a user can enter a voice chat room.

The Init API must be called in the same thread with other APIs. It is recommended to call all APIs in the main thread.

Function prototype

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

Parameter	Type	Description
sdkAppId	String	`AppId` provided by the GME service from the Tencent Cloud console
OpenId	String	`OpenId` can only be in Int64 type, which is passed after being converted to a string.

Returned values

Returned Value	Description
QAV_OK= 0	Initialized SDK successfully.
QAV_ERR_SDK_NOT_FULL_UPDATE= 7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.
If this error is reported during integration, please check the integrity and version of the SDK file as prompted.
If this error is returned after executable file export, please ignore it and try to avoid displaying it in the UI.

Sample code

```
_openId = _userIdText.text;
_appId = _appIdText.text;
[[ITMGContext GetInstance] InitEngine:SDKAPPID openID:_openId];
```

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 EnginePollHelper.m file in Demo.

Calling the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

Function prototype

```
-(void)Poll;
```

Sample code

```
[[ITMGContext GetInstance] Poll];
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause.
If you need to pause the audio when switching to the background, you can call the `Pause` API in the listening code used to switch to the background, and call the `Resume` API in the listening event used to resume the foreground.

Function prototype

```
-(QAVResult)Pause;
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

Function prototype

```
-(QAVResult)Resume;
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **Switching accounts requires uninitialization**.

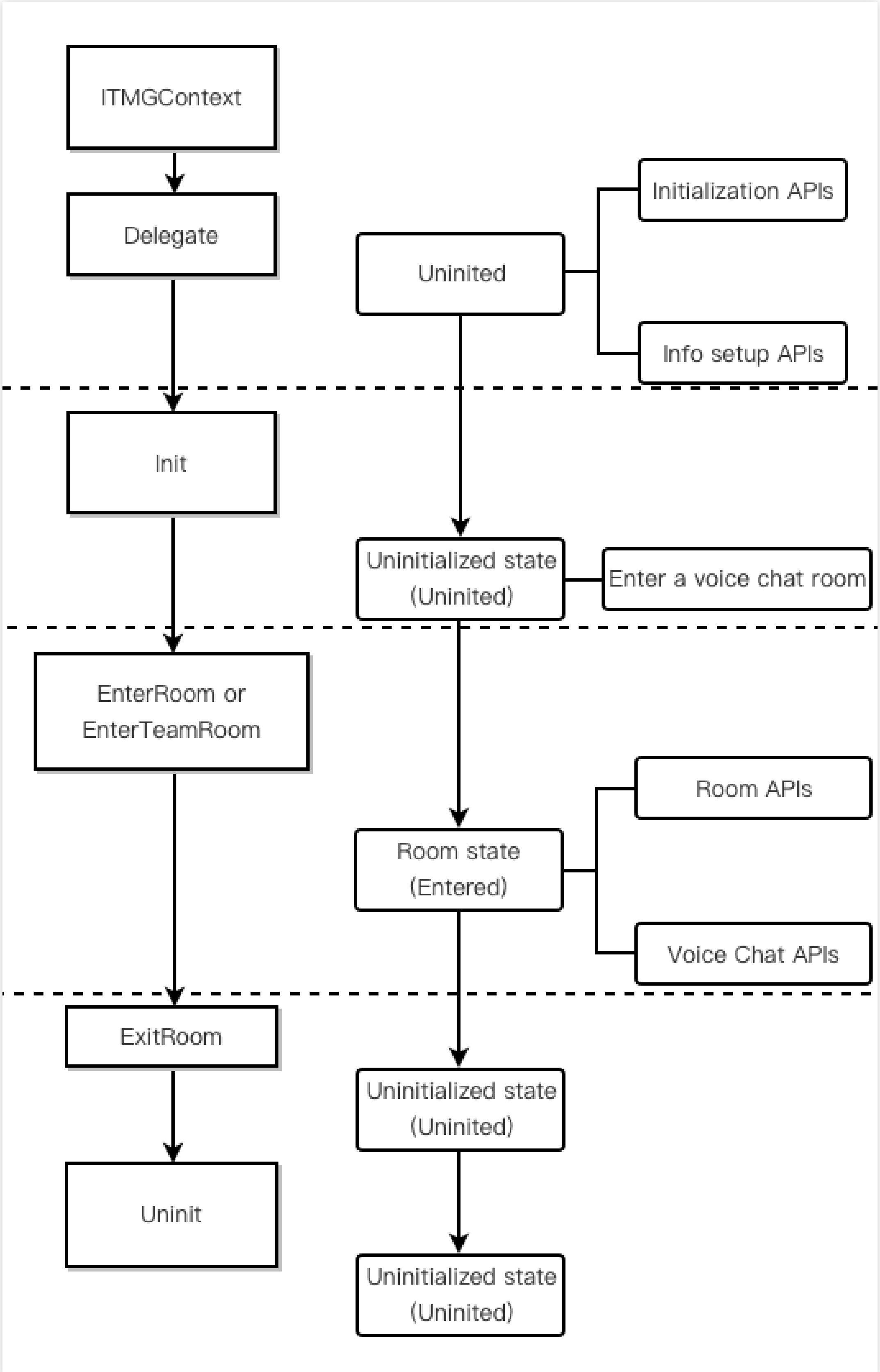
Function prototype

```
-(int)Uninit;
```

Sample code

```
[[ITMGContext GetInstance] Uninit];
```

Voice chat flowchart



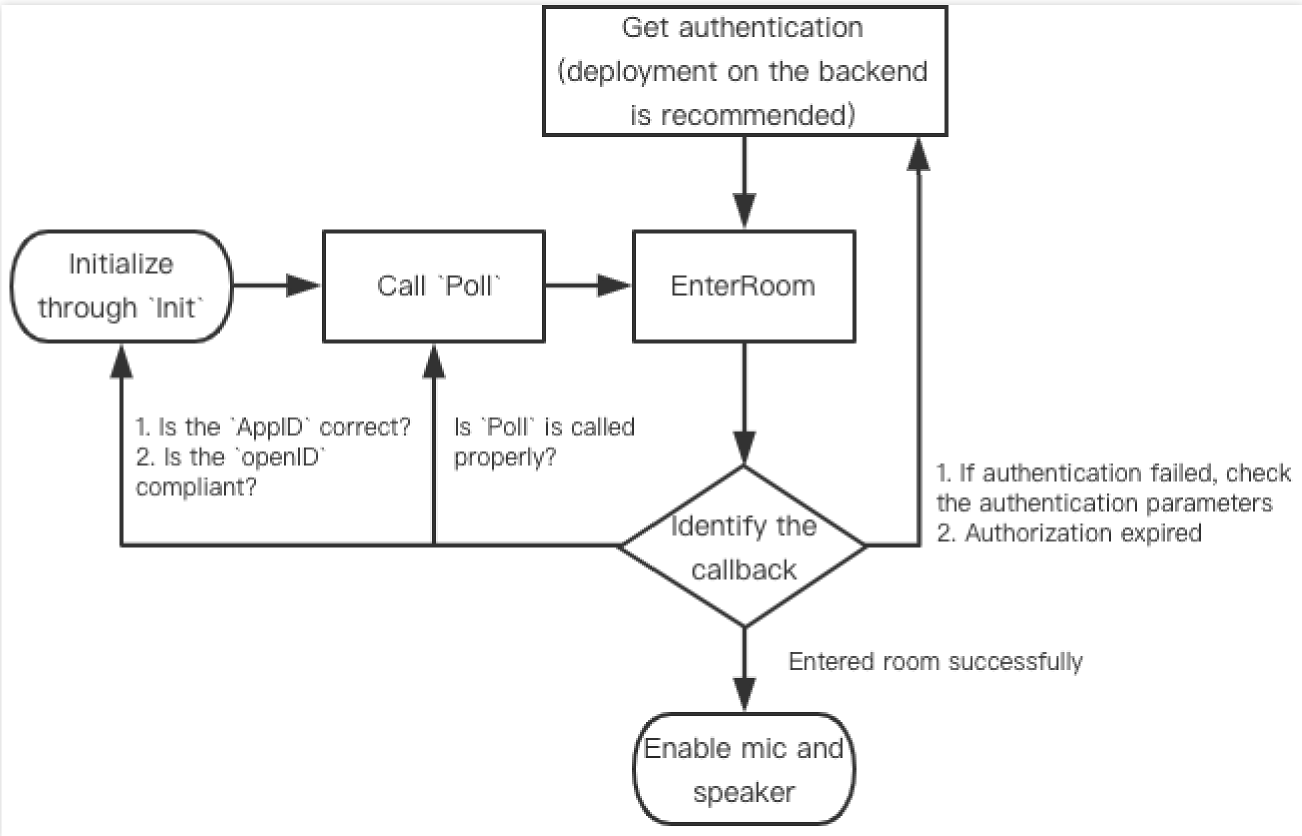
Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, please see FAQs About Voice Chat.

API	Description
GenAuthBuffer	Initializes authentication
EnterRoom	Enters room
IsRoomEntered	Indicates whether room entry is successful
ExitRoom	Exits room
ChangeRoomType	Modifies user's room audio type
GetRoomType	Gets user's room audio type

Voice chat room call flowchart



Entered the room successfully

If the room entry callback result is 0, the room entry is successful. The returned value of 0 from the `EnterRoom` API does not necessarily mean that the room entry is successful.

Authentication information

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.

Function prototype

```
@interface QAVAuthBuffer : NSObject
+ (NSData*) GenAuthBuffer:(unsigned int)appId roomId:(NSString*)roomId openID:
(NSString*)openID key:(NSString*)key;
+ @end
```

Parameter	Type	Description
appId	int	`AppId` from the Tencent Cloud console.
roomId	NSString	Room ID, which can contain up to 127 characters.
openID	NSString	User ID, which is the same as `openID` during initialization.
key	NSString	Permission key from the Tencent Cloud console.

Sample code

```
#import "GMESDK/QAVAuthBuffer.h"
NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:SDKAPPID3RD.intValue
roomId:_roomId openID:_openId key:AUTHKEY];
```

Entering a room

When a user enters a room with the generated authentication information, the `ITMG_MAIN_EVENT_TYPE_ENTER_ROOM` message will be received as a callback. Mic and speaker are not enabled by default after room entry. The returned value of `AV_OK` indicates a success.

The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will become smooth sound quality. Only when a member in the room calls the `ChangeRoomType` API, the audio type of the room will be changed.

For more information on how to choose a room audio type, please see Sound Quality Selection.

Function prototype

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

Parameter	Type	Description
roomId	NSString	Room ID, which can contain up to 127 characters
roomType	int	Room audio type
authBuffer	NSData	Authentication key

Sample code

```
[[ITMGContext GetInstance] EnterRoom:_roomId roomType:_roomType
authBuffer:authBuffer];
```

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 the billing continues if the client goes offline when using Voice Chat?

Sample code

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

```
-(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_ENTER_ROOM:
        {
            int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
            NSString* error_info = [data objectForKey:@"error_info"];
                //Receive the event of successful room entry
        }
            break;
    }
}
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnected callback prompt `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code Value	Cause and Suggested Solution
7006	Authentication failed: 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.

Exiting the room

This API is called to exit the current room. It is an async API. The returned value of `AV_OK` indicates a successful async delivery.

Note:

If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API during API call; instead, you can directly call the API.

Function prototype

```
-(int)ExitRoom
```

Sample code

```
[[ITMGContext GetInstance] 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`.

Sample code

```
-(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;
    }
}
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"","result":0}

Determining whether user has entered room

This API is used to determine whether the user has entered a room. A bool-type value will be returned. The call is invalid during the process of room entry.

Function prototype

```
-(BOOL)IsRoomEntered;
```

Sample code

```
[[ITMGContext GetInstance] IsRoomEntered];
```

Switching room

User can call this API to quickly switch the voice chat room after entering the room. After the room is switched, the device is not reset, that is, if the microphone is already enabled in this room, the microphone will keep enabled after the room is switched.

The callback for quickly switching rooms is `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM` , and the fields are `error_info` and `result` .

API prototype

```
-(int) SwitchRoom:(NSString *)roomID  authBuffer:(NSData*)authBuffer;
```

Type descriptions

Parameter	Type	Description
targetRoomID	NSString *	ID of the room to enter
authBuffer	NSData*	Generates a new authentication with the ID of the room to enter

Callback sample code

```
- (IBAction)swichRoom:(id)sender {
    NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:_appId.intValue 
roomID:_roomIdText.text openID:_openId key:_key];
    [[[ITMGContext GetInstance]GetRoom]SwitchRoom:_roomIdText.text 
authBuffer:authBuffer];
}

-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSString* log =[NSString stringWithFormat:@"OnEvent:%d,data:%@", 
(int)eventType, data];
```

```
    [self showLog:log];
    NSLog(@"====%@====",log);
    switch (eventType) {
    case ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM:
        {
            int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
            NSString* log = nil;
            if (result == QAV_OK) {
                log = [NSString stringWithFormat:@"switch room success."];
            } else {
                log = [NSString stringWithFormat:@"switch room failed."];
            }
            [self showLog:log];
            break;
        }
    }
}
```

Cross-room mic connection

Call this API to connect the microphones across rooms after entering the room. And the local user can communicate with the target OpenID user in the target room.

Example

User a is in room A, user b is in room B, and user a can talk with b through the cross-room API. When user c in room A speaks, users b and d in room B cannot hear. User c in room A can hear only the voice in room A and the voice of user b in room B but not other users in room B.

API prototype

```
-(int) StartRoomSharing:(NSString *)targetRoomID targetOpenID:(NSString
*)targetOpenID authBuffer:(NSData*)authBuffer;

-(int) StopRoomSharing;
```

Type descriptions

Parameter	Type	Description
targetRoomID	NSString *	ID of the room to connect mic
targetOpenID	NSString *	The target OpenID to connect mic

authBuffer	NSData*	Reserved flag. You just need to enter NULL.

Sample code

```
- (IBAction)shareRoom:(id)sender {
    if(_shareRoomSwitch.isOn){
        [[[ITMGContext GetInstance]GetRoom]StartRoomSharing:_shareRoomID.text
targetOpenID:_shareOpenID.text authBuffer:NULL];
    }else{
        [[[ITMGContext GetInstance]GetRoom]StopRoomSharing];
    }
}
}
```

Notifications of member room entry and speaking status

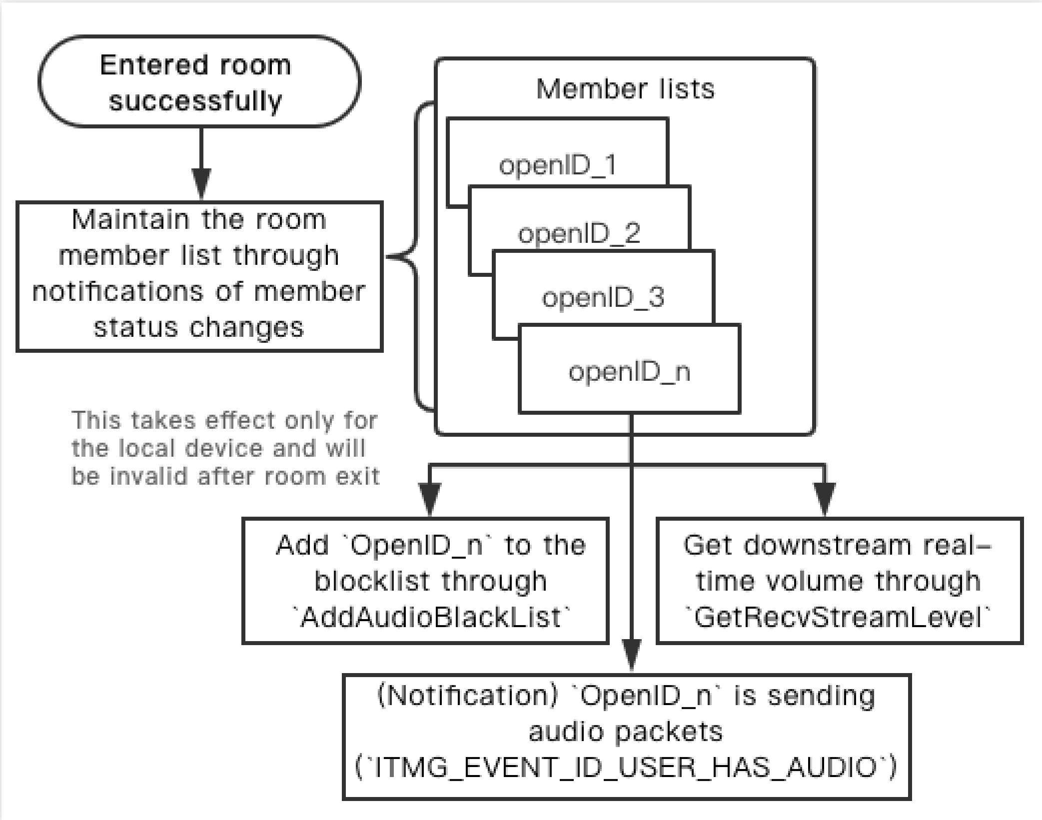
This API is used to obtain the user speaking in the room and display it in the UI, and to send a notification when someone entering or exiting the room.

Notification for this event will be sent only when the status changes. To get member status in real time, cache the notification when it is received at a higher layer. The event message is `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` , where the data contains `event_id` and `user_list` . The event message will be identified in the `OnEvent` function.

Notifications for audio events are subject to a threshold, and a notification will be sent only when this threshold is exceeded. The notification "A member has stopped sending audio packets” will be sent if no audio packets are received in more than two seconds. This event only returns the member speaking event, but not the specific volume level. If you need the specific volume levels of members in the room, use the `GetVolumeById` API.

event_id	Description	Maintenance
ITMG_EVENT_ID_USER_ENTER	A member enters the room	Member list
ITMG_EVENT_ID_USER_EXIT	A member exits the room	Member list
ITMG_EVENT_ID_USER_HAS_AUDIO	A member sends audio packets. This event can be used to determine whether a user is speaking and display the voiceprint effect. It can be called together with getRecvStreamLevel.	Chat member list
ITMG_EVENT_ID_USER_NO_AUDIO	A member stops sending audio packets	Chat member list

Room member maintenance flowchart



Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    ITMG_EVENT_ID_USER_UPDATE event_id=((NSNumber*)[data
objectForKey:@"event_id"]).intValue;
    NSMutableArray* uses = [NSMutableArray arrayWithArray: [data
objectForKey:@"user_list"]];
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_USER_UPDATE:
        {
        // Process
        //Parse the parameter to get `event_id` and `user_list`
            switch (eventID)
             {
             case ITMG_EVENT_ID_USER_ENTER:
                  // A member enters the room
                  break;
             case ITMG_EVENT_ID_USER_EXIT:
                  // A member exits the room
                break;
            case ITMG_EVENT_ID_USER_HAS_AUDIO:
```

```
                    // A member sends audio packets
                    break;
                case ITMG_EVENT_ID_USER_NO_AUDIO:
                    // A member stops sending audio packets
                    break;
                 }
            break;
            }
        }
}
```

Data details

Message	Data	Sample
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	event_id; user_list	{"event_id":0,"user_list":""}

Muting in a room

This API is used to add an ID to the audio data blacklist. This operation blocks audio from someone and only applies to the local device. A returned value of `0` indicates the call is successful. Assume that users A, B, and C are all speaking using their mic in a room:
If A blocks C, A can only hear B;
If B blocks neither A nor C, B can hear both of them;
If C blocks neither A nor B, C can hear both of them.
This API is suitable for scenarios where a user is muted in a room.

Function prototype

```
ITMGContext GetAudioCtrl -(QAVResult)AddAudioBlackList:(NSString*)openID;
```

Parameter	Type	Description
openId	NSString	ID to be blocked openid

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ] AddAudioBlackList[id]];
```

Unmuting

This API is used to remove an ID from the audio data blocklist. A returned value of `0` indicates the call is successful.

Function prototype

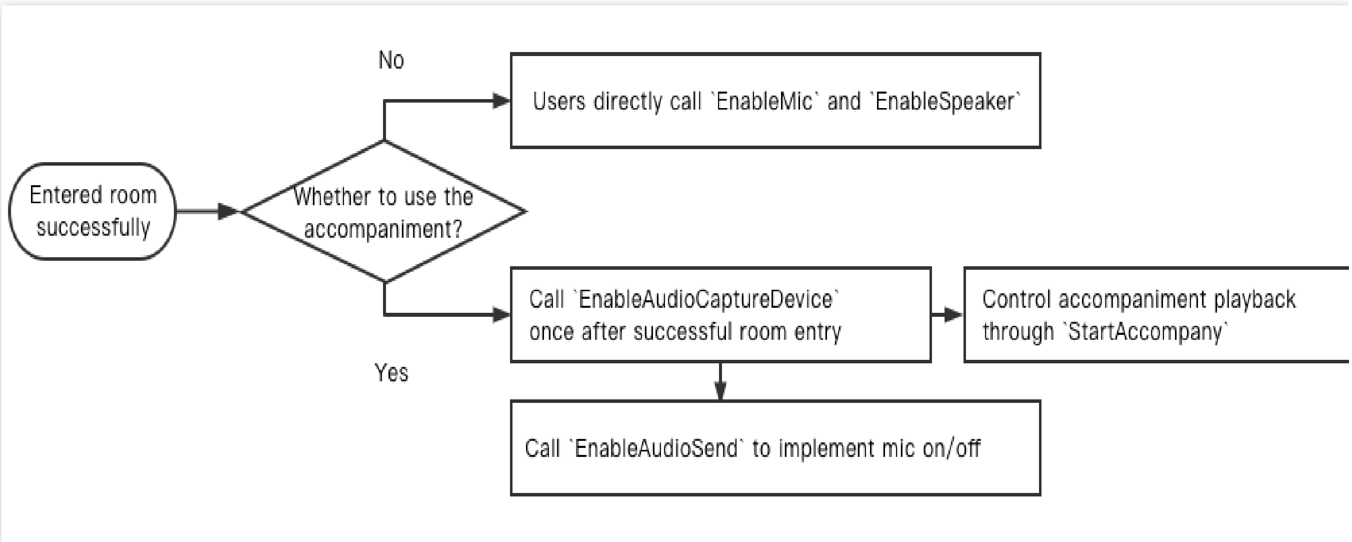
```
-(QAVResult)RemoveAudioBlackList:(NSString*)openID;
```

Parameter	Type	Description
openId	NSString	ID to be unblocked openid

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ] RemoveAudioBlackList[openId]];
```

Voice Chat Audio APIs



Notes on voice chat audio connection

The voice chat APIs can only be called after SDK initialization and room entry.

When Enable/Disable Mic/Speaker is clicked on the UI, the following practices are recommended:

For most game applications, it is recommended to call the `EnableMic` and `EnableSpeaker` APIs, which is equivalent to calling the `EnableAudioCaptureDevice/EnableAudioSend` and

`EnableAudioPlayDevice/EnableAudioRecv` APIs.
For other mobile applications (such as social networking applications), enabling/disabling a capturing device will restart both capturing and playback devices. If the application is playing back background music, it will also be interrupted. Playback will not be interrupted if the mic is enabled/disabled through control of upstreaming/downstreaming. **Calling method: call `EnableAudioCaptureDevice(true)` and `EnableAudioPlayDevice(true)` once after room entry, and call `EnableAudioSend/Recv` to send/receive audio streams when Enable/Disable Mic is clicked**.
For more information on how to release only a capturing or playback device, please see the `EnableAudioCaptureDevice` and `EnableAudioPlayDevice` .
Call the `pause` API to pause the audio engine and call the `resume` API to resume the audio engine.

Voice chat audio APIs

API	Description
EnableMic	Enables/disables mic
GetMicState	Gets mic status
EnableAudioCaptureDevice	Enables/disables capturing device
IsAudioCaptureDeviceEnabled	Gets capturing device status
EnableAudioSend	Enables/disables audio upstreaming
IsAudioSendEnabled	Gets audio upstreaming status
GetMicLevel	Gets real-time mic volume
GetSendStreamLevel	Gets real-time audio upstreaming volume
SetMicVolume	Sets mic volume
GetMicVolume	Gets mic volume
EnableSpeaker	Enables/disables speaker
GetSpeakerState	Gets speaker status
EnableAudioPlayDevice	Enables/disables playback device
IsAudioPlayDeviceEnabled	Gets playback device status
EnableAudioRecv	Enables/disables audio downstreaming
IsAudioRecvEnabled	Gets audio downstreaming status

GetSpeakerLevel	Gets real-time speaker volume
GetRecvStreamLevel	Gets real-time downstreaming audio levels of other members in room
SetSpeakerVolume	Sets speaker volume
GetSpeakerVolume	Gets speaker volume
EnableLoopBack	Enables/disables in-ear monitoring

Voice Chat Capturing APIs

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.

If accompaniment is used, please call this API as instructed in Accompaniment in Voice Chat.

EnableMic = EnableAudioCaptureDevice + EnableAudioSend

Function prototype

```
-(QAVResult)EnableMic:(BOOL)enable;
```

Parameter	Type	Description
isEnabled	boolean	To enable the mic, set this parameter to `YES` ; otherwise, set it to `NO` .

Sample code

```
// Enable mic
[[[ITMGContext GetInstance] GetAudioCtrl] EnableMic:YES];
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

Function prototype

```
-(int)GetMicState;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetMicState];
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.
This API can only be called after room entry. The device will be disabled automatically after room exit.
Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

Function prototype

```
-(QAVResult)EnableAudioCaptureDevice:(BOOL)enabled;
```

Parameter	Type	Description
enabled	BOOL	To enable the capturing device, set this parameter to `YES` , otherwise set it to `NO` .

Sample code

```
// Enable capturing device
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioCaptureDevice:enabled];
```

Getting the capturing device status

This API is used to get the status of a capturing device.

Function prototype

```
-(BOOL)IsAudioCaptureDeviceEnabled;
```

Sample code

```
BOOL IsAudioCaptureDevice = [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioCaptureDeviceEnabled];
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain mute. For more information on how to enable/disable the capturing device, please see the `EnableAudioCaptureDevice` API.

Function prototype

```
-(QAVResult)EnableAudioSend:(BOOL)enable;
```

Parameter	Type	Description
enable	BOOL	To enable audio upstreaming, set this parameter to `YES` ; otherwise, set it to `NO` .

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioSend:enabled];
```

Getting audio upstreaming status

This API is used to get the status of audio upstreaming.

Function prototype

```
-(BOOL)IsAudioSendEnabled;
```

Sample code

```
BOOL IsAudioSend = [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioSendEnabled];
```

Getting the real-time mic volume

This API is used to get the real-time mic volume. An int-type value in the range of 0-100 will be returned. It is recommended to call this API once every 20 ms.

This API is not applicable to the voice message service.

Function prototype

```
-(int)GetMicLevel;
```

```

```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetMicLevel];
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume. An int-type value in the range of 0-100 will be returned.

This API is not applicable to the voice message service.

Function prototype

```
-(int)GetSendStreamLevel();
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSendStreamLevel];
```

Setting the mic software volume

This API is used to set the mic volume. The corresponding parameter is `volume`, which is equivalent to attenuating or gaining the captured sound. 0 indicates that the audio is mute, while 100 indicates that the volume remains unchanged. The default value is 100.

This API is not applicable to the voice message service.

Function prototype

```
-(QAVResult)SetMicVolume:(int) volume;
```

Parameter	Type	Description
volume	int	Sets volume. Value range: 0-200

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] SetMicVolume:100];
```

Getting the mic software volume

This API is used to obtain the microphone volume. An "int" value is returned. Value 101 represents API SetMicVolume has not been called.

This API is not applicable to the voice message service.

Function prototype

```
-(int) GetMicVolume;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetMicVolume];
```

Voice Chat Playback APIs

Enabling or disabling the speaker

This API is used to enable/disable the speaker.

If accompaniment is used, please call this API as instructed in Accompaniment in Voice Chat**.**

EnableSpeaker = EnableAudioPlayDevice + EnableAudioRecv

Function prototype

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

Parameter	Type	Description
isEnabled	boolean	To disable the speaker, set this parameter to `NO` , otherwise set it to `YES` .

Sample code

```
// Enable the speaker
[[[ITMGContext GetInstance] GetAudioCtrl] EnableSpeaker:YES];
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, while 1, on.

Function prototype

```
-(int)GetSpeakerState;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSpeakerState];
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

Function prototype

```
-(QAVResult)EnableAudioPlayDevice:(BOOL)enabled;
```

Parameter	Type	Description
enabled	BOOL	To disable a playback device, set this parameter to `NO` , otherwise set it to `YES` .

Sample code

```
// Enable the playback device
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioPlayDevice:enabled];
```

Getting the playback device status

This API is used to get the status of a playback device.

Function prototype

```
-(BOOL)IsAudioPlayDeviceEnabled;
```

Sample code

```
BOOL IsAudioPlayDevice =  [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioPlayDeviceEnabled];
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain mute. For more information on how to enable/disable the playback device, please see the `EnableAudioPlayDevice` API.

Function prototype

```
-(QAVResult)EnableAudioRecv:(BOOL)enabled;
```

Parameter	Type	Description
enabled	BOOL	To enable audio downstreaming, set this parameter to `YES`; otherwise, set it to `NO`.

Sample code

```
[[[ITMGContext GetInstance]GetAudioCtrl ]EnableAudioRecv:enabled];
```

Getting audio downstreaming status

This API is used to get the status of audio downstreaming.

Function prototype

```
-(BOOL)IsAudioRecvEnabled;
```

Sample code

```
BOOL IsAudioRecv = [[[ITMGContext GetInstance] GetAudioCtrl]
IsAudioRecvEnabled];
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume level. An int-type value will be returned to indicate the volume level. It is recommended to call this API once every 20 ms.

Function prototype

```
-(int)GetSpeakerLevel;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSpeakerLevel];
```

Getting the real-time downstreaming audio levels of other members in room

This API is used to get the real-time audio downstreaming volume of other members in the room. An int-type value will be returned. Value range: 0-200.

Function prototype

```
-(int)GetRecvStreamLevel:(NSString*) openID;
```

Parameter	Type	Description
openID	NSString	`openId` of another member in the room

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetRecvStreamLevel:(NSString*) openId
```

Dynamic setting of the volume of a member in the room

This API is used to set the volume of a member in the room and only applies to the local device.

Function prototype

```
-(int) SetSpeakerVolumeByOpenID:(NSString *)openId volume:(int)volume;
```

Parameter description

Parameter	Type	Description

openId	String *	OpenID that needs to adjust the volume
volume	int	Range: [0-200]; Default: 100

Getting the set volume percentage

Call this API to get volume set by SetSpeakerVolumeByOpenID

API prototype

```
-(int) GetSpeakerVolumeByOpenID:(NSString *)openId;
```

Returned values

API returns volume percentage set by OpenID, 100 by default.

Setting the speaker volume

This API is used to set the speaker volume.
The corresponding parameter is volume. 0 indicates that the audio is mute, while 100 indicates that the volume remains unchanged. The default value is 100.

Function prototype

```
-(QAVResult)SetSpeakerVolume:(int)vol;
```

Parameter	Type	Description
vol	int	Sets volume. Value range: 0-200

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] SetSpeakerVolume:100];
```

Getting the speaker volume

This API is used to get the speaker volume. An int-type value will be returned to indicate the volume. 101 indicates that the `SetSpeakerVolume` API has not been called.
"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

Function prototype

```
-(int)GetSpeakerVolume;
```

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] GetSpeakerVolume];
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

Function prototype

```
-(QAVResult)EnableLoopBack:(BOOL)enable;
```

Parameter	Type	Description
enable	boolean	Specifies whether to enable in-ear monitoring.

Sample code

```
[[[ITMGContext GetInstance] GetAudioCtrl] EnableLoopBack:YES];
```

Modifying user's room audio type

This API is used to modify a user's room audio type. For the result, please see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` . The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

Function prototype

```
-(int)ChangeRoomType:(int)nRoomType;
```

Parameter	Type	Description
nRoomType	int	Target room type to be switched to. For room audio types, please see the `EnterRoom` API.

Sample code

```
[[[ITMGContext GetInstance]GetRoom ]ChangeRoomType:_roomType];
```

Getting user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, please see the `EnterRoom` API.

Function prototype

```
-(int)GetRoomType;
```

Sample code

```
[[[ITMGContext GetInstance]GetRoom ]GetRoomType];
```

Callback for modifying the room type

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result` , `error_info` , and `new_room_type` . The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description
ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	Indicates that the existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	Indicates that a user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).

ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	Indicates that a user is already in the room and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	Indicates that a room member calls the `ChangeRoomType` API to request a change of room audio type.

Sample code

```
-(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_CHANGE_ROOM_TYPE:
            NSLog(@"ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE:%@ ",data);
            int result = ((NSNumber*)[data objectForKey:@"result"]).intValue;
            int newRoomType = ((NSNumber*) [data
objectForKey:@"new_room_type"]).intValue;
            int subEventType = ((NSNumber*) [data
objectForKey:@"sub_event_type"]).intValue;
     }
}
```

Data details

Message	Data	Sa
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result;error_info;new_room_type;subEventType	{"e

The monitoring event of room call quality

The message for quality monitoring event triggered once every two seconds after room entry is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which represent the following information.

This API is used to monitor the network quality. If the user's network is poor, the business layer will remind the user to switch to a better network through the UI.

Parameter	Type	Description
weight	int	Value range: 1-50. 50 indicates excellent sound quality, 1 indicates very poor (barely usable) sound quality, and 0 represents an initial meaningless value. Generally, if the value is below 30, you can remind users that the network is poor and recommend them to switch the network.
loss	double	Upstream packet loss rate

delay	int	Voice chat delay in ms

Getting version number

This API is used to get the SDK version number for analysis.

Function prototype

```
-(NSString*)GetSDKVersion;
```

Sample code

```
[[ITMGContext GetInstance] GetSDKVersion];
```

Checking mic permission

This API is used to return the mic permission status.

Function prototype

```
-(ITMG_RECORD_PERMISSION)CheckMicPermission;
```

Parameter description

Parameter	Value	Description
ITMG_PERMISSION_GRANTED	0	Mic permission is granted.
ITMG_PERMISSION_Denied	1	Mic is disabled.
ITMG_PERMISSION_NotDetermined	2	No authorization box has been popped up to request the permission.
ITMG_PERMISSION_ERROR	3	An error occurred while calling the API.

Sample code

```
[[ITMGContext GetInstance] CheckMicPermission];
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

Function prototype

```
-(void)SetLogLevel:(ITMG_LOG_LEVEL)levelWrite (ITMG_LOG_LEVEL)levelPrint;
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
[[ITMGContext GetInstance] SetLogLevel:TMG_LOG_LEVEL_INFO TMG_LOG_LEVEL_INFO];
```

Setting the log printing path

This API is used to set the log printing path, which is /Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents by default.

Function prototype

```
-(void)SetLogPath:(NSString*)logDir;
```

Parameter	Type	Description
logDir	NSString	Path

Sample code

```
[[ITMGContext GetInstance] SetLogPath:Path];
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

Function prototype

```
-(NSString*)GetQualityTips;
```

Sample code

```
[[[ITMGContext GetInstance]GetRoom ] GetQualityTips];
```

Callback Messages

Message list

Message	Description
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	Indicates that a member enters an audio room.
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	Indicates that a member exits an audio room.
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	Indicates that a room is disconnected for network or other reasons.
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	Indicates a room type change event.
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Indicates that the room members are updated.

ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	Indicates the room quality information.

Data list

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"",
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	result; error_info	{"error_info":"",
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"w
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	result; error_info; sub_event_type; new_room_type	{"error_info":"",
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	result; error_info	{"deviceID":"{0 dd00542b47ae Audio)","error_
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	result; error_info	{"deviceID":"{0 dd00542b47ae Audio)","error_
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	result; error_info	{"deviceID":"{0 7e454093f229 Audio)","error_
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	result; error_info	{"deviceID":"{0 7e454093f229 Audio)","error_
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	user_list; event_id	{"event_id":1,"u
ITMG_MAIN_EVENT_TYPE_NUMBER_OF_USERS_UPDATE	AllUser; AccUser; ProxyUser	{"AllUser":3,"A
ITMG_MAIN_EVENT_TYPE_NUMBER_OF_AUDIOSTREAMS_UPDATE	AudioStreams	{"AudioStream
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	weight; loss; delay	{"weight":5,"los

Speech-to-Text Service

Last updated：2024-01-18 15:13:51

This document describes how to integrating with and debug the GME APIs for macOS.

Note:

This document applies to GME SDK version 2.9.

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:

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.

Non-streaming speech-to-text API ***SpeechToText()***: There can be up to 10 concurrent requests per account.

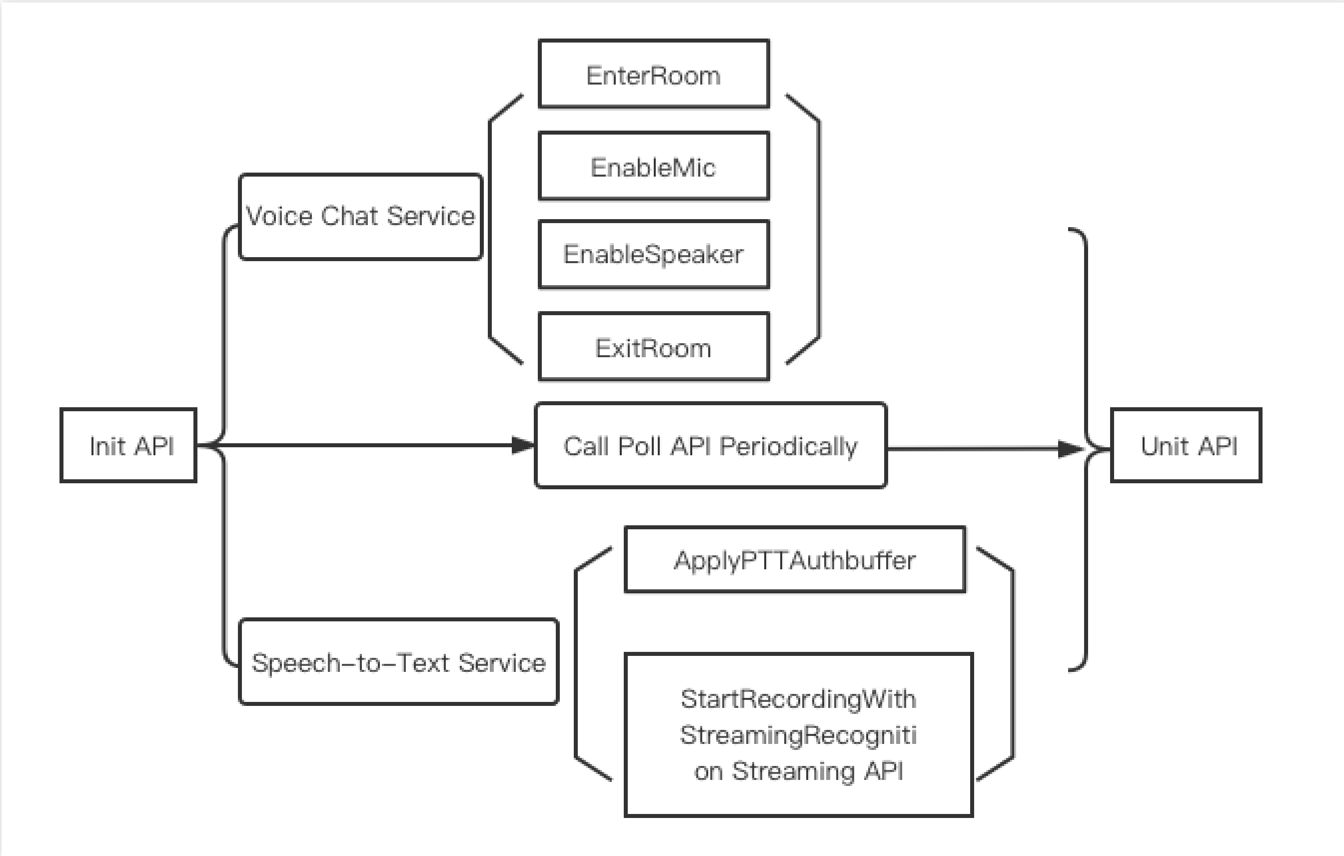
Streaming speech-to-text API ***StartRecordingWithStreamingRecognition()***: There can be up to 50 concurrent requests per account.

Real-time streaming speech-to-text API ***StartRealTimeASR()***: There can be up to 50 concurrent requests per account.

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**.

The billing will not start after initialization. Receiving or sending a voice message in speech-to-text service is counted as a voice message DAU.



Directions

1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Initializing authentication, API: ApplyPTTAuthbuffer
4. Starting streaming speech recognition, API: StartRecordingWithStreamingRecognition
5. Stop recording, API: StopRecording
6. Uninitializing GME, API: UnInit

Important notes

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error code, see Error Codes.

Core APIs

Before the initialization, the SDK is in the uninitialized status, and **you need to initialize it through the `Init` API** before you can use the voice chat and speech-to-text services.

Call the `Init` API before calling any APIs of GME.

If you have any questions when using the service, see General.

API	Description
InitEngine	Initializes GME
Poll	Triggers event callback
Pause	Pauses the system
Resume	Resumes the system
Uninit	Uninitializes GME
SetDefaultAudienceAudioCategory	Sets audio playback in background on device

Imported header files

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

Getting singleton

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

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

Sample code

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

Setting callbacks

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.

Sample code

`ITMGDelegate` is used for declaration.

```
@interface TMGDemoViewController ()<ITMGDelegate>{}
```

```
ITMGDelegate < NSObject >

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

The API callback messages is processed in `OnEvent`. For the message type, see `ITMG_MAIN_EVENT_TYPE`. The message content is a dictionary for parsing the API callback contents.

Function prototype

```
- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary*)data;
```

Sample code

```
//TMGRealTimeViewController.m
TMGRealTimeViewController ()< ITMGDelegate >


- (void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data {
    NSString *log = [NSString stringWithFormat:@"OnEvent:%d,data:%@", 
(int)eventType, data];
    [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:(%@)", result, error_info]];

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

// Refer to DispatchCenter.h and DispatchCenter.m
```

Initializing SDK

This API is used to initialize the GME service. It is recommended to call it when initializing the application. No fee is incurred for calling this API.

For more information on how to get the `sdkAppID` parameter, see Activating Services.

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

Note:

The Init API must be called in the same thread with other APIs. It is recommended to call all APIs in the main thread.

Function prototype

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

Parameter	Type	Description
sdkAppId	String	`AppId` provided by the GME service from the Tencent Cloud console
OpenId	String	`OpenId` can only be in Int64 type, which is passed after being converted to a string.

Returned Value	Description
QAV_OK= 0	Initialized SDK successfully.
QAV_ERR_SDK_NOT_FULL_UPDATE= 7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is only a reminder but will not cause an initialization failure.
If this error is reported during integration, please check the integrity and version of the SDK file as prompted.
If this error is returned after executable file export, please ignore it and try to avoid displaying it in the UI.

Sample code

```
_openId = _userIdText.text;
_appId = _appIdText.text;
[[ITMGContext GetInstance] InitEngine:SDKAPPID openID:_openId];
```

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.
For more information, see the EnginePollHelper.m file in Demo.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

Function prototype

```
-(void)Poll;
```

Sample code

```
[[ITMGContext GetInstance] Poll];
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause.
If you need to pause the audio when switching to the background, you can call the `Pause` API in the listening code used to switch to the background, and call the `Resume` API in the listening event used to resume the foreground.

Function prototype

```
-(QAVResult)Pause;
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

Function prototype

```
-(QAVResult)Resume;
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **Switching accounts requires uninitialization**.

Function prototype

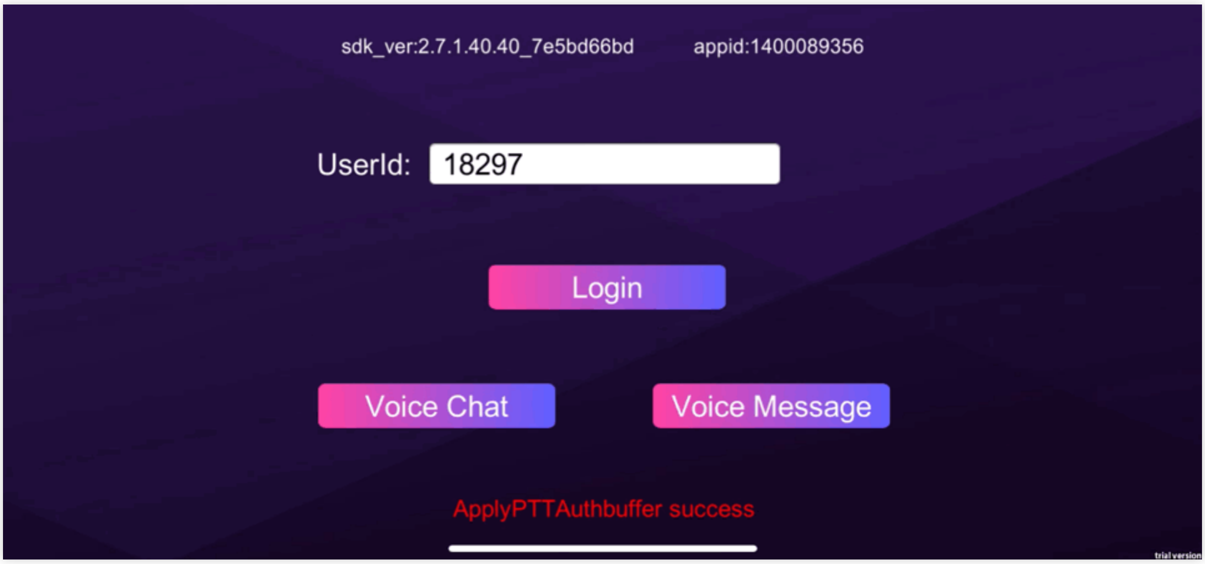
```
-(int)Uninit;
```

Sample code

```
[[ITMGContext GetInstance] Uninit];
```

Voice Messaging and Speech-to-Text

Voice messaging refers to recording and sending a voice message. At the same time, the voice message can be converted to text and translated, as shown below:

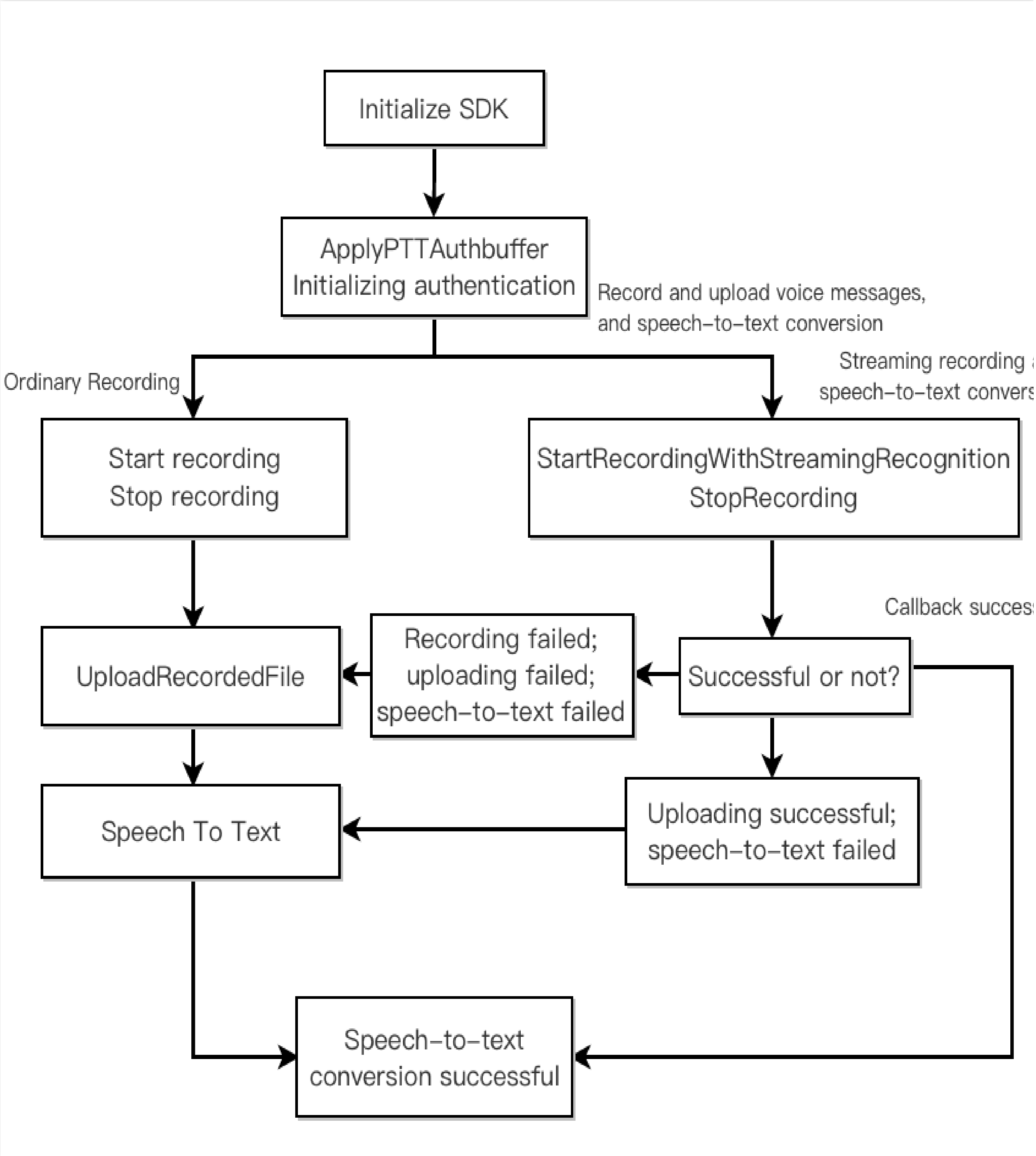


Note:

It is recommended to use the streaming speech-to-text service.

You do not need to enter a voice chat room when using the voice message service.

Voice messaging and speech-to-text conversion flowchart



Integrating Voice Messaging and Speech-to-Text Service

Voice messaging and speech-to-text APIs

API	Description

ApplyPTTAuthbuffer	Initializes authentication
SetMaxMessageLength	Specifies the maximum length of voice message
StartRecording	Starts recording
StartRecordingWithStreamingRecognition	Starts streaming recording
PauseRecording	Pauses recording
ResumeRecording	Resumes recording
StopRecording	Stops recording
CancelRecording	Cancels recording
GetMicLevel	Gets the real-time mic volume
SetMicVolume	Sets the recording volume
GetMicVolume	Gets the recording volume
GetSpeakerLevel	Gets the real-time speaker volume
SetSpeakerVolume	Sets the playback volume
GetSpeakerVolume	Gets the playback volume
UploadRecordedFile	Uploads the audio file
DownloadRecordedFile	Downloads the audio file
PlayRecordedFile	Plays back audio
StopPlayFile	Stops playing back audio
GetFileSize	Gets the audio file size
GetVoiceFileDuration	Gets the audio file duration
SpeechToText	Converts speech to text

Maximum recording duration

The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum recording duration to 10 seconds, please call the `SetMaxMessageLength` API to set it after initialization.

Initializing the SDK

Before the initialization, the SDK is in the uninitialized status, and you need to initialize it through the `Init` API before you can use the voice chat and voice message services.
If you have any questions when using the service, see Speech-to-text Conversion.

Authentication information

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.
To get authentication for voice message and speech-to-text, the room ID parameter must be set to `null` .

Function prototype

```
@interface QAVAuthBuffer : NSObject
+ (NSData*) GenAuthBuffer:(unsigned int)appId roomId:(NSString*)roomId openID:
(NSString*)openID key:(NSString*)key;
+ @end
```

Parameter	Type	Description
appId	int	`AppId` from the Tencent Cloud console.
roomId	NSString	Enter `null` .
openID	NSString	User ID, which is the same as `openID` during initialization.
key	NSString	Permission key from the Tencent Cloud console.

Sample code

```
#import "GMESDK/QAVAuthBuffer.h"
NSData* authBuffer = [QAVAuthBuffer GenAuthBuffer:SDKAPPID3RD.intValue
roomId:_roomId openID:_openId key:AUTHKEY];
```

Initializing authentication

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

Function prototype

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

Parameter	Type	Description
authBuffer	NSData*	Authentication

Sample code

```
[[[ITMGContext GetInstance]GetPTT]ApplyPTTAuthbuffer:(NSData *)authBuffer];
```

Streaming Speech Recognition

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

```
-(int)StartRecordingWithStreamingRecognition:(NSString *)filePath;
-(int)StartRecordingWithStreamingRecognition:(NSString *)filePath language:
(NSString*)speechLanguage translatelanguage:(NSString*)translateLanguage;
```

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

Sample code

```
recordfilePath = [docDir stringByAppendingFormat:@"/test_%d.ptt",index++];
[[[ITMGContext GetInstance] GetPTT]
StartRecordingWithStreamingRecognition:recordfilePath language:@"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` . Event messages are divided into:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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. `fileid` is fixed at `http://gme-v2-`

Note:

The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

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 conversion 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.

Sample code

```
- (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)
            {
                donwLoadUrlPath = data[@"file_id"];

                recordfilePath = [data objectForKey:@"file_path"];
                _localFileField.text = recordfilePath;

                _donwloadUrlField.text = [data objectForKey:@"file_id"] ;

                UITextField *_audiotoTextField =
(UITextField*)objc_getAssociatedObject(self, [PTT_AUDIO_TO_TEXT UTF8String]);
                _audiotoTextField.text = [data objectForKey:@"text"] ;
            }

        }
            break;
    }
}
```

Voice Message Recording

The recording process is as follows: start recording -> stop recording -> return recording callback -> start the next recording.

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

Function prototype

```
-(QAVResult)SetMaxMessageLength:(int)msTime
```

Parameter	Type	Description

msTime	int	Audio duration in ms. Value range: 1000 < msTime <= 58000

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SetMaxMessageLength:(int)msTime];
```

Starting recording

This API is used to start recording. The recording file must be uploaded first before you can perform operations such as speech-to-text conversion. **To stop recording, call `StopRecording`** .

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of stored audio file

Sample code

```
recordfilePath =[docDir stringByAppendingFormat:@"/test_%d.ptt",index++];
[[[ITMGContext GetInstance]GetPTT]StartRecording:recordfilePath];
```

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

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

Sample code

```
[[[ITMGContext GetInstance]GetPTT]StopRecording];
```

Callback for recording start

A callback will be executed through a delegate function to pass a message when recording is completed.

To stop recording, call `StopRecording` . The callback for recording start will be returned after the recording is stopped.

The callback function `OnEvent` will be called after recording is started. The event message `ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.

The passed parameter includes `result` and `file_path` .

Error codes

Error Code Value	Cause	Suggested Solution
4097	Parameter is empty.	Check whether the API parameters in the code are correct.
4098	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Ensure that the SDK recording feature is used at the right time.
4100	Audio data is not captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	Mic permission is required for using the SDK. To add the permission, see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE:
```

```
        {
        //Recording callback
        }
            break;
    }
}
```

Pausing recording

This API is used to pause recording. If you want to resume recording, please call the `ResumeRecording` API.

Function prototype

```
-(int)PauseRecording;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]PauseRecording;
```

Resuming recording

This API is used to resume recording.

Function prototype

```
-(int)ResumeRecording;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]ResumeRecording;
```

Canceling recording

This API is used to cancel recording. There is no callback after cancellation.

Function prototype

```
-(QAVResult)CancelRecording;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]CancelRecording];
```

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)GetMicLevel;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetMicLevel];
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)SetMicVolume:(int) volume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SetMicVolume:100];
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(int)GetMicVolume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetMicVolume];
```

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)GetSpeakerLevel;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetSpeakerLevel];
```

Setting the playback volume of voice message

This API is used to set the playback volume of voice messaging. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(QAVResult)SetSpeakerVolume:(int)volume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SetSpeakerVolume:100];
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice messaging. An int-type value will be returned. Value range: 0-200.

Note:

This API is different from the voice chat API and is in `ITMGPTT` .

Function prototype

```
-(int)GetSpeakerVolume;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetSpeakerVolume];
```

Voice Message Playback

Playing back audio

This API is used to play back audio.

Function prototype

```
-(int)PlayRecordedFile:(NSString*)filePath;
-(int)PlayRecordedFile:(NSString*)filePath VoiceType:(ITMG_VOICE_TYPE) type;
```

Parameter	Type	Description
downloadFilePath	NSString	Local audio file path
type	ITMG_VOICE_TYPE	Voice changer type. For more information, see Voice Changing Effects.

Error codes

Error Code Value	Cause	Suggested Solution

20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]PlayRecordedFile:path];
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameter includes `result` and `file_path`.

Error codes

Error Code Value	Cause	Suggested Solution
20481	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	Parameter is empty.	Check whether the API parameters in the code are correct.
20484	Internal error.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE:
        {
        // Callback for audio playback
        }
            break;
```

```
    }
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

Function prototype

```
-(int)StopPlayFile;
```

Sample code

```
[[[ITMGContext GetInstance]GetPTT]StopPlayFile];
```

Getting audio file size

This API is used to get the size of an audio file.

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of audio file, which is a local path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetFileSize:path];
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of audio file, which is a local path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]GetVoiceFileDuration:path];
```

Voice Message Upload and Download

Uploading an audio file

This API is used to upload an audio file.

Function prototype

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

Parameter	Type	Description
filePath	NSString	Path of uploaded audio file, which is a local path.

Sample code

```
[[[ITMGContext GetInstance]GetPTT]UploadRecordedFile:path];
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Error codes

Error Code Value	Cause	Suggested Solution
8193	An error occurred while accessing the	Ensure the existence of the file and the validity of

	file during upload.	the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
8195	A network error occurred.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8200	No `appinfo` is set.	Check whether the `apply` API is called or whether the input parameters are empty.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE:
        {
            if (data != NULL &&[[data objectForKey:@"result"] intValue]== 0)
            {
                _donwloadUrlField.text = [data objectForKey:@"file_id"] ;
                donwLoadUrlPath = [data objectForKey:@"file_id"] ;
            }
        }
            break;
    }
}
```

Downloading the audio file

This API is used to download an audio file.

Function prototype

```
-(void)DownloadRecordedFile:(NSString*)fileId downloadFilePath:
(NSString*)downloadFilePath
```

Parameter	Type	Description
fileID	NSString	File URL path
downloadFilePath	NSString	Local path of saved file

Sample code

```
[[[ITMGContext GetInstance]GetPTT]DownloadRecordedFile:fileIdpath
downloadFilePath:path];
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Error codes

Error Code Value	Cause	Suggested Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
12291	Network storage system exception	The server failed to get the audio file. Check whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	Server file system error.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed during the process of getting the download parameters.	Check whether the device can access the internet.

12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE:
        {
            if (data != NULL &&[[data objectForKey:@"result"] intValue]== 0)
            {
                _audiofileToPlayField.text = [data objectForKey:@"file_path"] ;
                donwLoadLocalPath = [data objectForKey:@"file_path"];
            }
            else
            {
                donwLoadLocalPath = NULL;
            }
        }
            break;
    }
}
```

Speech-to-Text Service

Converting audio file to text

This API is used to convert a specified audio file to text.

Function prototype

```
-(void)SpeechToText:(NSString*)fileID;
```

Parameter	Type	Description
fileID	NSString	URL of audio file

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SpeechToText:fileID];
```

Translating audio file into text in specified language

This API can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation.

Function prototype

```
-(void)SpeechToText:(NSString*)fileID (NSString*)speechLanguage
(NSString*)translateLanguage;
```

Parameter	Type	Description
fileID	NSString*	URL of audio file, which will be retained on the server for 90 days
speechLanguage	NSString*	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translateLanguage	NSString*	The language into which the audio file will be translated. For parameters, see Language Parameter Reference List. This parameter is currently unavailable. Enter the same value as that of `speechLanguage` .

Sample code

```
[[[ITMGContext GetInstance]GetPTT]SpeechToText:fileID speechLanguage:"cmn-Hans-
CN" translateLanguage:"cmn-Hans-CN"];
```

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the

`OnEvent` function.

The passed parameters include `result` , `file_path` and `text` (recognized text).

Error codes

Error Code Value	Cause	Suggested Solution
32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	Network failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	Incorrect speech-to-text conversion parameter.	Check whether the API parameter `fileid` in the code is empty.
32785	Speech-to-text translation returned an error.	Error with the backend of voice message and speech-to-text feature. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.

Sample code

```
-(void)OnEvent:(ITMG_MAIN_EVENT_TYPE)eventType data:(NSDictionary *)data{
    NSLog(@"OnEvent:%lu,data:%@",(unsigned long)eventType,data);
    switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE:
        {
            if (data != NULL &&[[data objectForKey:@"result"] intValue]== 0)
            {
                UITextField *_audiotoTextField =
(UITextField*)objc_getAssociatedObject(self, [PTT_AUDIO_TO_TEXT UTF8String]);
                _audiotoTextField.text = [data objectForKey:@"text"] ;
            }
        }
```

```
                break;
        }
}
```

Advanced APIs

Getting version number

This API is used to get the SDK version number for analysis.

Function prototype

```
-(NSString*)GetSDKVersion;
```

Sample code

```
[[ITMGContext GetInstance] GetSDKVersion];
```

Checking mic permission

This API is used to return the mic permission status.

Function prototype

```
-(ITMG_RECORD_PERMISSION)CheckMicPermission;
```

Parameter description

Parameter	Value	Description
ITMG_PERMISSION_GRANTED	0	Mic permission is granted.
ITMG_PERMISSION_Denied	1	Mic is disabled.
ITMG_PERMISSION_NotDetermined	2	No authorization box has been popped up to request the permission.
ITMG_PERMISSION_ERROR	3	An error occurred while calling the API.

Sample code

```
[[ITMGContext GetInstance] CheckMicPermission];
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

Function prototype

```
-(void)SetLogLevel:(ITMG_LOG_LEVEL)levelWrite (ITMG_LOG_LEVEL)levelPrint;
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written. `TMG_LOG_LEVEL_NONE` indicates not to write. Default value: TMG_LOG_LEVEL_INFO
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed. `TMG_LOG_LEVEL_NONE` indicates not to print. Default value: TMG_LOG_LEVEL_ERROR

ITMG_LOG_LEVEL

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
[[ITMGContext GetInstance] SetLogLevel:TMG_LOG_LEVEL_INFO TMG_LOG_LEVEL_INFO];
```

Setting the log printing path

. This API is used to set the log printing path, which is /Users/username/Library/Containers/xxx.xxx.xxx/Data/Documents by default.. Call before Init.

Function prototype

```
-(void)SetLogPath:(NSString*)logDir;
```

Parameter	Type	Description
logDir	NSString	Path

Sample code

```
[[ITMGContext GetInstance] SetLogPath:Path];
```

Callback Messages

Message list

Message	Description
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Indicates that PTT recording is completed.
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Indicates that PTT upload is completed.
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Indicates that PTT download is completed.
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Indicates that PTT playback is completed.
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Indicates that speech-to-text conversion is completed.

Data list

Message	Data	Sample
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	result;	{"file_path":

	file_path	
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	result; file_path;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	result; file_path;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	result; file_path	{"file_path":
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	result; text;file_id	{"file_id":"",'
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	result; file_path; text;file_id	{"file_id":"",'

H5 SDK
Project Configuration

Last updated：2024-01-18 15:13:51

This document describes how to get started with the GME APIs for HTML5.

Features Supported by the SDK for HTML5

Feature	Supported by HTML5
Basic voice chat features	Yes
Sound effect and accompaniment	Partially supported
3D voice	No
Range voice	No
Room management	No
Audio forwarding routing	No
Custom message channel	No
Voice messaging	No
Speech-to-text	No

Platforms Supported by the SDK for HTML5

OS/Platform	Browser/WebView	Version Requirement	Remarks
iOS	Safari	11.1.2	Safari on a later version
	Chrome	-	iOS currently doesn't support Chrome.
	Weixin browser	-	iOS currently doesn't support the Weixin browser.

Android	TBS (the default WebView of Weixin and Mobile QQ)	43600	The default built-in browser kernel of Weixin and Mobile QQ is TBS
	Chrome	60+	H.264 support is required
macOS	Chrome	47+	-
	Safari	11+	-
Windows (PC)	Chrome	52+	-
	QQ Browser	10.2	-

Preparations

The SDK can be obtained in the following steps:

1. Go to the SDK Download Guide page.
2. Locate the SDK resources for HTML5 on the page.
3. Click **Download**.

Frontend Project Configuration Steps

1. Open ports

If you have configured a firewall across your network, be sure to open the following ports:

Protocol	Port
TCP	8687
UDP	8000, 8800, 443

Import the SDK by using CDN.

2. Import frontend library files

Import `WebRTCService.min.js` into the project as demonstrated in the following code:

```
<head>
    <script src="../dist/WebRTCService.min.js"></script>
</head>
```

3. Add the audio tag

```
<div id="gme-audio-wrap"></div>
```

Server-Side Deployment Steps

The use of the GME SDK requires authentication which involves keys and is not suitable for implementation on the client. You are recommended to deploy it separately.
If only client implementation is needed for the time being, please refer to the provided demo project.

1. Download the program

Download the sample `authBuffer` program, which can sign the authentication information for a specified `SDKAppID`.

2. Configure the server-side authentication project

Go to the `signdemo` directory and modify the `config.js` file: open the `config.js` file, delete the default configuration, and call the `appidMap` function in the place where the code is deleted (the parameters are the `SDKAppid` applied for on the Tencent Cloud backend and the corresponding authentication key).

```
const AuthBufferConfig = function () {
    this.appidMap = {};
    this.appidMap["1400089356"] = "1cfbfd2a1a03a53e";
};
// Replace `1400089356` with the `sdkAppid` applied for on the Tencent Cloud
backend and replace `1cfbfd2a1a03a53e` with the authentication key
corresponding to the `sdkAppid`
```

Note:

The `AuthKey` must correspond to your `SDKAppid`.

3. Deploy the server-side authentication project

Go to the directory where the sample `authBuffer` program resides and run the following statement to install the dependencies:

```
npm i
```

Then, execute the `node index.js` script to run the signature service.

Note:

As the async syntax is used, make sure that your node is v8 or later. Run `node -v` on the command line to view the version.

4. Test the deployment result

You can run the following command on the command line for test (make sure that your system has a `curl` command):

```
// Generate a `userSig`:
curl "http://127.0.0.1:10005/" --data
"sdkappid=1400089356&roomid=1234123&openid=1234567"
```

After the signing program is executed, the authentication information will be returned as shown below:

```
{"userSig":"AqhHE7QHLFYPfV/zfyrdRYHfuUn6eOA8g/J6GMjVy//Shr5ByJPTi8hzR2KyXMvn","
errorCode":0}
```


API Documentation

Last updated：2024-01-18 15:13:51

Note:

Only some GME features are supported by the SDK for HTML5. Please refer to this document for the supported APIs and evaluate whether the SDK for HTML5 is appropriate for your business scenario.

API	Description
Init	Initializes API
SetTMGDelegate	Sets delegation
EnterRoom	Enters audio room
EnableMic	Turns on/off the capturing device
EnableSpeaker	Turns on/off the playback device
SetMicVolume	Sets mic volume
ExitRoom	Exits audio room

Note:

After a GME API is called successfully, `QAVError.OK` will be returned with the value being 0.

Authentication is required for room entry in GME. For more information, see the authentication section in relevant documentation.

Operation on devices should be performed after successful room entry.

Starting from Chrome 74, `navigator.mediaDevices` can only be used in an HTTPS environment; therefore, please use HTTPS.

Integrating JQ

You need to integrate JQ to use the demo.

```
<!--Step 2: Add the audio container-->
<!--Container, which is used to carry audio tags and cannot be omitted.-->
<div id="gme-audio-wrap"></div>
```

Initialization APIs

Before initialization, the SDK is in the uninitialized state. A room can be entered only after the initialization authentication is performed and the SDK is initialized.

Initializing the SDK

For more information on how to get parameters, see Access Guide.

This API requires the `SDKAppID` from the Tencent Cloud console and the `openId` as parameters. The `openId` uniquely identifies a user with the rules stipulated by the application developer and must be unique in the application (currently, only INT64 is supported).

Note:

The SDK must be initialized before a user can enter a room.

Function prototype

```
WebGMEAPI.fn.Init = function (document, SdkAppId, openId) {...}
```

Parameter	Description
document	HTML DOM Document object
SdkAppId	`SdkAppId` from the Tencent Cloud console
openId	Developer-defined user account with a value greater than 10,000, which is used to identify the user.

Sample code

```
const cSdkAppId = () => document.getElementById("input-SdkAppId").value;
const cOpenID = () => document.getElementById("input-OpenID").value;
gmeAPI.Init(document, cSdkAppId(), cOpenID());
```

Setting callbacks

The API class uses the `Delegate` method to send callback notifications to the application. Register the callback function to the SDK to receive callback messages. The callback function should be registered to the SDK before room entry.

Function prototype

```
WebGMEAPI.fn.SetTMGDelegate = function (delegate){...}
```

Parameter	Description
onEvent	SDK callback event

Sample code

```
gmeAPI.SetTMGDelegate(onEvent);
```

Voice Chat APIs

You should initialize and call the SDK to enter a room before voice chat can start.

Entering a room

When a user enters a room with the generated authentication information, the `ITMG_MAIN_EVENT_TYPE_ENTER_ROOM` message will be received as a callback. Mic and speaker are not turned on by default after room entry.

Function prototype

```
WebGMEAPI.fn.EnterRoom = function (roomId, roomType, authBuffer) {...}
```

Parameter	Description
roomId	Room ID, which can contain up to 127 characters
roomType	Room audio type
authBuffer	Authentication key. For more information on how to get it, see Project Configuration.

Sample code

```
function bindButtonEvents() {
        $("#start_btn").click(function () {
            console.log('start!');
            // Step 1: Get the `AuthBuffer`
            var FetchSigCgi = 'http://134.175.146.244:10005/';
            $.ajax({
                type: "POST",
                url: FetchSigCgi,
                dataType: 'json',
                data: {
```

```
                    sdkappid: cSdkAppId(),
                    roomid: cRoomNum(),
                    openid: cOpenID(),
                },
                success: function (json) {
                    // Step 2: `AuthBuffer` is obtained successfully
                    if (json && json.errorCode === 0) {
                        let userSig = json.userSig;
                        gmeAPI.Init(document, cSdkAppId(), cOpenID());
                        gmeAPI.SetTMGDelegate(onEvent);
                        gmeAPI.EnterRoom(cRoomNum(), 1, userSig);
                    } else {
                        console.error(json);
                    }
                },
                error: function (err) {
                    console.error(err);
                }
            });
        });
```

Event Callbacks

After the user enters the room, the message `ITMG_MAIN_EVENT_TYPE_ENTER_ROOM` will be sent and identified in the `OnEvent` function.

Sample code

```
 onEvent = function (eventType, result) {
         if (eventType === gmeAPI.event.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM)
        {
            // Entered room successfully
        }
        else if (eventType === gmeAPI.event.ITMG_MAIN_EVENT_TYPE_USER_UPDATE)
        {
            app._data.downStreamInfoList = result.PeerInfo;// Received peer
information. For more information, see the table below
            app._data.brSend = result.UploadBRSend;// Bitrate of the uploaded
audio data
            app._data.rtt = result.UploadRTT;// Upload RTT
        }
        else if (eventType === gmeAPI.event.ITMG_MAIN_EVENT_TYPE_EXIT_ROOM)
        {
            // Exited room successfully
        }
```

```
        else if (eventType ===
gmeAPI.event.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT)
        {
            // Room disconnected
        }
    };
```

The received peer information is as follows (downStreamInfoList):

Parameter	Description
brRecv	The received bitrate
delay	Receipt delay
jitterBufferMs	Delay caused by jitter
jitterReceived	The received Jitter

Exiting a room

This API is called to exit the current room. It is an async API. There will be a callback after room exit. The returned value of `AV_OK` indicates a successful async delivery.

Function prototype

```
WebGMEAPI.fn.ExitRoom = function (){...}
```

Sample code

```
gmeAPI.ExitRoom();
```

Turning on/off the mic

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

Function prototype

```
WebGMEAPI.fn.EnableMic = function (bEnable) {...}
```

Parameter	Description
isEnabled	To turn on the mic, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
gmeAPI.EnableMic(false);
```

Setting the mic volume

This API is used to set the mic volume. The corresponding parameter is `volume` . 0 indicates that the audio is mute, while 100 indicates that the volume remains unchanged. The default value is 100.

Function prototype

```
WebGMEAPI.fn.SetMicVolume = function (volume){...}
```

Parameter	Description
volume	Sets the volume. Value range: 0-100.

Sample code

```
gmeAPI.SetMicVolume(100);
```

Turning on/off the speaker

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

Function prototype

```
WebGMEAPI.fn.EnableSpeaker = function (bEnable){...}
```

Parameter	Description
isEnabled	To turn off the speaker, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

```
gmeAPI.EnableSpeaker(true);
```

Electron SDK

Integrating SDK

Last updated：2024-01-18 15:15:48

This document describes how to configure an Electron project for the GME APIs for Electron.

Supported Platforms

Windows

Importing the SDK

Step 1: Install Node.js

1. Download the latest version of the Node.js installation package `Windows Installer (.msi) 64-bit` for Windows.
2. Open `Node.js command prompt` in the application list and open a terminal window.

Step 2. Install Electron

Run the following command in the terminal to install Electron. V4.0.0 or later is recommended.

```
$ npm install electron -g
```

Step 3. Install the GME SDK for Electron

1. Use the following npm command in your Electron project to install the GME SDK:

```
$ npm install gme-electron-sdk@latest --save
```

2. In the project script, import and use the module:

```
const { GmeContext } = require('gme-electron-sdk');
// import gmesdk from 'gme-electron-sdk';
gmeContext = new GmeContext();
// Get the SDK version number
gmeContext.GetSDKVersion();
```

Step 4. Create an executable program

Install the packaging tool. We recommend you use Electron Forge. You can run the following command:

1. Add Electron Forge to your application's development dependencies and run the `import` command to set the scaffold of Forge:

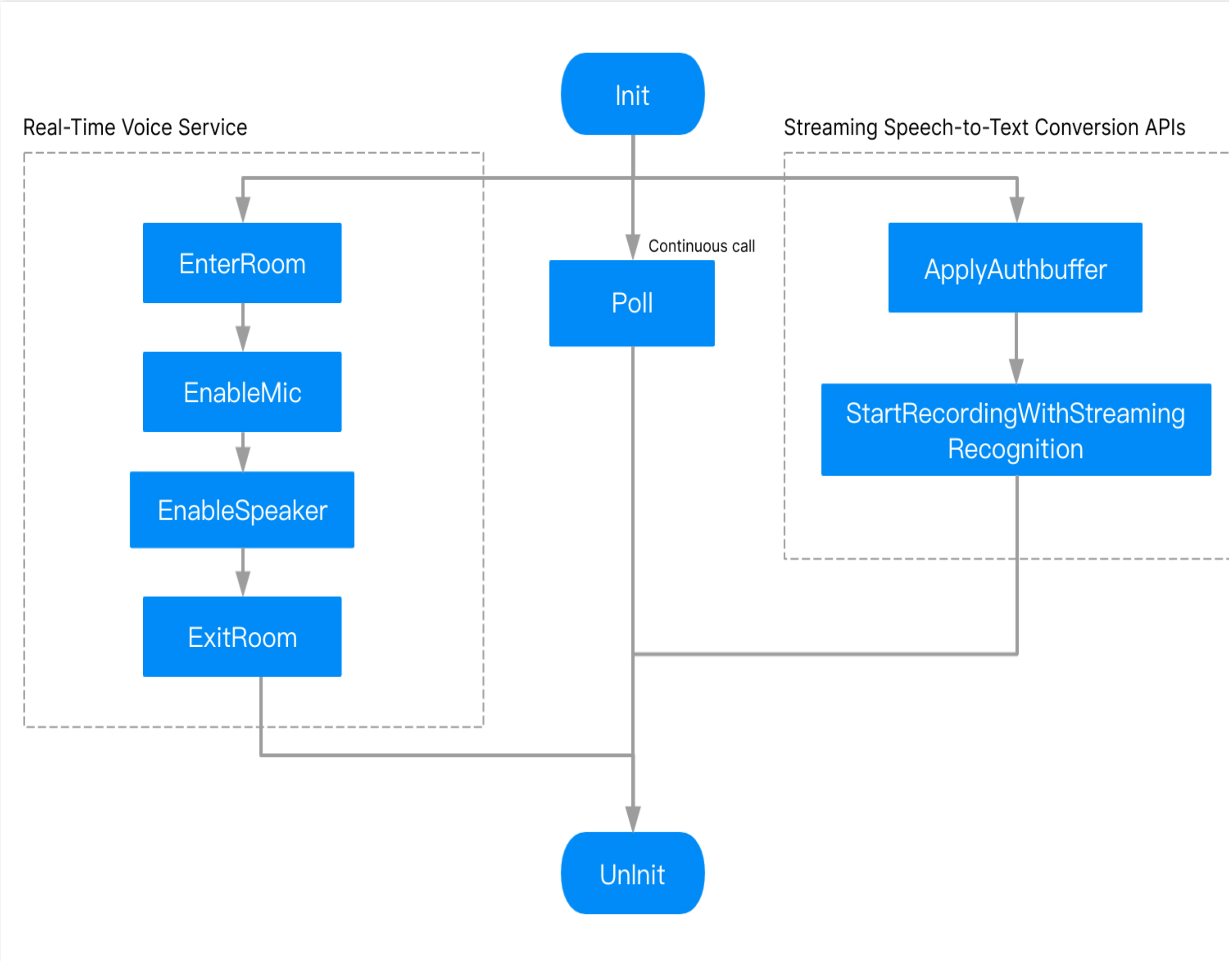
```
npm install --save-dev @electron-forge/cli
npx electron-forge import
```

2. Run the `make` command of Forge to create a distributable application:

```
npm run make
```

Voice Chat

Last updated：2024-01-18 15:15:48



This document describes how to integrate with and debug GME client APIs for the voice chat feature for Electron.

Key Considerations for Using GME

GME provides the voice chat, voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

Notes

You have created a GME application and obtained the SDK `AppID` and key. For more information, see Activating Services.

You have activated **GME voice chat, voice messaging, and speech-to-text services**. For more information, see Activating Services.

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `GmeError.AV_OK` will be returned with the value being `0` .

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error codes, see Error Codes.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:

1. Initializing GME
2. Calling `Poll` periodically to trigger callbacks
3. Entering a voice chat room
4. Turning on the mic
5. Turning on the speaker
6. Exiting the voice chat room
7. Uninitializing GME

Core APIs

API	Description
Init	Initializes GME.
Poll	Triggers an event callback.
Pause	Pauses the system.
Resume	Resumes the system.
Uninit	Uninitializes GME.

Importing the GME module

```
const { GmeContext } = require('gme-electron-sdk');
```

Getting an instance

To use the voice chat feature, get the `GmeSDK` object first.

```
context = new GmeContext();
```

Initializing the SDK

You need to initialize the SDK through the `Init` API before you can use the real-time voice chat, 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 GmeSDK
Init(appid: string, openid: string): number;
```

Parameter	Type	Description
sdkAppId	string	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	string	`openID` can only be in Int64 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.

Returned values

Returned Value	Description
GmeError.AV_OK= 0	The SDK was initialized successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Check whether the SDK file is complete. We recommend that you delete it and then import it again.

Note on 7015 error code

The error code 7015 is identified based on the MD5 value. If this error is reported during integration, check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party enhancement, the Unity packaging mechanism, and other factors, the MD5 value of the library file will be affected, resulting in misjudgment. Therefore, **ignore this error in the logic for official releases**, and avoid displaying it on the UI.

Sample code

```
string SDKAPPID3RD = "14000xxxxx";
string openId="10001";
number ret = context.Init(SDKAPPID3RD, openId);
// Determine whether the initialization is successful by the returned value
if (ret != GmeError.AV_OK)
{
        console.log("Failed to initialize the SDK:");
        return;
}
```

Setting callbacks

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.

```
SetTMGDelegate(cb: ITMGDelegate);
// When initializing the SDK
context =  GmeSDK.GetInstance();
context.setTMGDelegate(function(eventId, msg){
  if (type == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM)
  {
            // Processing callbacks
  }
});
```

Triggering event callback

You need to periodically call the `Poll` API to trigger event callbacks. 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.

Call the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
Poll():number;
```

Sample code

```
setInterval(function () {
      context.Poll();
    }, 50);
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. For example, when the application switches to the background (OnApplicationPause, isPause=True), and you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
Pause() :number
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
Resume() :number
```

Uninitializing SDK

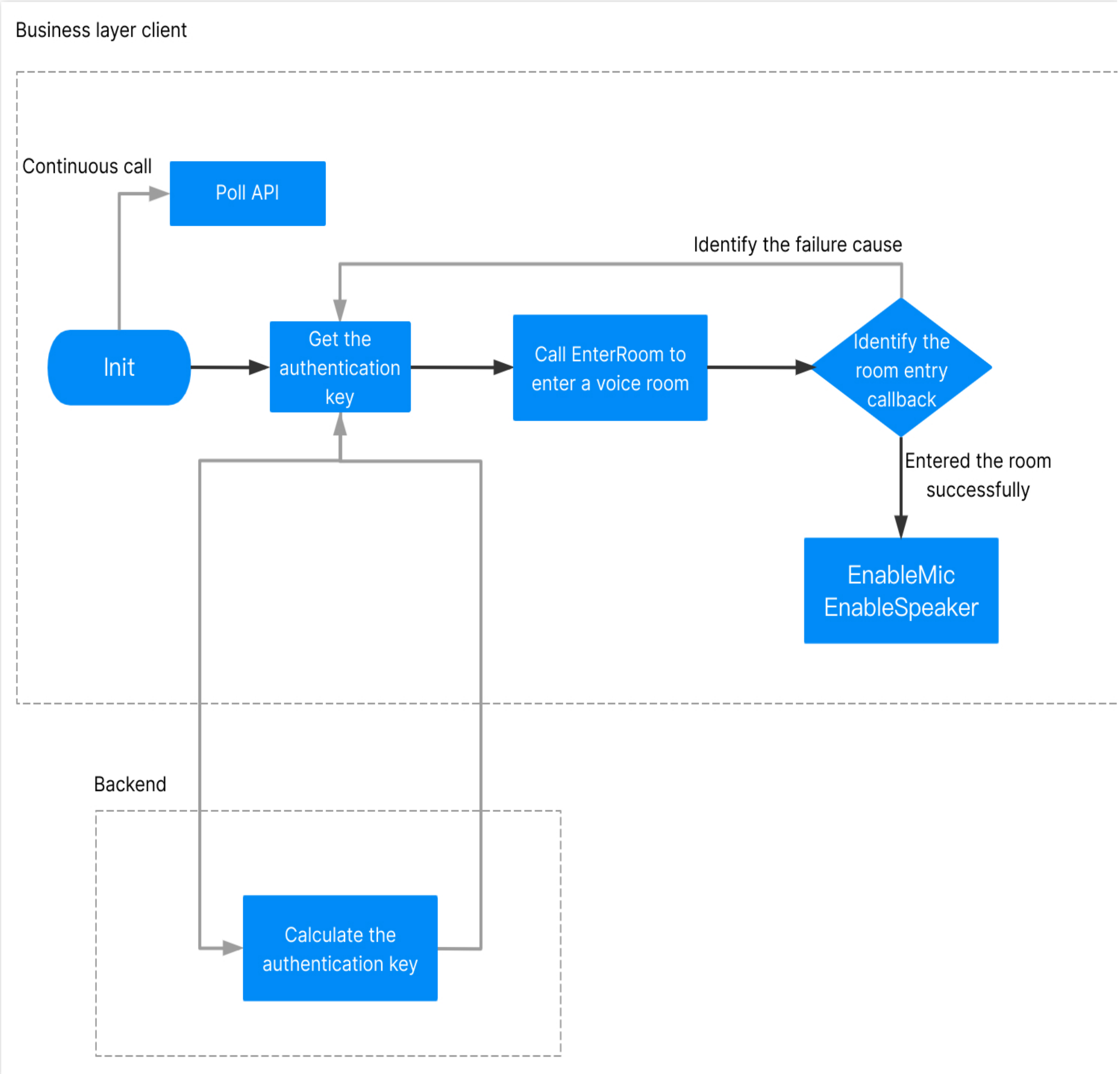
This API is used to uninitialize the SDK to make it uninitialized. **If the game account is bound to** `openid` , switching game account requires uninitializing GME and then using the new `openid` to initialize again.

API prototype

```
Uninit() : number;
```

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.
If you have any questions when using the service, see Sound and Audio.



API	Description
GenAuthBuffer	Calculates the local authentication key.
EnterRoom	Enters a room.
ExitRoom	Exits a room.
IsRoomEntered	Determines whether room entry is successful.

Local authentication key calculation

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

API prototype

```
GenAuthBuffer(appId: string,roomId: string, openId:string, appKey: number)
:string;
```

Parameter	Type	Description
appId	string	`AppID` from the Tencent Cloud console
roomId	string	Room ID, which can contain up to 127 characters.
openId	string	User ID, which is the same as `openID` during initialization.
key	number	Permission key from the Tencent Cloud console.

Sample code

```
 let userSig = context.GenAuthBuffer(this.appid, this.roomId, this.userId,
this.authKey)
 context.EnterRoom(this.roomId, this.roomType, userSig);
```

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.

Note：

If the room entry callback result is `0`, the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.

The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will become smooth sound quality. Only when a member in the room calls the `ChangeRoomType` API, the audio type of the room will be changed.

API prototype

```
EnterRoom(roomid: string, roomType: number, appKey: string) :number;
```

Parameter	Type	Description
roomId	string	Room ID, which can contain up to 127 characters.
roomType	ITMGRoomType	Room type. We recommend that you select `ITMG_ROOM_TYPE_FLUENCY` for games. For more information on room audio types, see Sound Quality.
appKey	string	Authentication key

Sample code

```
context.EnterRoom(roomID, ITMG_ROOM_TYPE_STANDARD, retAuthBuff);
```

Callback for room entry

After the user enters the room, the `ITMG_MAIN_EVENT_TYPE_ENTER_ROOM` event type will be called back to notify the room entry result, 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

Will the billing continue if the client is disconnected from the server when using the voice chat?

Sample code

```
// Listen on an event:
 gmeContext.setTMGDelegate(function(eventId, msg){
  switch (eventId) {
      case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
      {
      }
  }
});
```

Data details

Message	Data	Example
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result; error_info	{"error_info":"","result":0}
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnection callback notification `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code	Cause and Suggested Solution
7006	Authentication failure causes: `AppID` doesn't exist or is incorrect. An error occurred while authenticating `authbuff` . Authentication expired. `OpenId` is invalid.
7007	The user was already in another room.
1001	The user was already in the process of entering a room but repeated this operation. We recommend that you not call the room entry API until the room entry callback is returned.
1003	The user was already in the room and called the room entry 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.

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API; instead, you can directly call the `EnterRoom` API.

API prototype

```
ExitRoom(): number;
```

Sample code

```
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:

Sample code

```
gmeContext.setTMGDelegate(function(eventId, msg){
  switch (eventId) {
      case ITMG_MAIN_EVENT_TYPE_EXIT_ROOM:
      {
             // Process
        break;
      }
  }
});
```

Determining whether user has entered room

This API is used to determine whether the user has entered a room. A value in boolean type will be returned. Do not call this API during room entry.

API prototype

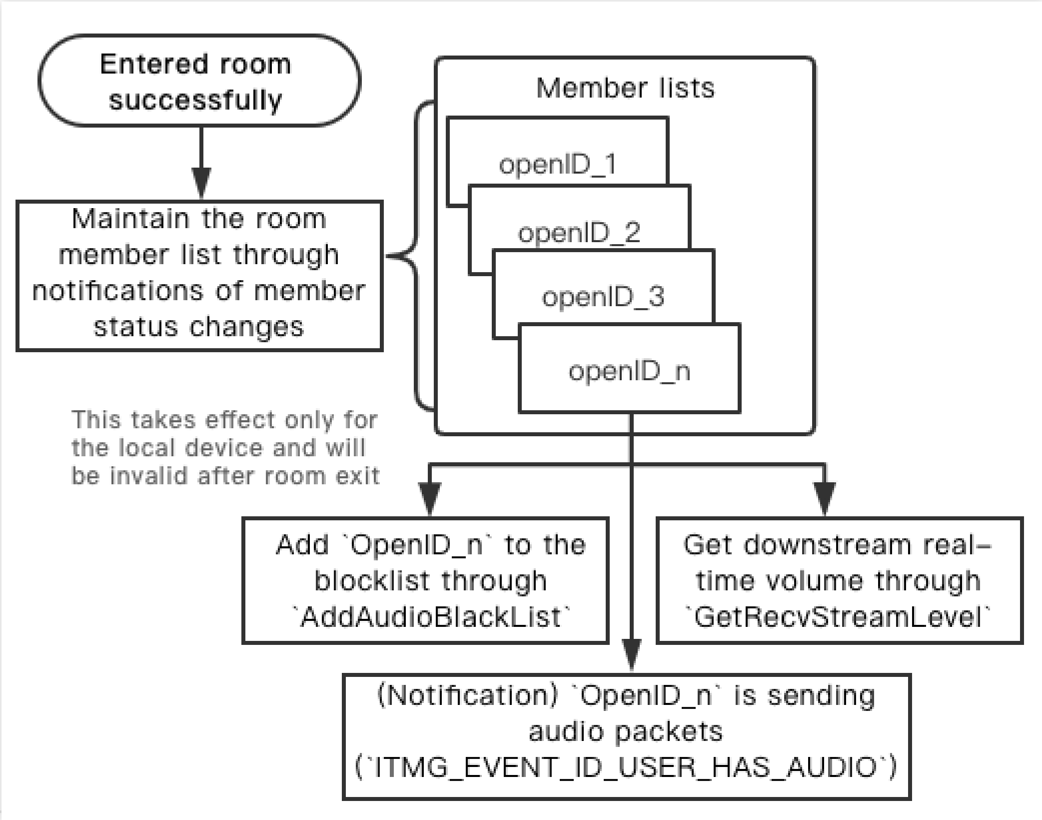
```
IsRoomEntered() :boolean
```

Sample code

```
context.IsRoomEntered();
```

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed.
AddAudioBlackList	Mutes a member in the room.
RemoveAudioBlackList	Unmutes a member.
IsOpenIdInAudioBlackList	Queries whether the user of the specified `openid` is muted.

Notification events of the member room entry and speaking status

This event is used to get the user speaking in the room and display the user on the UI, and to send a notification when someone enters or exits the room.

A notification for this event will be sent only when the status changes. To get the member status in real time, cache the notification when it is received at the business layer. The event message `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` containing `event_id`, `count`, and `openIdList` will be returned, which will be identified in the `OnEvent` notification.

Notifications of the `EVENT_ID_ENDPOINT_NO_AUDIO` audio event will be sent only when the threshold is exceeded; that is, other members in the room can receive the notification that the local user stops speaking only after the local client captures no voice for two seconds.

The audio event returns only the member speaking status but not the specific volume level. If you need the specific volume levels of members in the room, you can use the `GetVolumeById` API.

event_id	Description	Maintenance
EVENT_ID_ENDPOINT_ENTER	Return the `openid` of the member entering the room.	Member list
EVENT_ID_ENDPOINT_EXIT	Return the `openid` of the member exiting the room.	Member list
EVENT_ID_ENDPOINT_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
EVENT_ID_ENDPOINT_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
context.setTMGDelegate(function(eventId, msg){
  if (type == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM)
  {
               // Process
            switch (eventID)
             {
             case EVENT_ID_ENDPOINT_ENTER:
                  // A member enters the room
                  break;
             case EVENT_ID_ENDPOINT_EXIT:
                  // A member exits the room
                break;
            case EVENT_ID_ENDPOINT_HAS_AUDIO:
                // A member sends audio packets
                break;
            case EVENT_ID_ENDPOINT_NO_AUDIO:
                // A member stops sending audio packets
                break;

            default:
                break;
             }
        break;
  }
});
```

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B.

If B blocks neither A nor C, B can hear both of them.

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
AddAudioBlackList(openId: string) :number
```

Parameter	Type	Description
openId	string	`openid` of the user to be blocked

Sample code

```
context.AddAudioBlackList(openId);
```

Unmuting

This API is used to remove an ID from the audio data blacklist. A returned value of 0 indicates the call is successful.

API prototype

```
RemoveAudioBlackList(openId: string) :number
```

Parameter	Type	Description
openId	string	ID to be unblocked

Sample code

```
context.RemoveAudioBlackList(openId);
```

Querying whether a user is muted

This API is used to query whether an ID is blocked. The returned value `true` indicates that the ID is blocked, while `false` indicates not.

API prototype

```
IsOpenIdInAudioBlackList(openId: string) :boolean
```

Parameter	Type	Description
openId	string	ID to be queried

Sample code

```
boolean isInBlackList = context.IsOpenIdInAudioBlackList(openId);
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.
When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend that you call the `EnableMic` or `EnableSpeaker` API.
To enable the user to press the mic button on the UI to speak and release it to stop speaking, we recommend that you call `EnableAudioCaptureDevice` once during room entry and call `EnableAudioSend` to enable the user to speak while pressing the button.

API	Description
EnableMic	Enables/Disables the mic.
GetMicState	Gets the mic status.
EnableAudioCaptureDevice	Enables/Disables the capturing device.
IsAudioCaptureDeviceEnabled	Gets the capturing device status.
EnableAudioSend	Enables/Disables audio upstreaming.
IsAudioSendEnabled	Gets the audio upstreaming status.
GetMicLevel	Gets the real-time mic volume level.
GetSendStreamLevel	Gets the real-time audio upstreaming volume level.
SetMicVolume	Sets the mic volume level.

GetMicVolume	Gets the mic volume level.

Enabling or disabling mic

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

EnableMic = EnableAudioCaptureDevice + EnableAudioSend

API prototype

```
EnableMic(bEnable: boolean) : number
```

Parameter	Type	Description
isEnabled	boolean	To enable the mic, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
// Turn on mic
context.EnableMic(true);
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
GetMicState() :number
```

Sample code

```
context.GetMicState();
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.

This API can only be called after room entry. The device will be disabled automatically after room exit.

Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
EnableAudioCaptureDevice(enable:boolean) :number
```


Parameter	Type	Description
enable	boolean	To enable the capturing device, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable capturing device
context.EnableAudioCaptureDevice(true);
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
IsAudioCaptureDeviceEnabled():boolean
```

Sample code

```
boolean IsAudioCaptureDevice = context.IsAudioCaptureDeviceEnabled();
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain muted. For more information on how to enable/disable the capturing device, see the `EnableAudioCaptureDevice` API.

API prototype

```
EnableAudioSend(bEnable: boolean) :number
```

Parameter	Type	Description
isEnabled	boolean	To enable audio upstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
context.EnableAudioSend(true);
```


Getting audio upstreaming status

This API is used to get the status of audio upstreaming.

API prototype

```
IsAudioSendEnabled():boolean
```

Sample code

```
boolean IsAudioSend = context.IsAudioSendEnabled();
```

Getting the real-time mic volume

This API is used to get the real-time mic volume level. A number-type value in the range of 0–100 will be returned. We recommend that you call this API once every 20 ms.

API prototype

```
GetMicLevel():number
```

Sample code

```
context.GetMicLevel();
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume level. A number-type value in the range of 0–100 will be returned.

API prototype

```
GetSendStreamLevel() :number
```

Sample code

```
context.GetSendStreamLevel();
```

Setting the mic software volume

This API is used to set the mic volume level. The corresponding parameter is `volume` , which is equivalent to attenuating or gaining the captured sound.

API prototype

```
SetMicVolume(volume:number) :number
```

Parameter	Type	Description
volume	number	Value range: 0-200. Default value: `100` . `0` indicates that the audio is muted, while `100` indicates that the volume level remains unchanged.

Sample code

```
number micVol = (value * 100);
context.SetMicVolume (micVol);
```

Getting the mic software volume

This API is used to get the mic volume level. A number-type value will be returned. `101` indicates that the `SetMicVolume` API has not been called.

API prototype

```
GetMicVolume()
```

Sample code

```
context.GetMicVolume();
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker.
GetSpeakerState	Gets the speaker status.
EnableAudioPlayDevice	Enables/Disables the playback device.
IsAudioPlayDeviceEnabled	Gets the playback device status.
EnableAudioRecv	Enables/Disables audio downstreaming.
IsAudioRecvEnabled	Gets the audio downstreaming status.

GetSpeakerLevel	Gets the real-time speaker volume level.
GetRecvStreamLevel	Gets the real-time downstreaming audio levels of other members in the room.
SetSpeakerVolume	Sets the speaker volume level.
GetSpeakerVolume	Gets the speaker volume level.

Enabling or disabling speaker

This API is used to enable/disable the speaker. **EnableSpeaker = EnableAudioPlayDevice + EnableAudioRecv**

API prototype

```
EnableSpeaker(bEnable: boolean) : number;
```

Parameter	Type	Description
bEnable	boolean	To disable the speaker, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

```
// Turn on the speaker
context.EnableSpeaker(true);
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

API prototype

```
GetSpeakerState() :number
```

Sample code

```
context.GetSpeakerState();
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

API prototype

```
EnableAudioPlayDevice(enable:boolean) :number
```

Parameter	Type	Description
enable	boolean	To disable the playback device, set this parameter to `false` ; otherwise, set it to `true` .

Sample code

```
context.EnableAudioPlayDevice(true);
```

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
IsAudioPlayDeviceEnabled() :boolean
```

Sample code

```
boolean enable = context.IsAudioPlayDeviceEnabled();
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain muted. For more information on how to enable/disable the playback device, see the `EnableAudioPlayDevice` API.

API prototype

```
EnableAudioRecv(bEnable: boolean) :number
```

Parameter	Type	Description
isEnabled	boolean	To enable audio downstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
context.EnableAudioRecv(true);
```

Getting audio downstreaming status

This API is used to get the status of audio downstreaming.

API prototype

```
IsAudioRecvEnabled():boolean
```

Sample code

```
boolean IsAudioRecv = context.IsAudioRecvEnabled();
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume level. A number-type value will be returned to indicate the volume level. We recommend that you call this API once every 20 ms.

API prototype

```
GetSpeakerLevel():number
```

Sample code

```
context.GetSpeakerLevel();
```

Getting the real-time downstreaming audio levels of other members in room

This API is used to get the real-time audio downstreaming volume levels of other members in the room. A number-type value will be returned. Value range: 0–200.

API prototype

```
GetRecvStreamLevel(openId: string) :number
```

Parameter	Type	Description
openId	string	`openId` of another member in the room

Sample code

```
number level =GetRecvStreamLevel(openId);
```

Dynamically setting the volume of a member of the room

This API is used to set the volume of a member in the room. It takes effect only on the local.

API prototype

```
SetSpeakerVolumeByOpenID(openId: string, volume:number) :number;
```

Parameter	Type	Description
openId	string	`OpenID` of the target user
volume	number	Percentage. Recommended value range: 0–200. Default value: `100` .

Sample code

```
context.SetSpeakerVolumeByOpenID(openId,  100);
```

Getting volume percentage

This API is used to get the volume level set by `SetSpeakerVolumeByOpenID` .

API prototype

```
GetSpeakerVolumeByOpenID(openId: string) :number;
```

Parameter	Type	Description
openId	string	`OpenID` of the target user

Returned values

API returns volume percentage set by OpenID, where 100 is by default.

Sample code

```
context.GetSpeakerVolumeByOpenID(openId);
```

Setting the speaker volume

This API is used to set the speaker volume.

API prototype

```
SetSpeakerVolume(volume:number) :number
```

Parameter	Type	Description
volume	number	Volume level. Value range: 0–200. Default value: `100`. `0` indicates that the audio is muted, while `100` indicates that the volume level remains unchanged.

Sample code

```
number vol = 100;
context.SetSpeakerVolume(vol);
```

Getting the speaker volume

This API is used to get the speaker volume. A number-type value will be returned to indicate the volume. `101` indicates that the `SetSpeakerVolume` API has not been called.
"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

API prototype

```
GetSpeakerVolume() :number
```

Sample code

```
numbet volume = context.GetSpeakerVolume();
```

Device Selection APIs

Device selection APIs can be used only on PC.

API	Description
GetMicListCount	Gets the number of mics.
GetMicList	Enumerates mics.
GetSpeakerListCount	Gets the number of speakers.
GetSpeakerList	Enumerates speakers.

SelectMic	Selects a mic.
SelectSpeaker	Selects a speaker.

Getting the number of mics

This API is used to get the number of mics.

Function prototype

```
GetMicListCount() :number
```

Sample code

```
var micListCount = context.GetMicListCount();
```

Enumerating mics

This API is used together with the `GetMicListCount` API to enumerate mics.

Function prototype

```
GetMicList() :GmeAudioDeviceInfo[];
```

Sample code

```
var micList = context.GetMicList();
```

Selecting mic

This API is used to select a mic. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default mic will be selected.
The 0th device id returned in the GetMicList API is the default device of the call device. If there is a selected call device, it will be maintained by service. If it is unplugged, the call device will be changed back into the default device.

Function prototype

```
SelectMic(micId: string) :number;
```

Parameter	Type	Description
micId	string	Mic ID, which is from the list returned by `GetMicList`.

Sample code

```
context.SelectMic(deviceID);
```

This API is used to get the number of speakers.

Function prototype

```
 GetSpeakerListCount() :number;
```

Sample code

```
context.GetSpeakerListCount();
```

Enumerating speakers

This API is used together with the `GetSpeakerListCount` API to enumerate speakers.

Function prototype

```
GetSpeakerList(): GmeAudioDeviceInfo[]
```

Sample code

```
var speakList = GetSpeakerList();
```

Selecting speaker

This API is used to select a playback device. If this API is not called or `DEVICEID_DEFAULT` is passed in, the default playback device will be selected.

Function prototype

```
SelectSpeaker(speakerId: string) :number
```

Parameter	Type	Description
speakerId	string	Speaker ID, which is from the list returned by `GetSpeakerList` .

Sample code

```
var ret = SelectSpeaker(deviceID);
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

API prototype

```
EnableLoopBack(bEnable: boolean) :number
```

Parameter	Type	Description
enable	boolean	Specifies whether to enable.

Sample code

```
context.EnableLoopBack(true);
```

Getting user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, please see the `EnterRoom` API.

API prototype

```
GetRoomType() :number
```

Sample code

```
context.GetRoomType();
```

Changing the room type

This API is used to modify a user's room audio type. For the result, please see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE`. The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

API prototype

```
ChangeRoomType(roomType: number) :number
```

Parameter	Type	Description
roomtype	number	Room type to be switched to. For room audio types, see the `EnterRoom` API.

Sample code

```
context.ChangeRoomType(ITMG_ROOM_TYPE_FLUENCY);
```

Callback event

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result` , `error_info` , and `new_room_type` . The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description
ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	The existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	A user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).
ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	A user is already in the room, and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	A room member calls the `ChangeRoomType` API to request a change of the room audio type.

Sample code

```
context.setTMGDelegate(function(eventId, msg){
  if (ITMGContext.ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE ==
type)
        {
        // Process room type events
      }
});
```

The monitoring event of room call quality

This is the quality monitoring event used to listen on the network quality. If your network conditions are poor, the business layer will ask you to switch the network through the UI. This event is triggered once every two seconds after room entry, and its message is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which are as detailed below:

Parameter	Type	Description
weight	number	Value range: 1–50. `50` indicates excellent sound quality, `1` indicates very poor (barely usable) sound quality, and `0` represents an initial meaningless value. Generally, if the value is below 30, you can remind users that the network is poor and recommend them to switch the network.
Loss	var	Upstream packet loss rate
Delay	number	Voice chat delay in ms

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
GetSDKVersion() :string
```

Sample code

```
context.GetSDKVersion();
```

Setting the application name and version

This API is used to set the application name and version.

API prototype

```
SetAppVersion(appVersion: string) : number
```

Parameter description

Parameter	Type	Description
appVersion	string	Application name and version

Sample code

```
context.SetAppVersion("gme V2.0.0");
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
SetLogLevel(level: number) : number
```

Parameter description

Parameter	Type	Description
level	number	Sets the log level. `TMG_LOG_LEVEL_NONE` indicates not to log. Default value: `TMG_LOG_LEVEL_INFO` .

`level` description:

level	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
context.SetLogLevel(TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

Platform	Path
Windows	%appdata%\\GMEGLOBAL\\GME\\ProcessName

API prototype

```
SetLogPath(logPath: string)
```

Parameter	Type	Description
logPath	string	Path

Sample code

```
string logDir = ""// Set a path by yourself
context.SetLogPath(logDir);
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

API prototype

```
GetQualityTips() :string
```

Sample code

```
string tips = context.GetQualityTips();
```

Callback message

Message	Description	Data
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room.	result; error
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room.	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected due to a network or another issue.	result; error

ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Room members were updated.	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	The reconnection to the room started.	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	The reconnection to the room succeeded.	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched.	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status was changed.	result; error_info; sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started.	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped.	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker was changed.	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A new speaker was added.	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker was lost.	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A new mic was added.	result; error

ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic was lost.	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic was changed.	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	The room network quality changed.	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Voice message recording was completed.	result; file_p
ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Voice message upload was completed.	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Voice message download was completed.	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Voice message playback was completed.	result; file_p
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast speech-to-text conversion was completed.	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion	result; file_p text;file_id

	was completed.	
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	Streaming speech-to-text conversion is in progress.	result; file_p text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed.	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed.	result; text;file_id

Speech-to-Text Service

Last updated：2024-01-18 15:15:48

This document describes how to integrate with and debug GME client APIs for the voice messaging and speech-to-text services for Electron.

Key Considerations for Using GME

GME provides the real-time voice service and voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll`.

Notes

You have created a GME application and obtained the SDK `AppID` and key. For more information, see Activating Services.
You have activated **GME real-time voice service and voice messaging and speech-to-text services**. For more information, see Activating Services.
Configure your project before using GME; otherwise, the SDK will not take effect.
After a GME API is called successfully, `GmeError.AV_OK` will be returned with the value being `0`.
GME APIs should be called in the same thread.
The `Poll` API should be called periodically for GME to trigger event callbacks.
For detailed error codes, see Error Codes.

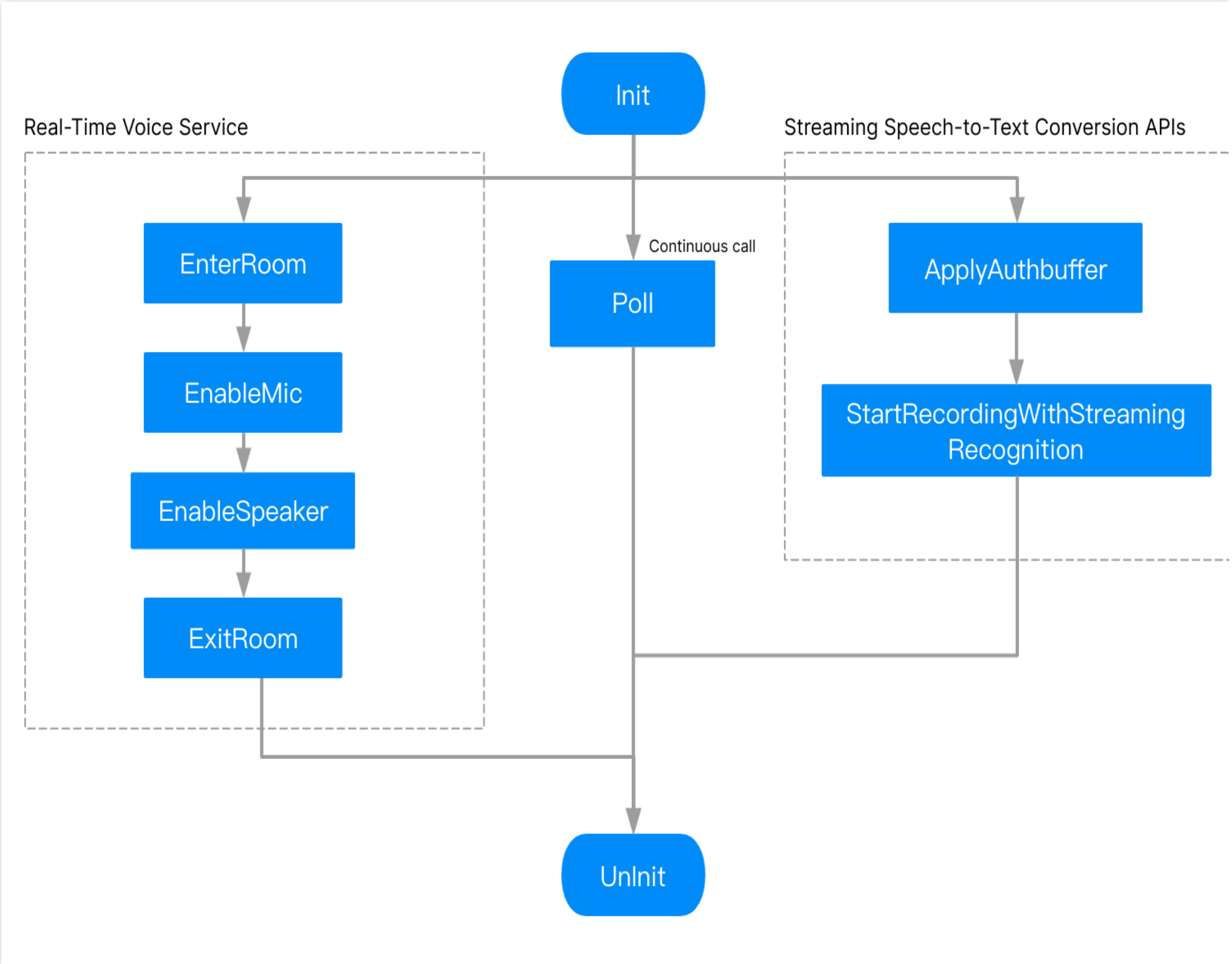
Note:

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.
Non-streaming speech-to-text API **PttSpeechToText()**: There can be up to 10 concurrent requests per account.
Streaming speech-to-text API **PttStartRecordingWithStreamingRecognition()**: There can be up to 50 concurrent requests per account.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initializing GME, API: Init
2. Calling Poll periodically to trigger event callbacks, API: Poll
3. Initializing authentication, API: ApplyPTTAuthbuffer
4. Starting streaming speech recognition, API: PttStartRecordingWithStreamingRecognition
5. Stopping recording, API: PttStopRecording
6. Uninitializing GME, API: UnInit

TS class

```
`GmeContext`: GME business implementation APIs
`GmeError`: GME error code definition class
```

Core APIs

API	Description
Init	Initializes GME
Poll	Triggers event callback
Uninit	Uninitializes GME

Importing the GME module

```
const { GmeContext } = require('gme-electron-sdk');
```

Getting an instance

var m_context = new GmeContext();

Initializing the 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

```
Init(appid: string, openid: string): number;
```

Parameter	Type	Description
sdkAppId	string	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	string	`openID` can only be in `Int64` 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.

Returned values

Returned Value	Description
GmeError.AV_OK= 0	Initialized the SDK successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Checks whether the SDK file is complete. It is recommended to delete it and then import the SDK again.

Notes on 7015 error code

The 7015 error code is judged by MD5. If this error is reported during integration, please check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Sample code

```
number ret = m_context.Init(sdkAppId, openID);
// Determine whether the initialization is successful by the returned value
if (ret != GmeError.AV_OK)
{
     console.log("Failed to initialize the SDK:");
        return;
}
```

Triggering event callback

Event callbacks can be triggered by calling the `Poll` API in the timer. 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.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
Poll();
```

Sample code

```
setInterval(function () {
      m_context.Poll();
    }, 50);
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

API prototype

```
Uninit() : number
```

Voice Messaging and Speech-to-Text Services

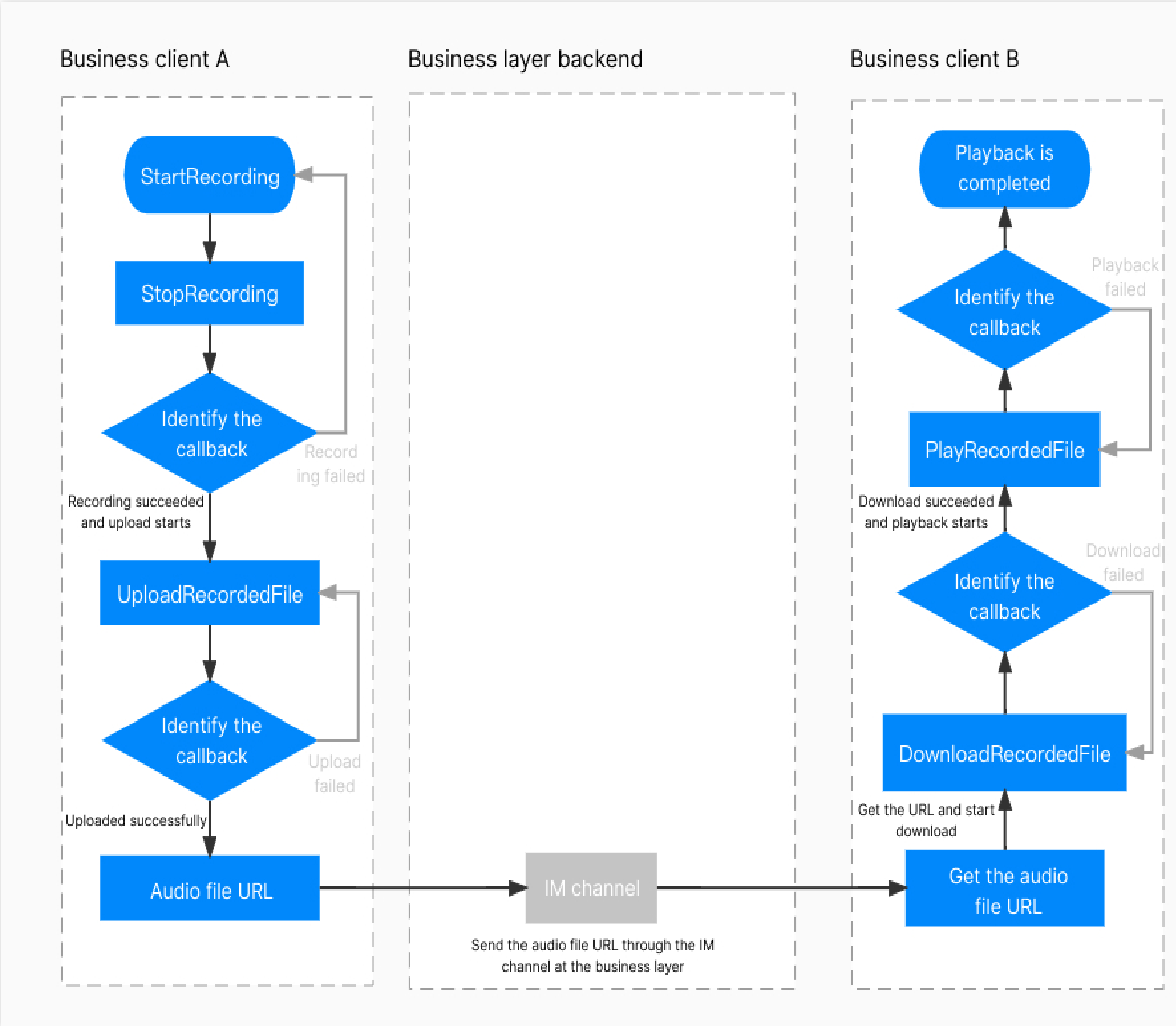
Note:

The speech-to-text service consists of fast recording-to-text conversion and streaming speech-to-text conversion.

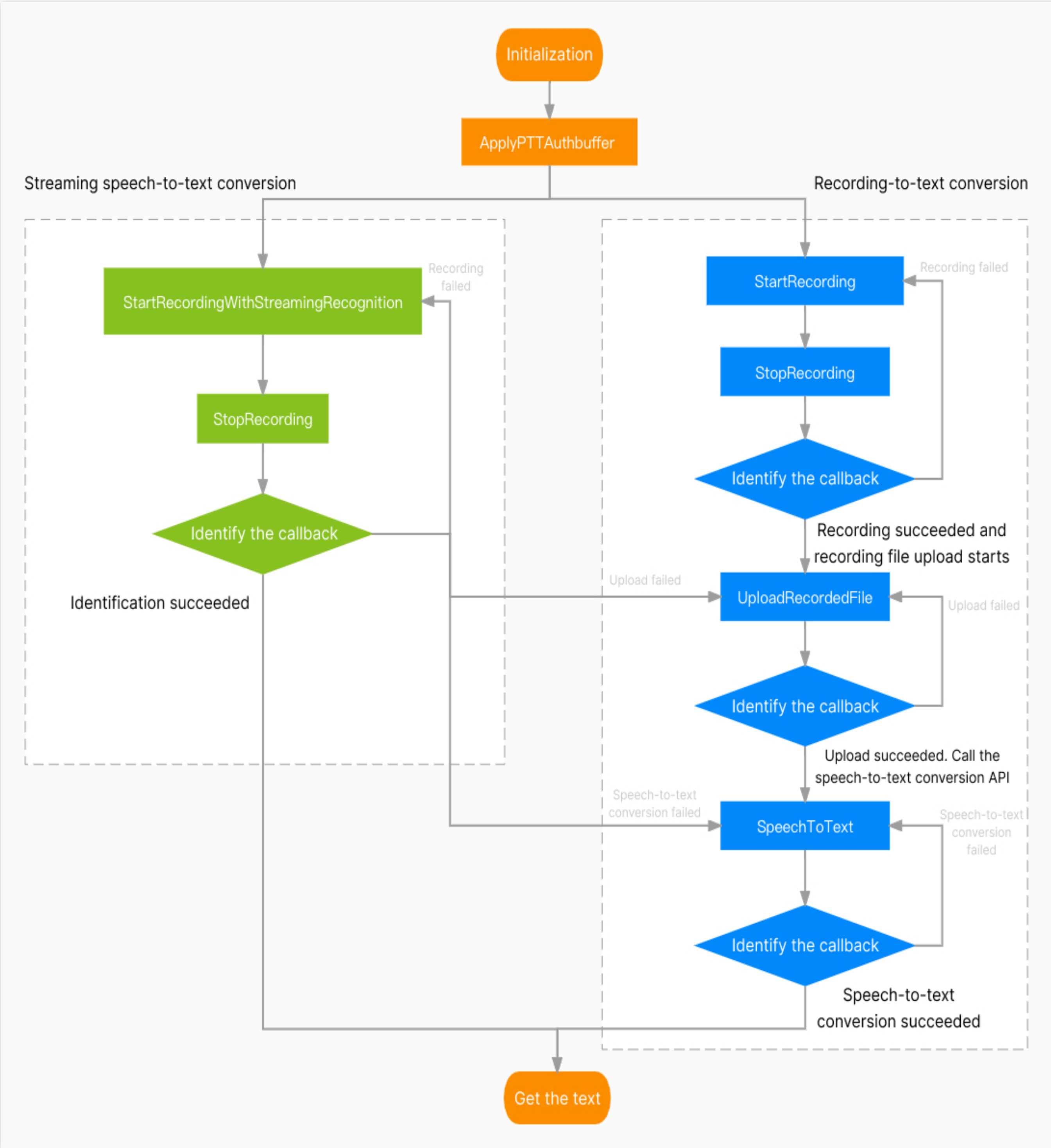
You do not need to enter a voice chat room when using the voice messaging service.

The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to modify the maximum recording duration to 10 seconds, call the `SetMaxMessageLength` API to set it after initialization.

Flowchart for using the voice message service



Flowchart for using the speech-to-text service



API	Description
GenAuthBuffer	Gets the authentication information
SetMaxMessageLength	Specifies the maximum length of voice message

Generating the local authentication key

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

API prototype

```
GenAuthBuffer(appId: string,roomId: string, openId:string, appKey: string)
:string
```

Parameter	Type	Description
appId	string	`AppId` from the Tencent Cloud console.
roomId	string	Enter `null` or an empty string
openId	string	User ID, which is the same as `OpenId` during initialization.
key	string	Permission key from the Tencent Cloud console.

Application authentication

After the authentication information is generated, the authentication is assigned to the SDK.

API prototype

```
ApplyPTTAuthbuffer(authBuffer: string) :number
```

Parameter	Type	Description
authBuffer	string	Authentication information

Sample code

```
var authBuffer = m_context.GetAuthBuffer(UserConfig.GetAppID(),
UserConfig.GetUserID(), null,UserConfig.GetAuthKey());
m_context.ApplyPTTAuthbuffer(authBuffer);
```

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

API prototype

```
PttSetMaxMessageLength(msTime: number) :number
```

Parameter	Type	Description
msTime	number	Audio duration in ms. Value range: 1000 < `msTime` <= 58000

Sample code

```
m_context.PttSetMaxMessageLength(58000);
```

Streaming Speech Recognition

Voice messaging and speech-to-text APIs

API	Description
PttStartRecordingWithStreamingRecognition	Starts streaming recording
PttStopRecording	Stops recording

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** Stop recording.

API prototype

```
PttStartRecordingWithStreamingRecognition(filePath: string, speechLanguage:
string, translateLanguage: string) :number
```

Parameter	Type	Description
filePath	string	Path of stored audio file
speechLanguage	string	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translateLanguage	string	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

```
string filePath = "xx/xxx/xxx.silk"
```

```
var ret = m_context.StartRecordingWithStreamingRecognition(filePath,"cmn-Hans-CN","cmn-Hans-CN");
if (ret == 0) {
    this.currentStatus = "Start streaming recording";
} else {
    this.currentStatus = "Failed to start streaming recording";
}
```

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen on callback messages in the `OnEvent` notification, which is as detailed below:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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 callback notification based on the actual needs. The passed parameters include the following four messages.

Message Name	Description
result	Return code indicating whether 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

Note:

The file_id is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRecognition_IS_RUNNING' message is listened.

Error codes

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 conversion 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.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

If the error code 4098 is reported, see Speech-to-text Conversion for solutions.

Sample code

```
m_context.setTMGDelegate(function(eventId, msg){
   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;
                    }
                }
});
```

Voice Message Recording

The recording process is as follows: start recording > stop recording > return recording callback > start the next recording.

Voice messaging and speech-to-text APIs

API	Description
PttStartRecording	Starts recording
PttPauseRecording	Pauses recording
PttResumeRecording	Resumes recording
PttStopRecording	Stops recording
PttCancelRecording	Cancels recording

Starting recording

This API is used to start recording.

API prototype

```
PttStartRecording(filePath: string) : number;
```

Parameter	Type	Description
filePath	string	Path of stored audio file

Sample code

```
string filepath = "xxxx/xxx.silk";
var ret = m_context.PttStartRecording(filepath);
```

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

```
PttStopRecording() :number;
```

Sample code

```
m_context.PttStopRecording();
```

Callback for recording start

A callback will be executed through a delegate function to pass a message when recording is completed.

To stop recording, call `StopRecording`. The callback for recording start will be returned after the recording is stopped.

Parameter	Type	Description
code	string	0: Recording is completed
filepath	string	Path of stored recording file, which must be accessible and cannot be the `fileid`

Error codes

Error Code Value	Cause	Suggested Solution
4097	Parameter is empty.	Check whether the API parameters in the code are correct.
4098	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Ensure that the SDK recording feature is used at the right time.
4100	Audio data is not captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	Mic permission is required for using the SDK. To add the permission, see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

```
m_context.setTMGDelegate(function(eventId, msg){
   switch (eventType) {
                   case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
```

```
                        {
                        // Process
                        break;
                            }
                        ...
                                case ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE:
                        {
                        // Process
                        break;
                        }
                    }
});
```

Pausing recording

This API is used to pause recording. If you want to resume recording, call the `PttResumeRecording` API.

API prototype

```
PttPauseRecording() : number
```

Sample code

```
number ret = m_context.PttPauseRecording();
```

Resuming recording

This API is used to resume recording.

API prototype

```
PttResumeRecording() : number;
```

Sample code

```
number ret = m_context.PttResumeRecording();
```

Canceling recording

This API is used to cancel recording. **There is no callback after cancellation**.

API prototype

```
PttCancelRecording() : number
```

Sample code

```
m_context.PttCancelRecording();
```

Voice Message Upload, Download, and Playback

API	Description
PttUploadRecordedFile	Uploads an audio file
PttDownloadRecordedFile	Downloads an audio file
PttPlayRecordedFile	Plays back an audio file
PttStopPlayFile	Stops playing back an audio file
PttGetFileSize	Gets the audio file size
PttGetVoiceFileDuration	Gets the audio file duration

Uploading an audio file

This API is used to upload an audio file.

API prototype

```
PttUploadRecordedFile(filePath: string) : number
```

Parameter	Type	Description
filePath	String	Path of uploaded audio file, which is a local path.

Sample code

```
var ret = m_context.PttUploadRecordedFile(filePath);
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	number	`0` : Recording is completed
filepath	string	Path of stored recording file
fileid	string	File URL

Error codes

Error Code Value	Cause	Suggested Solution
8193	An error occurred while accessing the file during upload.	Ensure the existence of the file and the validity of the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
8195	A network error occurred.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device can access the internet.
8200	No `appinfo` is set.	Check whether the `apply` API is called or whether the input parameters are empty.

Sample code

```
m_context.setTMGDelegate(function(eventId, msg){
 switch (eventType) {
            case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
            {
```

```
              // Process
              break;
                  }
              ...
                      case ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE:
              {
              // Process
              break;
              }
          }
});
```

Downloading the audio file

This API is used to download an audio file.

API prototype

```
PttDownloadRecordedFile(fileId: string, filePath: string) : number
```

Parameter	Type	Description
fileId	string	File URL
filePath	string	Local path of saved file, which must be accessible and cannot be the `fileid`

Sample code

```
var ret = m_context.PttDownloadRecordedFile(fileID,filePath);
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	number	`0` : Download is completed
filepath	string	Path of stored recording file
fileid	string	URL of recording file, which will be retained on the server for 90 days.

Error codes

Error Code Value	Cause	Suggested Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.
12291	Network storage system exception.	The server failed to get the audio file. Check whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	Server file system error.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed during the process of getting the download parameters.	Check whether the device can access the internet.
12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice message and speech-to-text feature is initialized.

Sample code

```
m_context.setTMGDelegate(function(eventId, msg){
 switch (eventType) {
            case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
             {
             // Process
             break;
              }
        }
});
```


Playing back audio

This API is used to play back audio.

API prototype

```
PttPlayRecordedFile(filePath: string, voiceType: ITMG_VOICE_TYPE) : number
```

Parameter	Type	Description
filePath	string	Local audio file path
voicetype	ITMG_VOICE_TYPE	Voice changer type. For more information, see Voice Changing.

Error codes

Error Code Value	Cause	Suggested Solution
20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
m_context.PlayRecordedFile(filePath);
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameter includes `result` and `file_path` .

Parameter	Type	Description
code	number	`0` : Playback is completed
filepath	string	Path of stored recording file

Error codes

Error Code Value	Cause	Suggested Solution
20481	Initialization error.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.

20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	Parameter is empty.	Check whether the API parameters in the code are correct.
20484	Internal error.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
m_context.setTMGDelegate(function(eventId, msg){
 switch (eventType) {
        case ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE:
        {
        // Process
        break;
         }
        }
});
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

API prototype

```
PttStopPlayFile() : number
```

Sample code

```
m_context.PttStopPlayFile();
```

Getting audio file size

This API is used to get the size of an audio file.

API prototype

```
PttGetFileSize(filePath: string) : number
```

Parameter	Type	Description
filePath	string	Path of audio file, which is a local path

Sample code

```
m_context.PttGetFileSize(filePath);
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

API prototype

```
PttGetVoiceFileDuration(filePath: string) : number
```

Parameter	Type	Description
filePath	string	Path of audio file, which is a local path

Sample code

```
number fileDuration = m_context.PttGetVoiceFileDuration(filePath);
```

Fast Recording-to-Text Conversion

Translating audio file into text in specified language

This API can specify a language for recognition or translate the information recognized in speech into a specified language and return the translation.

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

API prototype

```
PttSpeechToText(fileID: string, speechLanguage: string, translateLanguage:
string) : number
```

Parameter	Type	Description
fileID	string	URL of audio file, which will be retained on the server for 90 days

speechLanguage	string	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translatelanguage	string	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

```
m_context.PttSpeechToText(filePath,"cmn-Hans-CN","cmn-Hans-CN");
```

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result` , `file_path` and `text` (recognized text).

Parameter	Type	Description
result	number	`0` : Recording is completed
fileid	string	URL of recording file, which will be retained on the server for 90 days
text	string	Converted text

Error codes

Error Code Value	Cause	Suggested Solution
32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	Network failed.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.

32784	Incorrect speech-to-text conversion parameter.	Check whether the API parameter `fileid` in the code is empty.
32785	Speech-to-text translation returned an error.	Error with the backend of voice messaging and speech-to-text feature. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

Sample code

```
m_context.setTMGDelegate(function(eventId, msg){
 switch (eventType) {
         case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
            {
            // Process
            break;
                }
            ...
                    case ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE:
            {
            // Process
            break;
            }
});
```

Voice Message Volume Level APIs

API	Description
PttGetMicLevel	Gets real-time mic volume level
PttSetMicVolume	Sets recording volume level
PttGetMicVolume	Gets recording volume level

PttGetSpeakerLevel	Gets real-time speaker volume level
PttSetSpeakerVolume	Sets playback volume
PttGetSpeakerVolume	Gets playback volume level

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. A number-type value will be returned. Value range: 0–200.

API prototype

```
PttGetMicLevel():number
```

Sample code

```
m_context.PttGetMicLevel();
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

API prototype

```
PttSetMicVolume(vol:number) :number
```

Parameter	Type	Description
vol	number	Value range: 0–200. Default value: `100`. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
m_context.PttSetMicVolume(vol);
```

Getting the recording volume of voice message

This API is used to get the recording volume of voice message. A number-type value will be returned. Value range: 0–200.

API prototype

```
PttGetMicVolume() : number
```

Sample code

```
m_context.PttGetMicVolume();
```

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. A number-type value will be returned. Value range: 0-200.

API prototype

```
PttGetSpeakerLevel() : number;
```

Sample code

```
m_context.PttGetSpeakerLevel();
```

Setting the playback volume of voice message

This API is used to set the playback volume of voice message. Value range: 0-200.

API prototype

```
PttSetSpeakerVolume(vol: number) : number
```

Sample code

```
m_context.PttSetSpeakerVolume(100);
```

Getting the playback volume of voice message

This API is used to get the playback volume of voice message. A number-type value will be returned. Value range: 0-200.

API prototype

```
PttGetSpeakerVolume() : number
```

Sample code

```
m_context.PttGetSpeakerVolume();
```

Advanced APIs

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
GetSDKVersion() :string
```

Sample code

```
string sdkVersion = m_context.GetSDKVersion();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
SetLogLevel(level: number) : number
```

Parameter description

Parameter	Type	Description
level	ITMG_LOG_LEVEL	Sets the log level. `TMG_LOG_LEVEL_NONE` indicates not to log. Default value: `TMG_LOG_LEVEL_INFO` .

`level` description:

Value of `level`	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
m_context.SetLogLevel(TMG_LOG_LEVEL_INFO);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

OS	Path
Windows	%appdata%\\GMEGLOBAL\\GME\\ProcessName

API prototype

```
SetLogPath(logPath: string)
```

Parameter	Type	Description
logPath	string	Path

Sample code

```
string logDir = ""// Set a path by yourself
m_context.SetLogPath(logDir);
```

SDK for Flutter

Integrating SDK

Last updated：2024-01-18 15:15:48

This document describes how to configure a Flutter project for the GME APIs for Flutter.

Supported Platforms

The GME SDK for Flutter supports iOS and Android platforms.

Importing the SDK

Step 1. Download the GME SDK for Flutter

Download the SDK file in SDK Download Guide, which contains the GME plugin. Decompress the SDK file to a local directory.

Step 2. Add dependencies of the GME plugin to the Flutter project

Add GME dependencies to the `pubspec.yaml` file in your Flutter project. Note that the `path` parameter is the path where the SDK file is decompressed to.

```
dependencies:
  flutter:
    sdk: flutter
  gme:
    path: ../
```

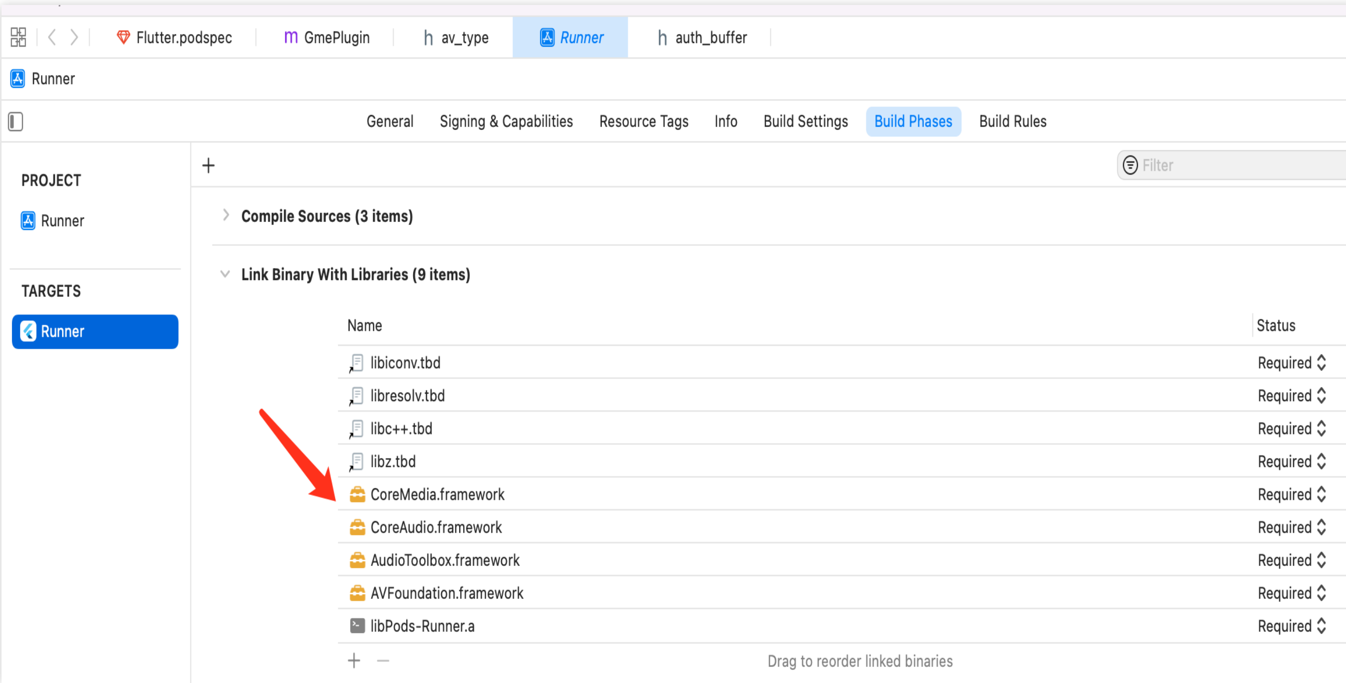
After saving the `pubspec.yaml` file, run the `flutter pub get` command on the CLI to make the GME plugin in the project take effect (if the Flutter plugin is configured in your IDE, this command will be executed automatically once the file is saved).

```
flutter pub get
```

Modifying the iOS Project

1. In Terminal, go to the iOS project directory of your Flutter project and run `pod install`.

2. In the Xcode project, configure the following GME dependent library files (you can skip this step if such dependencies already exist in your project):



3. The GME SDK for iOS requires the following permissions:
Required background modes: Allows running in the background (optional).
Microphone Usage Description: Allows access to microphone.

Modifying the Android Project

1. As GME requires permissions such as call permission and uses the permission management plugin `flutter permission-handler`, you need to modify the project as follows to use the SDK for Android 31 or later (skip this step if the SDK is already in the project):

```
android {
    compileSdkVersion 31

    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }

    kotlinOptions {
        jvmTarget = '1.8'
    }

    sourceSets {
        main.java.srcDirs += 'src/main/kotlin'
    }

    defaultConfig {
        // TODO: Specify your own unique Application ID (https://developer.android.com/studio/build/applica
        applicationId "com.example.flutter_android"
        minSdkVersion 16
        targetSdkVersion 31
        versionCode flutterVersionCode.toInteger()
        versionName flutterVersionName
    }

    buildTypes {
        release {
```

2. Add the project permissions to the Flutter project file `android/app/src/AndroidManifest.xml` (skip this step if such permissions have been added):

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

Real-time Voice

Last updated：2024-01-18 15:15:48

This document describes how to integrate with and debug GME client APIs for the real-time voice chat feature for Flutter.

Key Considerations for Using GME

GME provides the real-time voice chat service, voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll` .

Notes

You have created a GME application and obtained the SDK `AppID` and key. For more information, see Activating Services.

You have activated **GME real-time voice chat, voice messaging and speech-to-text services**. For more information, see Activating Services.

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `GmeError.AV_OK` will be returned with the value being `0` .

GME APIs should be called in the same thread.

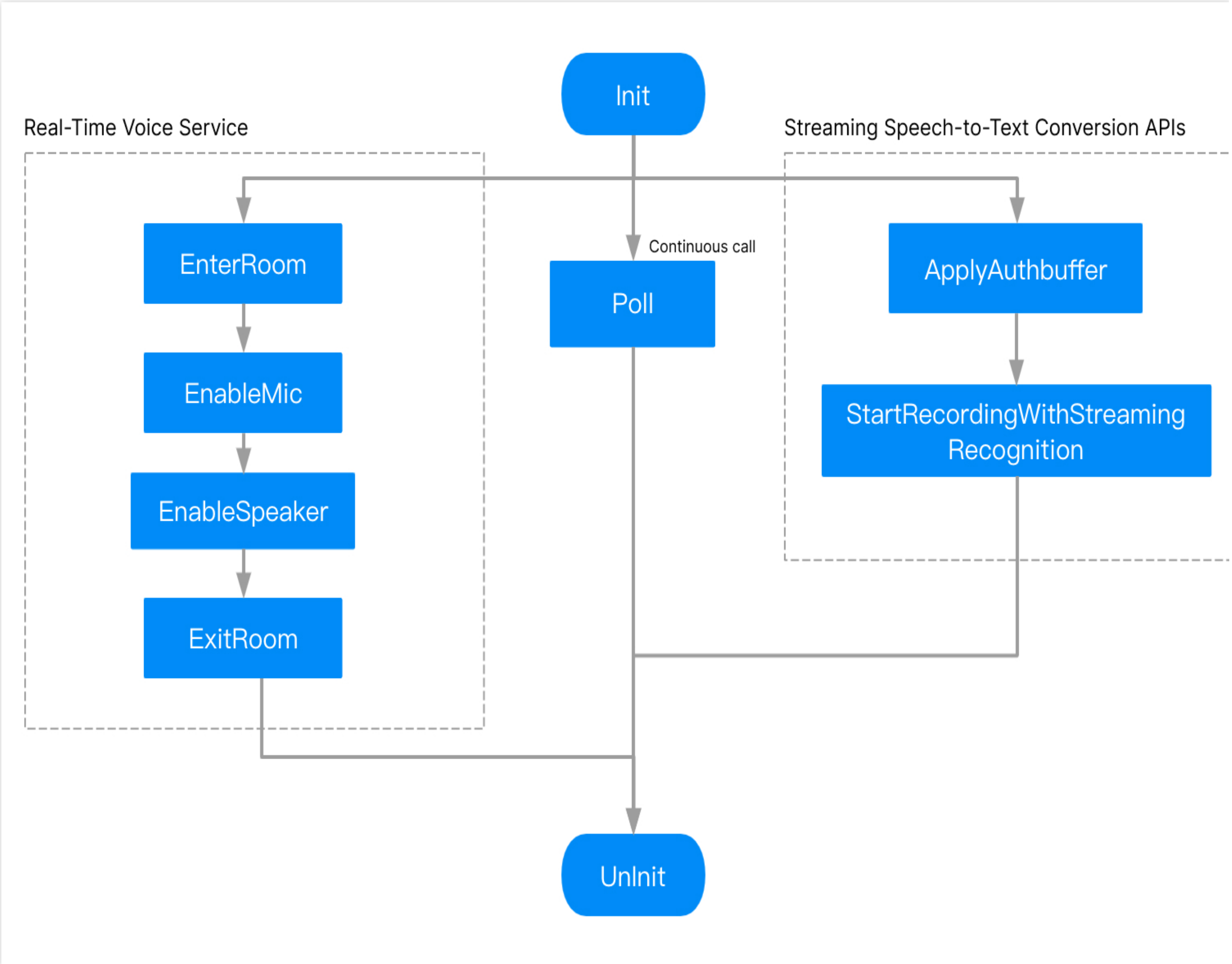
The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error codes, see Error Codes.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initialize GME.
2. Call `Poll` periodically to trigger event callbacks.
3. Enter a voice chat room.
4. Turn on the mic.
5. Turn on the speaker.
6. Exit the voice chat room.
7. Uninitialize GME.

Core APIs

API	Description
Init	Initializes GME.
Poll	Triggers the event callback.

Pause	Pauses the system.
Resume	Resumes the system.
Uninit	Uninitializes GME.

Importing the GME module

```
import 'package:gme/gme.dart';
import 'package:gme/gmeType.dart';
```

Getting an instance

To use the voice chat feature, get the `GmeSDK` object first.

```
ITMGContext context = ITMGContext.GetInstance();
```

Initializing the 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
Future<int> InitSDK(String appID, String openID)
```

Parameter	Type	Description
sdkAppId	string	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	string	`openID` can only be in `int64` 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.

Returned values

Returned Value	Description
GmeError.AV_OK= 0	SDK initialized successfully.

AV_ERR_SDK_NOT_FULL_UPDATE=7015	Solution: Check whether the SDK file is complete. We recommend that you delete it and then import the SDK again.

Note:

Notes on 7015 error code

The 7015 error code is identified by MD5. If this error is reported during integration, check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Due to the third-party enhancement, the Unity packaging mechanism, and other factors, the MD5 value of the library file will be affected, resulting in misjudgment. Therefore, **ignore this error in the logic for official releases**, and avoid displaying it on the UI.

Sample code

```
string SDKAPPID3RD = "14000xxxxx";
string openId="10001";
int res = await ITMGContext.GetInstance().InitSDK(SDKAPPID3RD, openId);

if (ret != GmeError.AV_OK)
{
    print("Init SDK Error");
    return;
}
```

Setting callbacks

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.

```
// When initializing the SDK
ITMGContext.GetInstance().SetEvent(handleEventMsg);
// Callback method
void handleEventMsg(int eventType, String data) async {
    // enterRoom event
    print("AddDelegate3" + eventType.toString());
    switch (eventType) {
      case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
        {
        // Callback of room entry
```

```
        }
        break;
      case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE:
        {
         // Callback of room switch
        }
        break;
    }
}
```

Triggering event callback

You need to periodically call the `Poll` API to trigger event callbacks. 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.

Note:

Call the `Poll` API periodically

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
Future<void> Poll();
```

Sample code

```
  Future<void> pollTimer() async {_pollTimer =
Timer.periodic(Duration(milliseconds: 100), (Timer timer) {
      ITMGContext.GetInstance().Poll();
    });
  }
```

Pausing the system

When a `Pause` event occurs in the system, the engine should also be notified for pause. For example, when the application switches to the background (OnApplicationPause, isPause=True), and you do not need the background to play back the audio in the room, please call `Pause` API to pause the GME service.

API prototype

```
Future<int> Pause()
```

Resuming the system

When a `Resume` event occurs in the system, the engine should also be notified for resumption. The `Resume` API only supports resuming voice chat.

API prototype

```
Future<int> Resume()
```

Uninitializing SDK

This API is used to uninitialize the SDK. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to initialize again.**

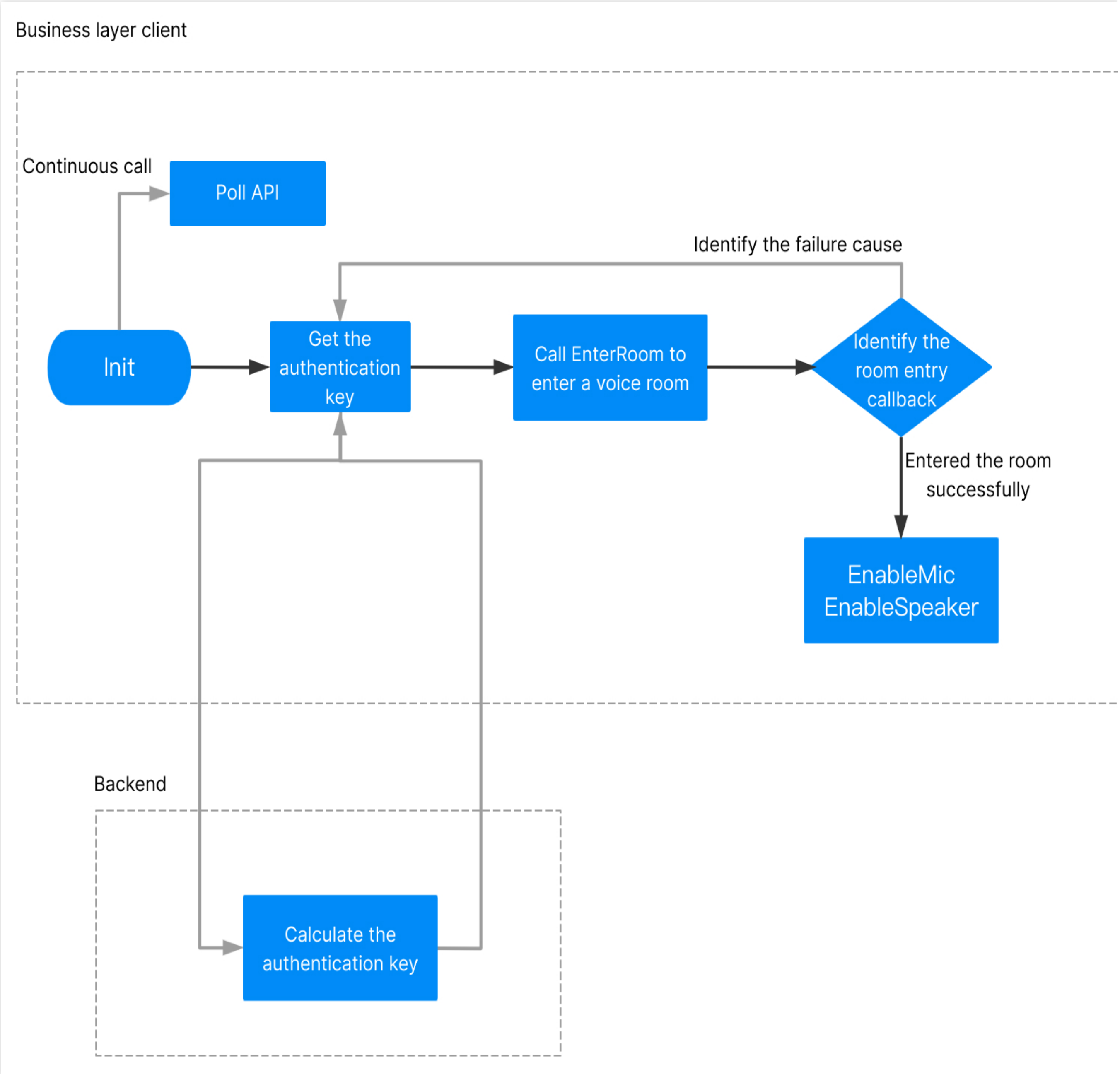
API prototype

```
Future<int> Uninit()
```

Voice Chat Room APIs

You should initialize and call the SDK to enter a room before voice chat can start.

If you have any questions when using the service, see Sound and Audio.



API	Description
GenAuthBuffer	Calculates the local authentication key.
EnterRoom	Enters a room.
ExitRoom	Exits a room.
IsRoomEntered	Determines whether room entry is successful.

Local authentication key calculation

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

API prototype

```
Future<Uint8List> GenAuthBuffer(String appID, String roomID, String openID,
String key)
```

Parameter	Type	Description
appID	string	`AppID` from the Tencent Cloud console
roomID	string	Room ID, which can contain up to 127 characters.
openID	string	User ID, which is the same as `openID` during initialization.
key	string	Permission key from the Tencent Cloud console.

Sample code

```
 Uint8List userSig = await
ITMGContext.GetInstance().GenAuthBuffer(_editAppID.text, _editRoomID.text,
_editOpenID.text, _editKey.text);
 int res = await ITMGContext.GetInstance().EnterRoom(_editRoomID.text, 1,
userSig);
```

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.

Note:

If the room entry callback result is `0`, the room entry is successful. If `0` is returned from the `EnterRoom` API, it doesn't necessarily mean that the room entry is successful.

The audio type of the room is determined by the first user entering the room. After that, if a member in the room changes the room type, it will take effect for all members there. For example, if the first user entering the room uses the smooth sound quality, and the second user entering the room uses the HD sound quality, the room audio type of the second user will change to the smooth sound quality. Only after a member in the room calls the `ChangeRoomType` API will the audio type of the room be changed.

API prototype

```
Future<int> EnterRoom(String roomID, int roomType, Uint8List authBuffer)
```

Parameter	Type	Description
roomId	string	Room ID, which can contain up to 127 characters.
roomType	ITMGRoomType	Room type. We recommend that you select `ITMG_ROOM_TYPE_FLUENCY` for games. For more information on room audio types, see Sound Quality.
appKey	Uint8List	Authentication key

Sample code

```
int res = await ITMGContext.GetInstance().EnterRoom(_editRoomID.text, 1,
authBuffer);
```

Callback for room entry

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

Note:

Billing references:

Purchase Guide

Billing

Will the billing continue if the client is disconnected from the server when using the voice chat?

Sample code

```
// Listen on an event:
void handleEventMsg(int eventType, String data) async {
  switch (eventType) {
      case ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
      {
        // The process after room entry
      }
  }
}
ITMGContext.GetInstance().SetEvent(handleEventMsg);
```

Data details

Message	Data	Sample
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	result;	{"error_info":"","result":0}

	error_info	
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	result; error_info	{"error_info":"waiting timeout, please check your network","result":0}

If the network is disconnected, there will be a disconnection callback notification `ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT` . At this time, the SDK will automatically reconnect, and the callback is `ITMG_MAIN_EVENT_TYPE_RECONNECT_START` . When the reconnection is successful, there will be a callback `ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS` .

Error codes

Error Code	Cause and 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	The user was already in another room.
1001	The user was already in the process of entering a room but repeated this operation. We recommend that you not 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.

Exiting a room

This API is used to exit the current room. It is an async API. The returned value `AV_OK` indicates a successful async delivery. If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API; instead, you can directly call the `EnterRoom` API.

API prototype

```
Future<int> ExitRoom()
```

Sample code

```
ITMGContext.GetInstance().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:

Sample code

```
void handleEventMsg(int eventType, String data) async{
  switch (eventType) {
      case ITMG_MAIN_EVENT_TYPE_EXIT_ROOM:
      {
      // The process after room exit
        break;
      }
  }
}
ITMGContext.GetInstance().SetEvent(handleEventMsg);
```

Determining whether user has entered room

This API is used to determine whether the user has entered a room. A value in bool type will be returned. Do not call this API during room entry.

API prototype

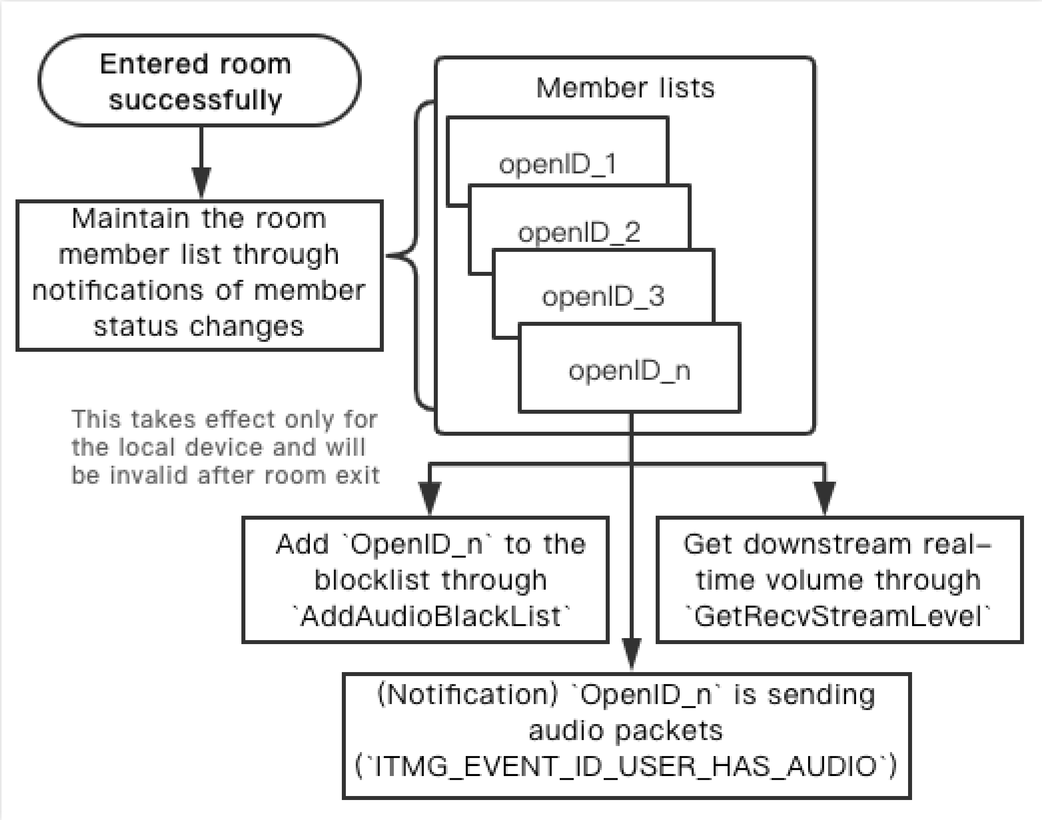
```
Future<bool> IsRoomEntered()
```

Sample code

```
bool res = await ITMGContext.GetInstance().IsRoomEntered();
```

Room Status Maintenance

APIs in this section are used to display speaking members and members entering or exiting the room and mute a member in the room at the business layer.



API/Notification	Description
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	The member status changed.
AddAudioBlackList	Mutes a member in the room.
RemoveAudioBlackList	Unmutes a user.
IsOpenIdInAudioBlackList	Queries whether the user of the specified `openid` is muted.

Notification events of member room entry and speaking status

This event is used to get speaking users in the room and display the users on the UI, and to send a notification when someone enters or exits the room.

A notification for this event will be sent only when the status changes. To get the member status in real time, cache the notification when it is received at the business layer. The event message `ITMG_MAIN_EVNET_TYPE_USER_UPDATE` containing `event_id` , `count` , and `openIdList` will be returned, which will be identified in the `OnEvent` notification.

Notifications of the `EVENT_ID_ENDPOINT_NO_AUDIO` audio event will be sent only when the threshold is exceeded, that is, other members in the room can receive the notification that the local user stops speaking only after the local client captures no voice for two seconds.

The audio event returns only the member speaking status but not the specific volume level. If you need the specific volume levels of members in the room, you can use the `GetVolumeById` API.

event_id	Description	Maintenance
EVENT_ID_ENDPOINT_ENTER	Return the `openid` of the member entering the room.	Member list
EVENT_ID_ENDPOINT_EXIT	Return the `openid` of the member exiting the room.	Member list
EVENT_ID_ENDPOINT_HAS_AUDIO	Return the `openid` of the member sending audio packets in the room. This event can be used to determine whether a user is speaking and display the voiceprint effect.	Chat member list
EVENT_ID_ENDPOINT_NO_AUDIO	Return the `openid` of the member stopping sending audio packets in the room.	Chat member list

Sample code

```
void handleEventMsg(int eventType, String data) async {
  if (eventType == ITMG_MAIN_EVENT_TYPE_ENTER_ROOM)
  {
               // Process
            switch (eventID)
             {
             case EVENT_ID_ENDPOINT_ENTER:
                  // A member enters the room
                  break;
             case EVENT_ID_ENDPOINT_EXIT:
                  // A member exits the room
                break;
            case EVENT_ID_ENDPOINT_HAS_AUDIO:
                // A member sends audio packets
                break;
            case EVENT_ID_ENDPOINT_NO_AUDIO:
                // A member stops sending audio packets
                break;

            default:
                break;
             }
        break;
  }
}
```

Muting a member in the room

This API is used to add an ID to the audio data blocklist. This operation blocks audio from someone and only applies to the local device without affecting other devices. The returned value `0` indicates that the call is successful.

Assume that users A, B, and C are all speaking using their mic in a room:

If A blocks C, A can only hear B;

If B blocks neither A nor C, B can hear both of them;

If C blocks neither A nor B, C can hear both of them.

This API is suitable for scenarios where a user is muted in a room.

API prototype

```
Future<int> AddAudioBlackList(String openID)
```

Parameter	Type	Description
openID	string	`openid` of the user to be blocked

Sample code

```
res = await
ITMGContext.GetInstance().GetAudioCtrl().AddAudioBlackList(_editRoomManagerID.t
ext);
```

Unmuting

This API is used to remove an ID from the audio data blocklist. A returned value of 0 indicates the call is successful.

API prototype

```
Future<int> RemoveAudioBlackList(String openID)
```

Parameter	Type	Description
openId	string	ID to be unblocked

Sample code

```
res = await
ITMGContext.GetInstance().GetAudioCtrl().RemoveAudioBlackList(_editRoomManagerI
D.text);
```

Voice Chat Capturing APIs

The voice chat APIs can only be called after SDK initialization and room entry.

When the user clicks the button of enabling/disabling the mic or speaker on the UI, we recommend you call the `EnableMic` or `EnableSpeaker` API.

To enable the user to press the mic button on the UI to speak and release it to stop speaking, we recommend you call `EnableAudioCaptureDevice` once during room entry and call `EnableAudioSend` to enable the user to speak while pressing the button.

API	Description
EnableMic	Enables/Disables the mic.
GetMicState	Gets the mic status.
EnableAudioCaptureDevice	Enables/Disables the capturing device.
IsAudioCaptureDeviceEnabled	Gets the capturing device status.
EnableAudioSend	Enables/Disables audio upstreaming.
IsAudioSendEnabled	Gets the audio upstreaming status.
GetMicLevel	Gets the real-time mic volume level.
GetSendStreamLevel	Gets the real-time audio upstreaming volume level.
SetMicVolume	Sets the mic volume level.
GetMicVolume	Gets the mic volume level.

Enabling or disabling mic

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

EnableMic = EnableAudioCaptureDevice + EnableAudioSend

API prototype

```
Future<int> EnableMic(bool enable)
```

Parameter	Type	Description
isEnabled	bool	To enable the mic, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
// Turn on mic
int res = await ITMGContext.GetInstance().GetAudioCtrl().EnableMic(true);
```

Getting the mic status

This API is used to get the mic status. The returned value 0 indicates that the mic is off, while 1 is on.

API prototype

```
Future<int> GetMicState()
```

Sample code

```
int micState = await ITMGContext.GetInstance().GetAudioCtrl().GetMicState();
```

Enabling or disabling capturing device

This API is used to enable/disable a capturing device. The device is not enabled by default after room entry.
This API can only be called after room entry. The device will be disabled automatically after room exit.
Operations such as permission application and volume type adjustment will generally be performed when a capturing device is enabled on a mobile device.

API prototype

```
Future<int> EnableAudioCaptureDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To enable the capturing device, set this parameter to `true` , otherwise, set it to `false` .

Sample code

```
// Enable capturing device
int res = await
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioCaptureDevice(true);
```

Getting the capturing device status

This API is used to get the status of a capturing device.

API prototype

```
Future<bool> IsAudioCaptureDeviceEnabled()
```

Sample code

```
bool res = await
ITMGContext.GetInstance().GetAudioCtrl().IsAudioCaptureDeviceEnabled();
```

Enabling or disabling audio upstreaming

This API is used to enable/disable audio upstreaming. If a capturing device is already enabled, it will send captured audio data; otherwise, it will remain mute. For more information on how to enable/disable the capturing device, see the `EnableAudioCaptureDevice` API.

API prototype

```
Future<int> EnableAudioSend(bool enable)
```

Parameter	Type	Description
isEnabled	bool	To enable audio upstreaming, set this parameter to `true` ; otherwise, set it to `false` .

Sample code

```
int res = await
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioSend(isCheck);
```

Getting audio upstreaming status

This API is used to get the status of audio upstreaming.

API prototype

```
Future<bool> IsAudioSendEnabled()
```

Sample code

```
bool IsAudioSend = await
ITMGContext.GetInstance().GetAudioCtrl().IsAudioSendEnabled();
```


Getting the real-time mic volume

This API is used to get the real-time mic volume level. A number-type value in the range of 0–100 will be returned. We recommend that you call this API once every 20 ms.

API prototype

```
Future<int> GetMicLevel()
```

Sample code

```
int res = await ITMGContext.GetInstance().GetAudioCtrl().GetMicLevel();
```

Getting the real-time audio upstreaming volume

This API is used to get the local real-time audio upstreaming volume level. A number-type value in the range of 0–100 will be returned.

API prototype

```
Future<int> GetSendStreamLevel()
```

Sample code

```
int res = await ITMGContext.GetInstance().GetAudioCtrl().GetSendStreamLevel();
```

Setting the mic software volume

This API is used to set the mic volume level. The corresponding parameter is `volume` , which is equivalent to attenuating or gaining the captured sound.

API prototype

```
 Future<int> SetMicVolume(int volume)
```

Parameter	Type	Description
volume	number	Value range: 0–200. Default value: \`100\`. \`0\` indicates that the audio is mute, while \`100\` indicates that the volume level remains unchanged.

Sample code

```
int volume = 100;
```

```
int res = await ITMGContext.GetInstance().GetAudioCtrl().SetMicVolume(volume);
```

Getting the mic software volume

This API is used to get the mic volume level. A number-type value will be returned. `101` indicates that the `SetMicVolume` API has not been called.

API prototype

```
Future<int> GetMicVolume()
```

Sample code

```
int res = await ITMGContext.GetInstance().GetAudioCtrl().GetMicVolume();
```

Voice Chat Playback APIs

API	Description
EnableSpeaker	Enables/Disables the speaker.
GetSpeakerState	Gets the speaker status.
EnableAudioPlayDevice	Enables/Disables the playback device.
IsAudioPlayDeviceEnabled	Gets the playback device status.
EnableAudioRecv	Enables/Disables audio downstreaming.
IsAudioRecvEnabled	Gets the audio downstreaming status.
GetSpeakerLevel	Gets the real-time speaker volume level.
GetRecvStreamLevel	Gets the real-time downstreaming audio levels of other members in the room.
SetSpeakerVolume	Sets the speaker volume level.
GetSpeakerVolume	Gets the speaker volume level.

Enabling or disabling speaker

This API is used to enable/disable the speaker. **EnableSpeaker = EnableAudioPlayDevice + EnableAudioRecv**

API prototype

```
Future<int> EnableSpeaker(bool enable)
```

Parameter	Type	Description
bEnable	bool	To disable the speaker, set this parameter to `false`; otherwise, set it to `true`.

Sample code

```
// Turn on the speaker
await ITMGContext.GetInstance().GetAudioCtrl().EnableSpeaker(isCheck);
```

Getting the speaker status

This API is used to get the speaker status. 0 indicates that the speaker is off, and 1 is on.

API prototype

```
Future<int> GetSpeakerState()
```

Sample code

```
int spkState = await
ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerState();
```

Enabling or disabling playback device

This API is used to enable/disable a playback device.

API prototype

```
Future<int> EnableAudioPlayDevice(bool enable)
```

Parameter	Type	Description
enable	bool	To disable the playback device, set this parameter to `false`; otherwise, set it to `true`.

Sample code

```
int res = await
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioPlayDevice(isCheck);
```

Getting the playback device status

This API is used to get the status of a playback device.

API prototype

```
Future<bool> IsAudioPlayDeviceEnabled()
```

Sample code

```
bool res = await
ITMGContext.GetInstance().GetAudioCtrl().IsAudioPlayDeviceEnabled();
```

Enabling or disabling audio downstreaming

This API is used to enable/disable audio downstreaming. If a playback device is already enabled, it will play back audio data from other members in the room; otherwise, it will remain mute. For more information on how to enable/disable the playback device, see the `EnableAudioPlayDevice` API.

API prototype

```
Future<int> EnableAudioRecv(bool enable)
```

Parameter	Type	Description
isEnabled	bool	To enable audio downstreaming, set this parameter to `true`; otherwise, set it to `false`.

Sample code

```
int res = await
ITMGContext.GetInstance().GetAudioCtrl().EnableAudioRecv(isCheck);
```

Getting audio downstreaming status

This API is used to get the status of audio downstreaming.

API prototype

```
Future<bool> IsAudioRecvEnabled()
```

Sample code

```
bool res = await ITMGContext.GetInstance().GetAudioCtrl().IsAudioRecvEnabled();
```

Getting the real-time speaker volume

This API is used to get the real-time speaker volume level. A number-type value will be returned to indicate the volume level. We recommend that you call this API once every 20 ms.

API prototype

```
Future<int> GetSpeakerLevel()
```

Sample code

```
bool res = await ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerLevel();
```

Getting the real-time downstreaming audio levels of other members in room

This API is used to get the real-time audio downstreaming volume of other members in the room. A number-type value will be returned. Value range: 0–200.

API prototype

```
Future<int> GetRecvStreamLevel(String openID)
```

Parameter	Type	Description
openId	string	`openId` of another member in the room

Sample code

```
int res = await
ITMGContext.GetInstance().GetAudioCtrl().GetRecvStreamLevel(_editRoomManagerID.
text);
```

Dynamically setting the volume of a member of the room

This API is used to set the volume of a member in the room. It takes effect only on the local.

API prototype

```
Future<int> SetSpeakerVolumeByOpenID(String openId, int volume)
```

Parameter	Type	Description
openId	string	`OpenID` of the target user
volume	number	Percentage. Recommended value range: 0-200. Default value: `100`.

Sample code

```
int res = await
ITMGContext.GetInstance().GetAudioCtrl().SetSpeakerVolumeByOpenID(_editRoomMana
gerID.text, 100);
```

Getting volume percentage

This API is used to get the volume level set by `SetSpeakerVolumeByOpenID` .

API prototype

```
Future<int> GetSpeakerVolumeByOpenID(String openId)
```

Parameter	Type	Description
openId	string	`OpenID` of the target user

Returned values

API returns volume percentage set by OpenID, where 100 is by default.

Sample code

```
int res = await
ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerVolumeByOpenID(_editRoomMana
gerID.text);
```

Setting the speaker volume

This API is used to set the speaker volume.

API prototype

```
Future<int> SetSpeakerVolume(int volume)
```

Parameter	Type	Description
volume	number	Volume level. Value range: 0-200. Default value: `100`. `0` indicates that the

		audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
int volume = value.toInt();
int res = await
ITMGContext.GetInstance().GetAudioCtrl().SetSpeakerVolume(volume);
```

Getting the speaker volume

This API is used to get the speaker volume. A number-type value will be returned to indicate the volume. `101` indicates that the `SetSpeakerVolume` API has not been called.

"Level" indicates the real-time volume, and "Volume" the speaker volume. The final volume = Level * Volume%. For example, if the "Level" is 100 and "Volume" is 60, the final volume is "60".

API prototype

```
Future<int> GetSpeakerVolume()
```

Sample code

```
int res = await ITMGContext.GetInstance().GetAudioCtrl().GetSpeakerVolume();
```

Advanced APIs

Enabling in-ear monitoring

This API is used to enable in-ear monitoring. You need to call `EnableLoopBack+EnableSpeaker` before you can hear your own voice.

API prototype

```
Future<int> EnableLoopBack(bool enable)
```

Parameter	Type	Description
enable	bool	Specifies whether to enable.

Sample code


```
int res = await ITMGContext.GetInstance().GetAudioCtrl().EnableLoopBack(true);
```

Getting user's room audio type

This API is used to get a user's room audio type. The returned value is the room audio type. Value 0 indicates that an error occurred while getting the user's room audio type. For room audio types, see the `EnterRoom` API.

API prototype

```
Future<int> GetRoomType()
```

Sample code

```
int curType = await ITMGContext.GetInstance().GetRoom().GetRoomType();
```

Changing the room type

This API is used to modify a user's room audio type. For the result, see the callback event. The event type is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE`. The audio type of the room is determined by the first user to enter the room. After that, if a member in the room changes the room type, it will take effect for all members there.

API prototype

```
Future<int> ChangeRoomType(int roomType)
```

Parameter	Type	Description
roomtype	number	Room type to be switched to. For room audio types, see the `EnterRoom` API.

Sample code

```
int res = await ITMGContext.GetInstance().GetRoom().ChangeRoomType(1);
```

Callback event

After the room type is set, the event message `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE` will be returned in the callback. The returned parameters include `result`, `error_info`, and `new_room_type`. The `new_room_type` represents the following information. The event message will be identified in the `OnEvent` function.

Event Subtype	Parameter	Description

ITMG_ROOM_CHANGE_EVENT_ENTERROOM	1	Indicates that the existing audio type is inconsistent with and changed to that of the entered room.
ITMG_ROOM_CHANGE_EVENT_START	2	Indicates that a user is already in the room and the audio type starts changing (e.g., calling the `ChangeRoomType` API to change the audio type).
ITMG_ROOM_CHANGE_EVENT_COMPLETE	3	Indicates that a user is already in the room and the audio type has been changed.
ITMG_ROOM_CHANGE_EVENT_REQUEST	4	Indicates that a room member calls the `ChangeRoomType` API to request a change of room audio type.

Sample code

```
case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE:
  {
        // Process room type events
  }
  break;
```

The monitoring event of room call quality

This is the quality monitoring event used to listen on the network quality. If your network conditions are poor, the business layer will ask you to switch the network through the UI. This event is triggered once every two seconds after room entry, and its message is `ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY` . The returned parameters include `weight` , `loss` , and `delay` , which are as detailed below:

Parameter	Type	Description
weight	number	Value range: 1–50. `50` indicates excellent sound quality, `1` indicates very poor (barely usable) sound quality, and `0` represents an initial meaningless value. Generally, if the value is below 30, you can remind users that the network is poor and recommend them to switch the network.
loss	var	Upstream packet loss rate
delay	number	Voice chat delay in ms

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
Future<String> GetSDKVersion()
```

Sample code

```
_sdkVersions = await ITMGContext.GetInstance().GetSDKVersion();
```

Setting the application name and version

This API is used to set the application name and version.

API prototype

```
Future<void> SetAppVersion(String appVersion)
```

Parameter description

Parameter	Type	Description
appVersion	string	Application name and version

Sample code

```
await ITMGContext.GetInstance().SetAppVersion("gme V2.0.0");
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
Future<int> SetLogLevel(int levelWrite, int levelPrint)
```

Parameter description

Parameter	Type	Description
level	number	Sets the log level. `TMG_LOG_LEVEL_NONE` indicates not to log. Default value: `TMG_LOG_LEVEL_INFO`.

`level` description:

level	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs
TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContext.GetInstance().SetLogLevel(ITMG_LOG_LEVEL.TMG_LOG_LEVEL_ERROR,
ITMG_LOG_LEVEL.TMG_LOG_LEVEL_ERROR);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

API prototype

```
Future<int> SetLogPath(String logDir)
```

Parameter	Type	Description
logPath	string	Path

Sample code

```
String curPath = ""// Set a path by yourself
ITMGContext.GetInstance().SetLogPath(curPath);
```

Getting the diagnostic messages

This API is used to get information on the quality of real-time audio/video calls, which is mainly used to view real-time call quality and troubleshoot and can be ignored on the business side.

API prototype

```
Future<String> GetQualityTips()
```

Sample code

```
String curQualityTips = await
ITMGContext.GetInstance().GetRoom().GetQualityTips();
```

Callback message

Message	Description	Data
ITMG_MAIN_EVENT_TYPE_ENTER_ROOM	A member entered the audio room.	result; error
ITMG_MAIN_EVENT_TYPE_EXIT_ROOM	A member exited the audio room.	result; error
ITMG_MAIN_EVENT_TYPE_ROOM_DISCONNECT	The room was disconnected for network or other reasons.	result; error
ITMG_MAIN_EVNET_TYPE_USER_UPDATE	Room members were updated.	user_list; event_id
ITMG_MAIN_EVENT_TYPE_RECONNECT_START	The reconnection to the room started.	result; error
ITMG_MAIN_EVENT_TYPE_RECONNECT_SUCCESS	The reconnection to the room succeeded.	result; error
ITMG_MAIN_EVENT_TYPE_SWITCH_ROOM	The room was quickly switched.	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_TYPE	The room status was changed.	result; error_info;

		sub_event_ new_room_
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_START	Cross-room mic connect started.	result;
ITMG_MAIN_EVENT_TYPE_ROOM_SHARING_STOP	Cross-room mic connect stopped.	result;
ITMG_MAIN_EVENT_TYPE_SPEAKER_DEFAULT_DEVICE_CHANGED	The default speaker device was changed.	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_NEW_DEVICE	A new speaker device was added.	result; error
ITMG_MAIN_EVENT_TYPE_SPEAKER_LOST_DEVICE	A speaker device was lost.	result; error
ITMG_MAIN_EVENT_TYPE_MIC_NEW_DEVICE	A new mic device was added.	result; error
ITMG_MAIN_EVENT_TYPE_MIC_LOST_DEVICE	A mic device was lost.	result; error
ITMG_MAIN_EVENT_TYPE_MIC_DEFAULT_DEVICE_CHANGED	The default mic device was changed.	result; error
ITMG_MAIN_EVENT_TYPE_CHANGE_ROOM_QUALITY	The room quality changed.	weight; loss delay
ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE	Voice message recording was completed.	result; file_

ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE	Voice message upload was completed.	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE	Voice message download was completed.	result; file_path;file
ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE	Voice message playback was completed.	result; file_
ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE	Fast speech-to-text conversion was completed.	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE	Streaming speech-to-text conversion was completed.	result; file_ text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING	Streaming speech-to-text conversion is in progress.	result; file_ text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TEXT2SPEECH_COMPLETE	Text-to-speech conversion was completed.	result; text;file_id
ITMG_MAIN_EVNET_TYPE_PTT_TRANSLATE_TEXT_COMPLETE	Text translation was completed.	result; text;file_id

Speech-to-Text Service

Last updated：2024-01-18 15:15:48

This document describes how to integrate with and debug GME client APIs for the voice messaging and speech-to-text services for Flutter.

Key Considerations for Using GME

GME provides the real-time voice service and voice messaging and speech-to-text services, which all depend on core APIs such as `Init` and `Poll`.

Notes

You have created a GME application and obtained the SDK `AppID` and key. For more information, see Activating Services.

You have activated **GME real-time voice and voice messaging and speech-to-text services**. For more information, see Activating Services.

Configure your project before using GME; otherwise, the SDK will not take effect.

After a GME API is called successfully, `GmeError.AV_OK` will be returned with the value being `0`.

GME APIs should be called in the same thread.

The `Poll` API should be called periodically for GME to trigger event callbacks.

For detailed error codes, see Error Codes.

Note:

There is a default call rate limit for speech-to-text APIs. For more information on how calls are billed within the limit, see Purchase Guide. If you want to increase the limit or learn more about how excessive calls are billed, submit a ticket.

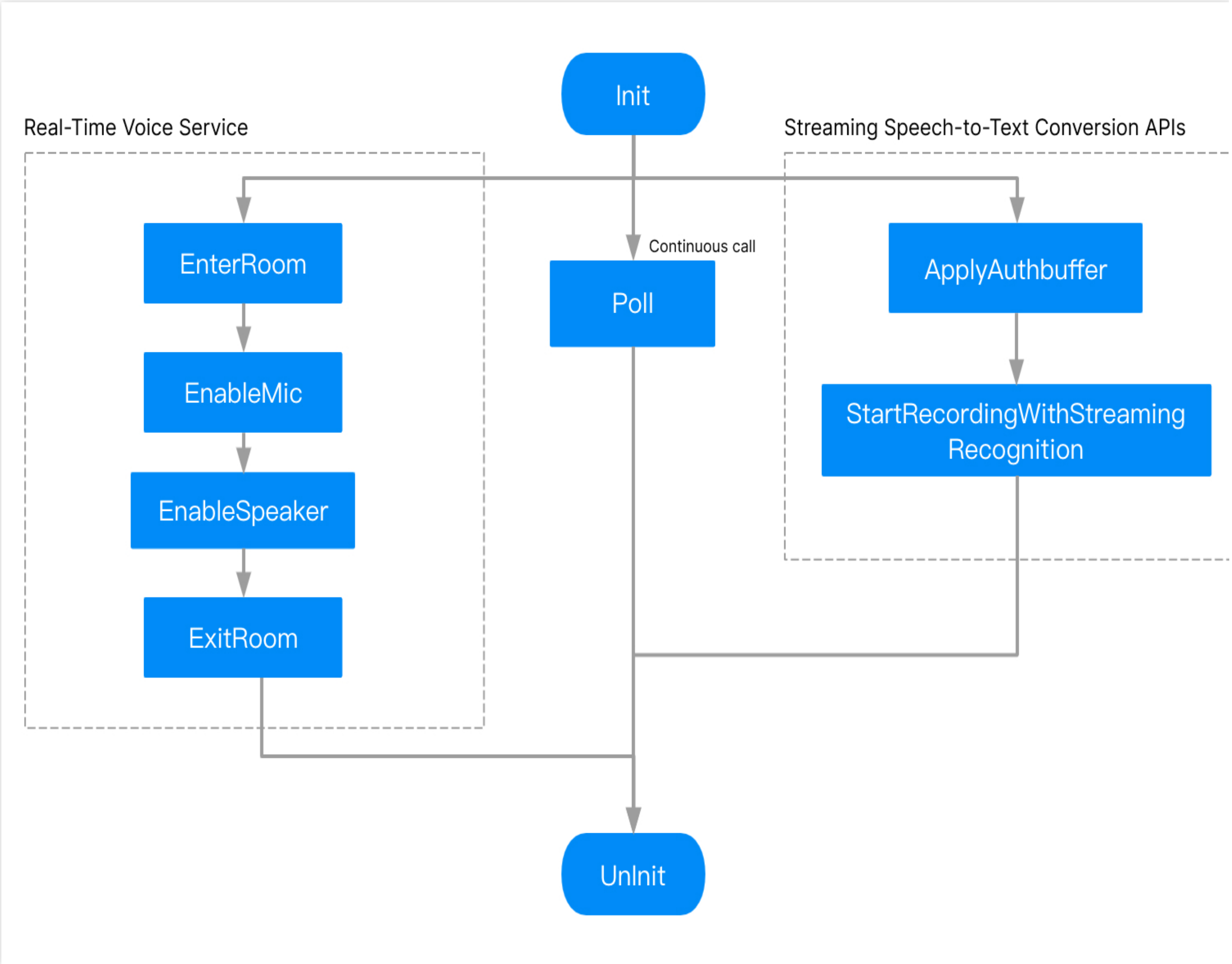
Non-streaming speech-to-text API ***SpeechToText()***: There can be up to 10 concurrent requests per account.

Streaming speech-to-text API ***StartRecordingWithStreamingRecognition()***: There can be up to 50 concurrent requests per account.

Integrating the SDK

Directions

Key processes involved in SDK integration are as follows:



1. Initializing GME
2. Call `Poll` periodically to trigger event callbacks.
3. Initialize authentication.
4. Start streaming speech recognition.
5. Stop recording.
6. Uninitialize GME.

.dart files

```
gme.dart       GME business implementation APIs
gmeType.dart   GME type definition file
gmeError.dart  GME error type definition file
```

Core APIs

API	Description
InitSDK	Initializes GME.
Poll	Triggers the event callback.
Uninit	Uninitializes GME.

Importing the GME module

```
import 'package:gme/gme.dart';
import 'package:gme/gmeType.dart';
```

Getting an instance

var m_context = await ITMGContext.GetInstance();

Initializing the SDK

You need to initialize the SDK through the `Init` API before you can use the real-time voice chat, voice messaging, 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

```
Future<int> InitSDK(String appID, String openID)
```

Parameter	Type	Description
sdkAppId	string	`AppID` provided in the GME console, which can be obtained as instructed in Activating Services.
openID	string	`openID` can only be in `Int64` 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.

Returned values

Returned Value	Description
GmeError.AV_OK= 0	SDK initialized successfully.
AV_ERR_SDK_NOT_FULL_UPDATE=7015	Solution: Check whether the SDK file is complete. We recommend that you delete it and then import the SDK again.

Note:

Notes on 7015 error code

The 7015 error code is identified by MD5. If this error is reported during integration, check the integrity and version of the SDK file as prompted.

The returned value `AV_ERR_SDK_NOT_FULL_UPDATE` is **only a reminder** but will not cause an initialization failure.

Sample code

```
int res = await
ITMGContext.GetInstance().InitSDK(_editAppID.text,_editOpenID.text);
// Determine whether the initialization is successful by the returned value
if (ret != GmeError.AV_OK)
{
    print("Failed to initialize the SDK:");
    return;
}
```

Triggering event callback

Event callbacks can be triggered by calling the `Poll` API in the timer. 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.

Note:

The `Poll` API must be called periodically and in the main thread to avoid abnormal API callbacks.

API prototype

```
Future<void> Poll()
```

Sample code

```
Future<void> pollTimer() async {
  _pollTimer = Timer.periodic(Duration(milliseconds: 100), (Timer timer) {
  ITMGContext.GetInstance().Poll();
  });
}
```

Uninitializing SDK

This API is used to uninitialize the SDK to make it uninitialized. **If the game business account is bound to `openid`, switching game account requires uninitializing GME and then using the new `openid` to**

initialize again.

API prototype

```
Future<int> Uninit()
```

Voice Messaging and Speech-to-Text Services

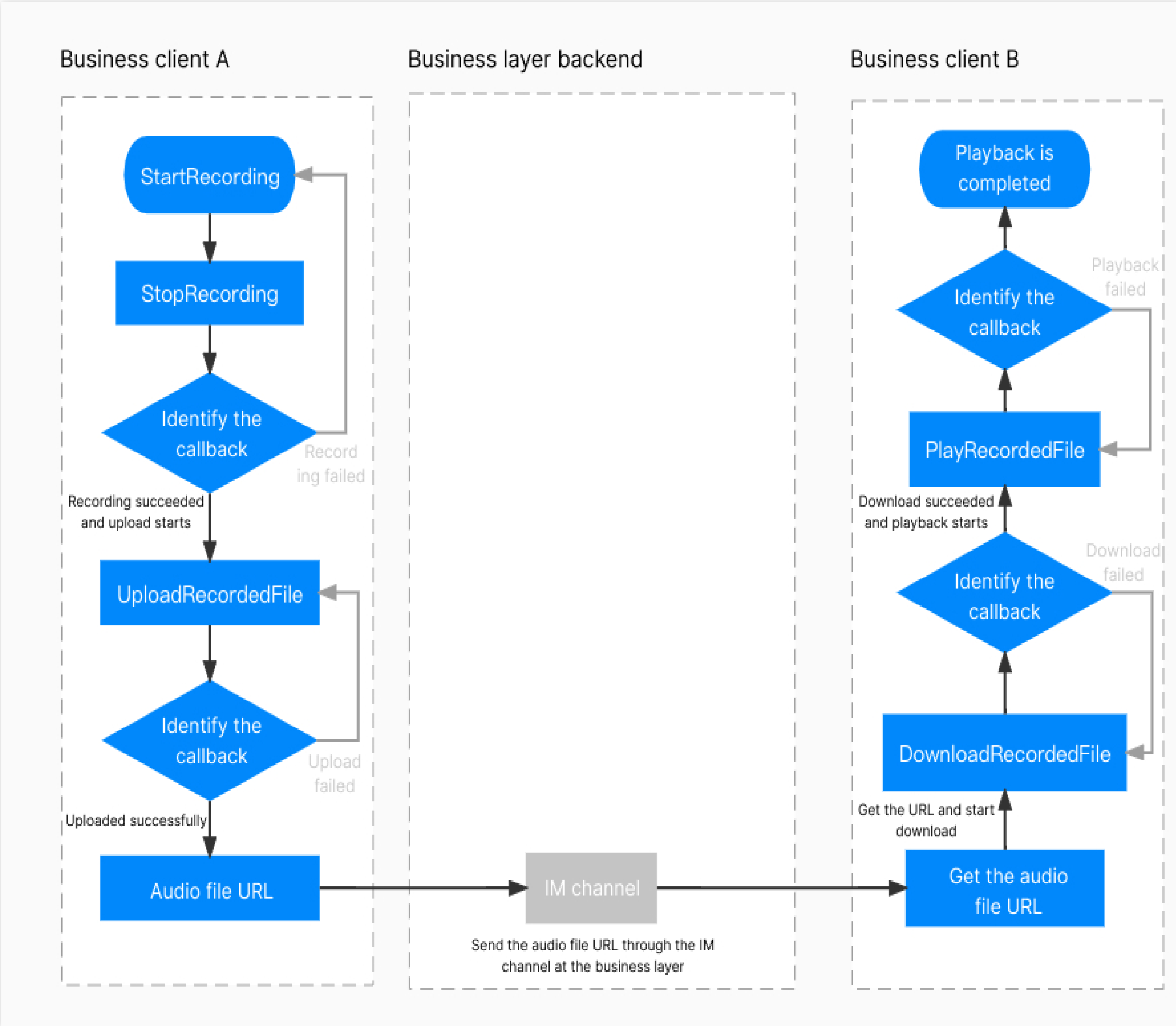
Note:

The speech-to-text service consists of fast recording-to-text conversion and streaming speech-to-text conversion.

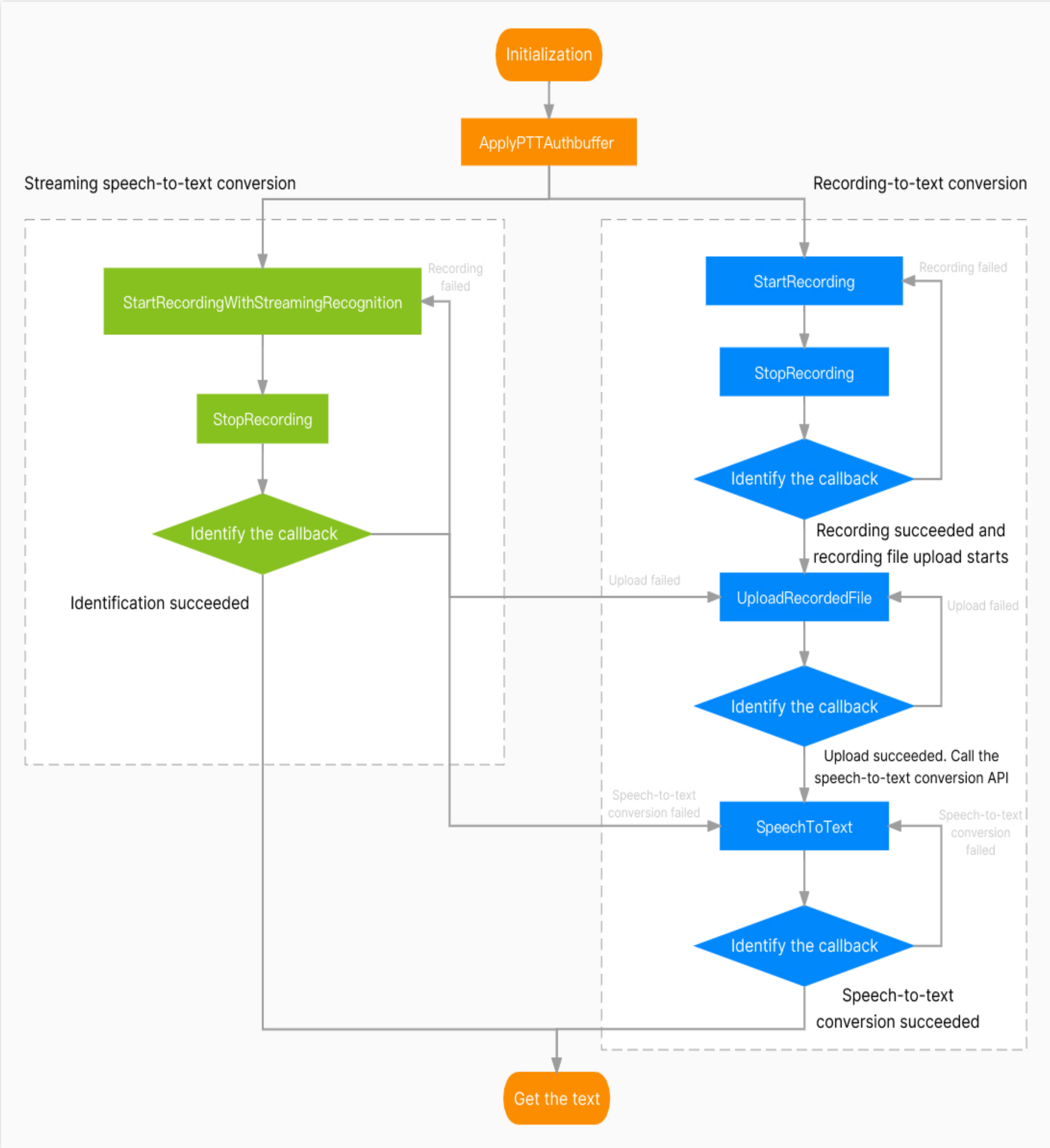
You do not need to enter a voice chat room when using the voice messaging service.

The maximum recording duration of a voice message is 58 seconds by default, and the minimum recording duration cannot be less than 1 second. If you want to customize the recording duration, for example, to change the maximum recording duration to 10 seconds, call the `SetMaxMessageLength` API to set it after initialization.

Flowchart for using the voice message service



Flowchart for using the speech-to-text service



API	Description
GenAuthBuffer	Gets the authentication information.
SetMaxMessageLength	Specifies the maximum duration of a voice message.

Generating the local authentication key

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

API prototype

```
Future<Uint8List> GenAuthBuffer(String appID, String roomID, String openID,
String key)
```

Parameter	Type	Description
appId	string	`AppId` from the Tencent Cloud console
roomId	string	Enter `null` or an empty string.
openId	string	User ID, which is the same as `OpenId` during initialization.
key	string	Permission key from the Tencent Cloud console.

Application authentication

After the authentication information is generated, the authentication is assigned to the SDK.

API prototype

```
Future<int> ApplyPTTAuthbuffer(Uint8List authBuffer)
```

Sample code

```
Uint8List authBuffer = await
ITMGContext.GetInstance().GenAuthBuffer(_editAppID.text, _editRoomID.text,
_editOpenID.text, _editKey.text);
m_context.ApplyPTTAuthbuffer(authBuffer);
```

Specifying the maximum duration of voice message

This API is used to specify the maximum duration of a voice message, which can be up to 58 seconds.

API prototype

```
Future<int> SetMaxMessageLength(int msTime)
```

Parameter	Type	Description
msTime	number	Voice message duration in ms. Value range: 1000 < `msTime` <= 58000

Sample code

```
ITMGContext.GetInstance().GetPTT().SetMaxMessageLength(fileLen);
```

Streaming Speech Recognition

Voice messaging and speech-to-text APIs

API	Description
StartRecordingWithStreamingRecognition	Starts streaming recording.
StopRecording	Stops recording.

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 text recognized in speech into a specified language and return the translation. **To stop recording, call** StopRecording**.**

API prototype

```
Future<int> StartRecordingWithStreamingRecognition(String filePath, String
speechLanguage, String translateLanguage)
```

Parameter	Type	Description
filePath	string	Path of the stored audio file
speechLanguage	string	The language in which the voice message file is to be converted to text. For parameters, see Language Parameter Reference List.
translateLanguage	string	The language in which the voice message file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

```
string filePath = "xx/xxx/xxx.silk"
int res = await
ITMGContext.GetInstance().GetPTT().StartRecordingWithStreamingRecognition(fileP
ath, strCurLanguage, strCurLanguage);
```

```
if (ret == 0) {
    this.currentStatus = "Start streaming recording";
} else {
    this.currentStatus = "Failed to start streaming recording";
}
```

Note:

Translation incurs additional fees. For more information, see Purchase Guide.

Callback for streaming speech recognition

After streaming speech recognition is started, you need to listen on callback messages in the `OnEvent` notification, which is as detailed below:

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE` 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.

`ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING` 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 callback notification based on the actual needs. The passed parameters include the following four messages.

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

Note:

The `file_id` is empty when the 'ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING' message is listened for.

Error codes

Error Code	Description	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 conversion failed, but recording	The message returned contains a backend URL after successful upload. Call the `SpeechToText` API to

	and upload succeeded.	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.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

If the error code 4098 is reported, see Speech-to-text Conversion for solutions.

Sample code

```
void handleEventMsg(int eventType, String data){
   switch (eventType) {
        case
ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_COMPLETE:
        {
            HandleSTREAM2TEXTComplete(data,true);
            break;
        }
        ...
        case
ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_STREAMINGRECOGNITION_IS_RUNNING:
        {
            HandleSTREAM2TEXTComplete(data, false);
            break;
        }
    }
}
ITMGContext.GetInstance().SetEvent(handleEventMsg);
```

Voice Message Recording

The recording process is as follows: start recording > stop recording > return recording callback > start the next recording.

Voice messaging and speech-to-text APIs

API	Description
StartRecording	Starts recording.
PauseRecording	Pauses recording.
ResumeRecording	Resumes recording.
StopRecording	Stops recording.
CancelRecording	Cancels recording.

Starting recording

This API is used to start recording.

API prototype

```
Future<int> StartRecording(String filePath)
```

Parameter	Type	Description
filePath	string	Path of the stored voice message file

Sample code

```
string filepath = "xxxx/xxx.silk";
int res = await ITMGContext.GetInstance().GetPTT().StartRecording(filepath);
```

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

```
Future<int> StopRecording()
```

Sample code

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

Callback for recording start

A callback will be executed through a delegate function to pass a message when recording is completed.

To stop recording, call `StopRecording` . The callback for recording start will be returned after the recording is stopped.

Parameter	Type	Description
code	string	`0`: Recording is completed.
filepath	string	Path of the stored recording file, which must be accessible and cannot be the `fileid`.

Error codes

Error Code	Cause	Solution
4097	A parameter is empty.	Check whether the API parameters in the code are correct.
4098	An initialization error occurred.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.
4099	Recording is in progress.	Make sure that the SDK recording feature is used at the right time.
4100	No audio data is captured.	Check whether the mic is working properly.
4101	An error occurred while accessing the file during recording.	Ensure the existence of the file and the validity of the file path.
4102	The mic is not authorized.	The mic permission is required for using the SDK. To add the permission, see the SDK project configuration document for the corresponding engine or platform.
4103	The recording duration is too short.	The recording duration should be in ms and longer than 1,000 ms.
4104	No recording operation is started.	Check whether the recording starting API has been called.

Sample code

```
void handleEventMsg(int eventType, String data){
```

```
    switch (eventType) {
        case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
        {
          // Process
          break;
        }
        ...
        case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_RECORD_COMPLETE:
        {
          // Process
          break;
        }
     }
}
ITMGContext.GetInstance().SetEvent(handleEventMsg);
```

Pausing recording

This API is used to pause recording. If you want to resume recording, call the `ResumeRecording` API.

API prototype

```
Future<int> PauseRecording()
```

Sample code

```
ITMGContext.GetInstance().GetPTT().PauseRecording();
```

Resuming recording

This API is used to resume recording.

API prototype

```
Future<int> ResumeRecording()
```

Sample code

```
ITMGContext.GetInstance().GetPTT().ResumeRecording();
```

Canceling recording

This API is used to cancel recording. **There is no callback after cancellation**.

API prototype

```
Future<int> CancelRecording()
```

Sample code

```
ITMGContext.GetInstance().GetPTT().CancelRecording();
```

Voice Message Upload, Download, and Playback

API	Description
UploadRecordedFile	Uploads an audio file.
DownloadRecordedFile	Downloads an audio file.
PlayRecordedFile	Plays back audio.
StopPlayFile	Stops playing back audio.
GetFileSize	Gets the audio file size.
GetVoiceFileDuration	Gets the audio file duration.

Uploading an audio file

This API is used to upload an audio file.

API prototype

```
Future<int> UploadRecordedFile(String filePath)
```

Parameter	Type	Description
filePath	String	Path of the uploaded audio file, which is a local path.

Sample code

```
ITMGContext.GetInstance().GetPTT().UploadRecordedFile(_filePath);
```

Callback for audio file upload completion

After the audio file is uploaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result` , `file_path` , and `file_id` .

Parameter	Type	Description
result	number	`0`: Recording is completed.
filepath	string	Path of the stored recording file
fileid	string	File URL

Error codes

Error Code	Cause	Solution
8193	An error occurred while accessing the file during upload.	Ensure the existence of the file and the validity of the file path.
8194	Signature verification failed.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
8195	A network error occurred.	Check whether the device can access the internet.
8196	The network failed while getting the upload parameters.	Check whether the authentication is correct and whether the device network can normally access the internet.
8197	The packet returned during the process of getting the upload parameters is empty.	Check whether the authentication is correct and whether the device network can normally access the internet.
8198	Failed to decode the packet returned during the process of getting the upload parameters.	Check whether the authentication is correct and whether the device network can normally access the internet.
8200	No `appinfo` is set.	Check whether the `apply` API is called or whether the input parameters are empty.

Sample code

```
void handleEventMsg(int eventType, String data){
 switch (eventType) {
```

```
        ...
        case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_UPLOAD_COMPLETE:
        {
            // Process
            break;
        }
    }
}
ITMGContext.GetInstance().SetEvent(handleEventMsg);
```

Downloading the audio file

This API is used to download an audio file.

API prototype

```
Future<int> DownloadRecordedFile(String fileId, String filePath)
```

Parameter	Type	Description
fileId	string	File URL
filePath	string	Local path of the saved file, which must be accessible and cannot be the `fileid`.

Sample code

```
ITMGContext.GetInstance().GetPTT().DownloadRecordedFile(_fileId, _filePath);
```

Callback for audio file download completion

After the audio file is downloaded, the event message `ITMG_MAIN_EVNET_TYPE_PTT_DOWNLOAD_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameters include `result`, `file_path`, and `file_id`.

Parameter	Type	Description
result	number	`0`: Download is completed.
filepath	string	Path of the stored recording file
fileid	string	URL of the recording file, which will be retained on the server for 90 days.

Error codes

Error Code	Cause	Solution
12289	An error occurred while accessing the file during download.	Check whether the file path is valid.
12290	Signature verification failed.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.
12291	A network storage system exception occurred.	The server failed to get the audio file. Check whether the API parameter `fileid` is correct, whether the network is normal, and whether the file exists in COS.
12292	A server file system error occurred.	Check whether the device can access the internet and whether the file exists on the server.
12293	The HTTP network failed while getting the download parameters.	Check whether the device can access the internet.
12294	The packet returned during the process of getting the download parameters is empty.	Check whether the device can access the internet.
12295	Failed to decode the packet returned during the process of getting the download parameters.	Check whether the device can access the internet.
12297	No `appinfo` is set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.

Sample code

```
void handleEventMsg(int eventType, String data){
    switch (eventType) {
        ...
        case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
        {
          // Process
          break;
        }
    }
}
ITMGContext.GetInstance().SetEvent(handleEventMsg);
```

Playing back audio

This API is used to play back audio.

API prototype

```
Future<int> PlayRecordedFile(String filePath, int voiceType)
```

Parameter	Type	Description
filePath	string	Local audio file path
voicetype	ITMG_VOICE_TYPE	Voice changing type. For more information, see Voice Changing.

Error codes

Error Code	Cause	Solution
20485	Playback is not started.	Ensure the existence of the file and the validity of the file path.

Sample code

```
int res = await ITMGContext.GetInstance().GetPTT().PlayRecordedFile(_filePath,
_nVoiceType);
```

Callback for audio playback

After the audio is played back, the event message `ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE` will be returned, which will be identified in the `OnEvent` function.
The passed parameter includes `result` and `file_path`.

Parameter	Type	Description
code	number	`0`: Playback is completed.
filepath	string	Path of the stored recording file

Error codes

Error Code	Cause	Solution
20481	An initialization error occurred.	Check whether the device is being used, whether the permissions are normal, and whether the initialization is normal.

20482	During playback, the client tried to interrupt and play back the next one but failed (which should succeed normally).	Check whether the code logic is correct.
20483	A parameter is empty.	Check whether the API parameters in the code are correct.
20484	An internal error occurred.	An error occurred while initializing the player. This error code is generally caused by failure in decoding, and the error should be located with the aid of logs.

Sample code

```
void handleEventMsg(int eventType, String data){
 switch (eventType) {
    case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_PLAY_COMPLETE:
    {
        // Process
        break;
    }
   }
}
```

Stopping audio playback

This API is used to stop audio playback. There will be a callback for playback completion when the playback stops.

API prototype

```
Future<int> StopPlayFile()
```

Sample code

```
ITMGContext.GetInstance().GetPTT().StopPlayFile();
```

Getting audio file size

This API is used to get the size of an audio file.

API prototype

```
Future<int> GetFileSize(String filePath)
```

Parameter	Type	Description
filePath	string	Path of the audio file, which is a local path

Sample code

```
final int res = await
ITMGContext.GetInstance().GetPTT().GetFileSize(_filePath);
```

Getting audio file duration

This API is used to get the duration of an audio file in milliseconds.

API prototype

```
Future<int> GetVoiceFileDuration(String filePath)
```

Parameter	Type	Description
filePath	string	Path of the audio file, which is a local path

Sample code

```
final int res = await
ITMGContext.GetInstance().GetPTT().GetVoiceFileDuration(_filePath);
```

Fast Recording-to-Text Conversion

Translating audio file into text in specified language

This API can specify a language for recognition or translate the text recognized in speech into a specified language and return the translation.

Note：

Translation incurs additional fees. For more information, see Purchase Guide.

API prototype

```
Future<int> SpeechToText(String fileId, String speechLanguage, String
translateLanguage)
```

Parameter	Type	Description

fileID	string	URL of the audio file, which will be retained on the server for 90 days.
speechLanguage	string	The language in which the audio file is to be converted to text. For parameters, see Language Parameter Reference List.
translatelanguage	string	The language in which the audio file is to be translated to text. For parameters, see Language Parameter Reference List.

Sample code

```
ITMGContext.GetInstance().GetPTT().SpeechToText(_fileId, "cmn-Hans-CN", "cmn-Hans-CN");
```

Callback for recognition

After the specified audio file is converted to text, the event message ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE will be returned, which will be identified in the `OnEvent` function.

The passed parameters include `result`, `file_path` and `text` (recognized text).

Parameter	Type	Description
result	number	\`0\`: Recording is completed.
fileid	string	URL of the audio file, which will be retained on the server for 90 days.
text	string	Converted text

Error codes

Error Code	Cause	Solution
32769	An internal error occurred.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32770	A network error occurred.	Check whether the device can access the internet.
32772	Failed to decode the returned packet.	Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32774	No \`appinfo\` is set.	Check whether the authentication key is correct and whether the voice messaging and speech-to-text feature is initialized.

32776	`authbuffer` check failed.	Check whether `authbuffer` is correct.
32784	The speech-to-text conversion parameter is incorrect.	Check whether the API parameter `fileid` in the code is empty.
32785	Speech-to-text translation returned an error.	An error occurred in the voice messaging and speech-to-text feature on the backend. Analyze logs, get the actual error code returned from the backend to the client, and ask backend personnel for assistance.
32787	Speech-to-text conversion succeeded, but the text translation service was not activated.	Activate the text translation service in the console.
32788	Speech-to-text conversion succeeded, but the language parameter of the text translation service was invalid.	Check the parameter passed in.

Sample code

```
void handleEventMsg(int eventType, String data){
 switch (eventType) {
    case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVENT_TYPE_ENTER_ROOM:
        {
          // Process
          break;
        }
        ...
        case ITMG_MAIN_EVENT_TYPE.ITMG_MAIN_EVNET_TYPE_PTT_SPEECH2TEXT_COMPLETE:
        {
          // Process
          break;
        }
    }
}
```

Voice Message Volume Level APIs

API	Description
GetMicLevel	Gets the real-time mic volume level.
SetMicVolume	Sets the recording volume level.

GetMicVolume	Gets the recording volume level.
GetSpeakerLevel	Gets the real-time speaker volume level.
SetSpeakerVolume	Sets the playback volume level.
GetSpeakerVolume	Gets the playback volume level.

Getting the real-time mic volume of voice message

This API is used to get the real-time mic volume. A number-type value will be returned. Value range: 0-200.

API prototype

```
Future<int> GetMicLevel()
```

Sample code

```
final int res = await ITMGContext.GetInstance().GetPTT().GetMicLevel();
```

Setting the recording volume of voice message

This API is used to set the recording volume of voice message. Value range: 0-200.

API prototype

```
Future<int> SetMicVolume(int volume)
```

Parameter	Type	Description
vol	number	Value range: 0-200. Default value: `100`. `0` indicates that the audio is mute, while `100` indicates that the volume level remains unchanged.

Sample code

```
final int res = await ITMGContext.GetInstance().GetPTT().SetMicVolume(100);
```

Getting the recording volume of voice message

This API is used to get the recording volume of a voice message. A number-type value will be returned. Value range: 0-200.

API prototype

```
Future<int> GetMicVolume()
```

Sample code

```
final int res = await ITMGContext.GetInstance().GetPTT().GetMicVolume();
```

Getting the real-time speaker volume of voice message

This API is used to get the real-time speaker volume. A number-type value will be returned. Value range: 0-200.

API prototype

```
Future<int> GetSpeakerLevel()
```

Sample code

```
final int res = await ITMGContext.GetInstance().GetPTT().GetSpeakerLevel();
```

Setting the playback volume of voice message

This API is used to set the playback volume of voice message. Value range: 0-200.

API prototype

```
Future<int> SetSpeakerVolume(int volume)
```

Sample code

```
final int res = await ITMGContext.GetInstance().GetPTT().SetSpeakerVolume(100);
```

Getting the playback volume of voice message

This API is used to get the playback volume of a voice message. A number-type value will be returned. Value range: 0-200.

API prototype

```
Future<int> GetSpeakerVolume()
```

Sample code

```
final int res = await ITMGContext.GetInstance().GetPTT().GetSpeakerVolume();
```

Advanced APIs

Getting version number

This API is used to get the SDK version number for analysis.

API prototype

```
Future<String> GetSDKVersion()
```

Sample code

```
_sdkVersions = await ITMGContext.GetInstance().GetSDKVersion();
```

Setting log printing level

This API is used to set the level of logs to be printed, and needs to be called before the initialization. It is recommended to keep the default level.

API prototype

```
Future<int> SetLogLevel(int levelWrite, int levelPrint)
```

Parameter description

Parameter	Type	Description
level	ITMG_LOG_LEVEL	Sets the log level. `TMG_LOG_LEVEL_NONE` indicates not to log. Default value: `TMG_LOG_LEVEL_INFO`.

`level` description:

level	Description
TMG_LOG_LEVEL_NONE	Does not print logs
TMG_LOG_LEVEL_ERROR	Prints error logs (default)
TMG_LOG_LEVEL_INFO	Prints info logs

TMG_LOG_LEVEL_DEBUG	Prints debug logs
TMG_LOG_LEVEL_VERBOSE	Prints verbose logs

Sample code

```
ITMGContext.GetInstance().SetLogLevel(ITMG_LOG_LEVEL.TMG_LOG_LEVEL_ERROR,ITMG_L
OG_LEVEL.TMG_LOG_LEVEL_ERROR);
```

Setting the log printing path

This API is used to set the log printing path. The default path is as follows. It needs to be called before Init.

API prototype

```
Future<int> SetLogPath(String logDir)
```

Parameter	Type	Description
logPath	string	Path

Sample code

```
String logDir = ""// Set a path by yourself
ITMGContext.GetInstance().SetLogPath(curPath);
```

SDK Version Upgrade Guide

Last updated：2024-01-18 15:15:48

This document describes the upgrade of GME.

Upgrade from GME 2.x to 2.9

SDK updates

Dynamic library split

Rename Android package

The GME SDK is updated with the following new library files in addition to libgmesdk.

Library files' corresponding features

The new version of GME splits the dynamic libraries to reduce the package size. You can only import the library files you need. For example, if you only need the voice changing feature, to import `libgme_soundtouch` is good.

Library File	Feature
libgmefdkaac	1. Used to enter an SD or HD voice room. 2. Used to play back accompaniment files in ACC format
libgmefaad2	Used to play back accompaniment files in MP4 format
libgmeogg	Used to play back accompaniment files in OGG format
libgmelamemp3	Used to play back accompaniment files in MP3 format
libgmesoundtouch	Used for voice changing and pitch changing

Upgrade Notice

For iOS upgrade, please see iOS Project Upgrade guide.

For Android upgrade, you need to rename package(change Tencent into GME) and modify obfuscation configuration. Please see Project Export.

For Unity upgrade, if you used SD or HD sound quality, or accompaniment, please see Using HD Sound Quality.

Upgrade from GME 2.2 to 2.3.5

SDK updates

New features

Offline voice can be used during voice chat now.

Voice chat can filter offensive, insecure, or inappropriate information.

HTML5-based voice chat is supported now, making voice chat available across all operating systems.

Android v8a architecture is supported now.

Low-latency capture and playback is adaptive to Android now.

Optimizations

Optimized the range voice APIs of the SDK to lower the access threshold.

Optimized noise reduction for voice.

Greatly reduced memory usage by the SDK.

Changes in Major APIs

EnterRoom

The room entering operation has been changed from sync to async. If the return value is 0, the async delivery is successful and waiting to be processed by the callback function; otherwise, the async delivery fails.

```
public abstract int EnterRoom();
```

ExitRoom

The room exiting operation has been changed from sync to async. It is handled in the same way as the RoomExitComplete callback function. If the return value is AV_OK, the async delivery is successful.

Note:

If there is a scenario in the application where room entry is performed immediately after room exit, you don't need to wait for the `RoomExitComplete` callback notification from the `ExitRoom` API during API call; instead, you can directly call the API.

```
public abstract int ExitRoom();
```

Changes in Error Codes

For uniform processing of all error codes, use !AV_OK.

To handle the errors separately, focus on the type of error returned by the API.

Note:

Error code "1" has no specific meaning and will no longer be returned since v2.3.5, so it has been deleted.

Changes in Other APIs

PauseAudio/ResumeAudio

```
public int PauseAudio()
public int ResumeAudio()
```

If the `ITMGAudioCtrl::PauseAudio` or `ResumeAudio` API is called in an SDK before v2.3, please see the table below for version comparison.

Before v2.3	v2.3
For mutual exclusivity with other modules	Change PauseAudio to Pause and change ResumeAudio to Resume
For using offline voice in voice chat	Delete PauseAudio and ResumeAudio

Changes in the Parameters of the SetLogLevel API

Original API

```
ITMGContext virtual void SetLogLevel(int logLevel, bool enableWrite, bool
enablePrint)
```

New API

```
ITMGContext virtual void SetLogLevel(ITMG_LOG_LEVEL levelWrite, ITMG_LOG_LEVEL
levelPrint)
```

Parameter description

Parameter	Type	Description
levelWrite	ITMG_LOG_LEVEL	Sets the level of logs to be written, TMG_LOG_LEVEL_NONE means not to write
levelPrint	ITMG_LOG_LEVEL	Sets the level of logs to be printed, TMG_LOG_LEVEL_NONE means not to print

ITMG_LOG_LEVEL Type

ITMG_LOG_LEVEL	Description
TMG_LOG_LEVEL_NONE=0	Do not print logs
TMG_LOG_LEVEL_ERROR=1	Prints error logs (default)
TMG_LOG_LEVEL_INFO=2	Prints prompt logs
TMG_LOG_LEVEL_DEBUG=3	Prints development and debugging logs
TMG_LOG_LEVEL_VERBOSE=4	Prints high-frequency logs

Upgrade from GME 2.3.5 to 2.5.1

New APIs

GetSendStreamLevel

This API is used to get the real-time audio upstreaming volume level. An int-type value will be returned. Value range: 0–100.

```
ITMGContextGetInstance()->GetAudioCtrl()->GetSendStreamLevel();
```

GetRecvStreamLevel

This API is used to get the real-time audio downstreaming volume levels of other members in the room. An int-type value will be returned. Value range: 0–100.

```
iter->second.level = ITMGContextGetInstance()->GetAudioCtrl()-
>GetRecvStreamLevel(iter->second.openid.c_str());
```

API changes

Type change for returned values of voice messaging and speech-to-text APIs

The type of returned values of the following APIs has been changed to `int` .

```
StartRecording
UploadRecordedFile
DownloadRecordedFile
PlayRecordedFile
SpeechToText
```

Upgrade from GME 2.5 to 2.7

New APIs

PlayRecordedFile(const char* filePath, ITMG_VOICE_TYPE voiceType)

This API is used to playback voice message with voice changing effects.

SetAccompanyKey(int nKey)

This API is used to set the voice chat accompaniment up and down.

Error Codes

Last updated：2024-12-05 15:40:31

Feature Description

If there is an Error field in the response, it means that the API call failed. For example:

```
{
    "Response": {
        "Error": {
            "Code": "AuthFailure.SignatureFailure",
            "Message": "The provided credentials could not be validated. Please
check your signature is correct."
        },
        "RequestId": "ed93f3cb-f35e-473f-b9f3-0d451b8b79c6"
    }
}
```

Code in Error indicates the error code, and Message indicates the specific information of the error.

Error Code List

Common Error Codes

Error Code	Description
ActionOffline	This API has been deprecated.
AuthFailure.InvalidAuthorization	`Authorization` in the request header is invalid.
AuthFailure.InvalidSecretId	Invalid key (not a TencentCloud API key type).
AuthFailure.MFAFailure	MFA failed.
AuthFailure.SecretIdNotFound	Key does not exist. Check if the key has been deleted or disabled in the console, and if not, check if the key is correctly entered. Note that whitespaces should not exist before or after the key.
AuthFailure.SignatureExpire	Signature expired. Timestamp and server time cannot differ by more than five minutes. Please

	ensure your current local time matches the standard time.
AuthFailure.SignatureFailure	Invalid signature. Signature calculation error. Please ensure you've followed the signature calculation process described in the Signature API documentation.
AuthFailure.TokenFailure	Token error.
AuthFailure.UnauthorizedOperation	The request is not authorized. For more information, see the CAM documentation.
DryRunOperation	DryRun Operation. It means that the request would have succeeded, but the DryRun parameter was used.
FailedOperation	Operation failed.
InternalError	Internal error.
InvalidAction	The API does not exist.
InvalidParameter	Incorrect parameter.
InvalidParameterValue	Invalid parameter value.
InvalidRequest	The multipart format of the request body is incorrect.
IpInBlacklist	Your IP is in uin IP blacklist.
IpNotInWhitelist	Your IP is not in uin IP whitelist.
LimitExceeded	Quota limit exceeded.
MissingParameter	A parameter is missing.
NoSuchProduct	The product does not exist.
NoSuchVersion	The API version does not exist.
RequestLimitExceeded	The number of requests exceeds the frequency limit.
RequestLimitExceeded.GlobalRegionUinLimitExceeded	Uin exceeds the frequency limit.
RequestLimitExceeded.IPLimitExceeded	The number of ip requests exceeds the frequency limit.
RequestLimitExceeded.UinLimitExceeded	The number of uin requests exceeds the frequency

	limit.
RequestSizeLimitExceeded	The request size exceeds the upper limit.
ResourceInUse	Resource is in use.
ResourceInsufficient	Insufficient resource.
ResourceNotFound	The resource does not exist.
ResourceUnavailable	Resource is unavailable.
ResponseSizeLimitExceeded	The response size exceeds the upper limit.
ServiceUnavailable	Service is unavailable now.
UnauthorizedOperation	Unauthorized operation.
UnknownParameter	Unknown parameter.
UnsupportedOperation	Unsupported operation.
UnsupportedProtocol	HTTP(S) request protocol error; only GET and POST requests are supported.
UnsupportedRegion	API does not support the requested region.

Service Error Codes

Error Code	Description
FailedOperation.UserFeeNegative	Operation not allowed as your account is in arrears.
InvalidParameter.DateInvalid	Invalid date.
InvalidParameter.DateOutOfSixtyDays	The entered query date range is longer than 60 days.
InvalidParameter.TagKey	Incorrect tag.
InvalidParameter.TimeRangeError	Incorrect query time range.
InvalidParameterValue.InvalidBizId	Invalid BizId.(SDKAppid)
InvalidParameterValue.InvalidRecordMode	Invalid RecordMode.
InvalidParameterValue.InvalidRoomId	Invalid RoomId.
InvalidParameterValue.InvalidSubscribeRecordUserIds	Incorrect blocklist/allowlist format.

InvalidParameterValue.InvalidSubscribeUserIds	The number of entries on the allowlist exceeds 20.
InvalidParameterValue.InvalidTaskId	Invalid taskid.
InvalidParameterValue.InvalidUNSubscribeUserIds	The number of entries on the blocklist exceeds 20.
LimitExceeded.Application	The number of created applications has reached the upper limit.
OperationDenied	Operation denied.
ResourceInUse.TaskInUse	The task already exists.
ResourceNotFound.BizidIsNotFound	Incorrect application ID.
ResourceNotFound.RoomNotFound	The room does not exist.
ResourceNotFound.TaskNotFound	The task ID does not exist.
UnauthorizedOperation.CreateAppDenied	Application creation is not authorized.
UnauthorizedOperation.UnRealNameAuth	Unverified user.
UnsupportedOperation.ServiceNotOpened	The recording service is not activated.

Toolchain

Last updated：2024-01-18 15:15:48

This document lists the toolchain for GME integration.

V2.9.1

Platform	Description
Win32	Windows Software Development Kit (Visual Studio 2015): 10.0.14393.0 Platform Toolset: v140_xp
macOS	MacOS Minimum Deployment Target: 10.10 macOS SDK: 12.0 Xcode®: 13.1 Target Architectures: x86_64
iOS	iOS Minimum Deployment Target: 9.0 iOS SDK: 15.0 Xcode®: 13.1 Target Architectures: arm64, armv7, arm64(simulator), x86_64(simulator)
Android	Android SDK: Minimum API 16 Android SDK Tools: 30.0.3 Android NDK: r23
PS4	SDK: SDK 6.000
PS5	SDK: SDK 3.000
Switch	SDK: DevEnv8.3.0
XboxOne	Microsoft® XDK: 10.0.17134.5055 (July 2018 qfe7)