目录

[1 RTChatSDK接入说明 3](#_Toc396996424)

[1.1 IOS应用项目中引入开发接口类 3](#_Toc396996425)

[1.2 设置回调代理类及初始化SDK 3](#_Toc396996426)

[1.3 实现SDK中OpenAPI的接口回调函数 4](#_Toc396996427)

[1.4 调用OpenAPI 7](#_Toc396996428)

[1.4.1 初始化SDK 7](#_Toc396996429)

[1.4.2 请求登陆 7](#_Toc396996430)

[1.4.3 请求房间列表 7](#_Toc396996431)

[1.4.4 创建房间 7](#_Toc396996432)

[1.4.5 申请加入房间 8](#_Toc396996433)

[1.4.6 退出房间 8](#_Toc396996434)

[1.4.7 最小化时清理数据 8](#_Toc396996435)

[1.4.8 激活SDK 8](#_Toc396996436)

[1.4.9 加入麦序及离开麦序 8](#_Toc396996437)

[1.4.10 设置是否接受随机聊天 8](#_Toc396996438)

[1.4.11 设置本人静音 9](#_Toc396996439)

# 1 RTChatSDK接入说明

在cocos2dx中使用RTChatSDK\_V1.0时，主要有以下六个步骤：

**Step1：**IOS应用项目中引入开发接口类；

**Step2**：设置回调代理类及初始化SDK;

**Step4**：调用OpenAPI。

**Step5**：在AndroidManifest中增加SDK需要的配置；

**Step6**：调用创建SDK的主要实现类的实例；

## 1.1 IOS应用项目中引入开发接口类

在Cocos2dx项目中添加如下类：

RTChat/RTChaCommonTypes.h //主要定义了数据结构及枚举类型

RTChat/RTChatSdk.h //定义了RTChatSDK底层C++接口函数

RTChat/RTChatSdkIosHelper.h //封装OC接口代码，转换到C++底层引擎接口，定义引擎回调代理函数

RTChat/RTChatSdkIosHelper.mm

将以上几个类添加到IOS应用项目中。

Xcode Build Phases需要引入的系统框架

AVFoundation.framework

AudioToolBox.framework

SystemConfiguration.framework

MobileCoreService.framework

Libz.1.2.5.dylib

libRTChat.a

## 1.2 设置回调代理类及初始化SDK

在工程项目中新建一个语音数据类，继承RTChatSdkIosHelperDelegate，实现onRecvRTChatSDkMsg函数，示例代码如下：

######### VoiceDataManager.h################

#import " RTChatSdkIosHelper.h"

@interface VoiceDataManager : NSObject <RTChatSdkIosHelperDelegate>

+ (VoiceDataManager \*) instance;

@end

######### VoiceDataManager.mm################

#import "VoiceDataManager.h"

@implementation VoiceDataManager

+(VoiceDataManager \*)instance

{

static VoiceDataManager \*sharedInstance = nil;

static dispatch\_once\_t predicate;

dispatch\_once(&predicate, ^{

sharedInstance = [[self alloc] init];

});

return sharedInstance;

}

-(id)init

{

id ret = [super init];

if (!ret) {

return nil;

}

//初始化语音SDK

[[RTChatSdkIosHelper sharedInstance]initHelper:@"jianghu123" key:@"fyuhueirhufda" uniqueid:@"8ddusa09"];

[[RTChatSdkIosHelper sharedInstance]setDelegate:self];

return ret;

}

//语音SDK消息代理函数

-(void)onRecvRTChatSDkMsg:(id)object

{

CallBackObject \*data = object;

if (data) {

switch (data.cmdType) {

case enNotifyCreateResult:

{

if (data.error == OPERATION\_OK) {

StNotifyCreateResult\* result = (StNotifyCreateResult\*)[data.data bytes];

NSLog(@"房间创建成功,roomid=%llu", result->roomid);

}

else {

NSLog(@"房间创建失败");

}

break;

}

case enNotifyRoomList:

{

if (data.error == OPERATION\_OK) {

StNotifyRoomList\* roomlist = (StNotifyRoomList\*)[data.data bytes];

NSLog(@"%d", roomlist->size);

}

}

break;

}

case enNotifyEnterResult:

{

if (data.error == OPERATION\_OK) {

StNotifyEnterResult\* enterresult = (StNotifyEnterResult\*)[data.data bytes];

NSLog(@"成功进入房间%llu", enterresult->roomid);

[[RTChatSdkIosHelper sharedInstance] setMuteSelf:true];

}

break;

}

case enRequestRec:

{

if (data.error == OPERATION\_OK) {

StRequestRec\* result = (StRequestRec\*)[data.data bytes];

self.remoteurlstr = [NSString stringWithUTF8String:result->urlbuf];

}

else {

self.remoteurlstr = @"本次录音失败";

}

break;

}

case enNotifyRandPlay:

{

break;

}

default:

break;

}

}

}

@end

在AppDelegate类中初始化聊天SDK:

RTChatSDKMain::sharedInstance().initSDK("testapp1","cocos2dx32test","test1222222");

其中，initSDK函数中的三个参数分别依次为：appid,appkey,userid。

在主线程帧循环中驱动RTChatSdkIosHelper类：

RTChatSdkIosHelper::instance().visit();

## 1.4 调用OpenAPI

SDK中面向OC开发者的接口类都定义在RTChatSdkIosHelper.h中，开发者可以在自己的工程代码任意位置调用这些接口代码，注意调用处有可能需要用户将文件扩展名由.m改为.mm

调用方式，示例如下：

### 1.4.1 初始化SDK

[[RTChatSdkIosHelper sharedInstance]initHelper:@"xxxx" key:@"xxxx" uniqueid:@"xxxxx"];

参数说明：

appid 你应用的appid

key 应用的appkey

uniqueid 用户id

### 1.4.2 请求登陆

[[RTChatSdkIosHelper sharedInstance] requestLogin];

### 1.4.3 请求房间列表

[[RTChatSdkIosHelper sharedInstance] requestRoomList];

### 1.4.4 创建房间

[[RTChatSdkIosHelper sharedInstance] createRoom: enRoomTypereason: enRoomReason];

参数说明：

enum enRoomType

{

ROOM\_TYPE\_MIN = 0,

ROOM\_TYPE\_QUEUE = 1, /// 单人队列排麦模式(只能一个人拿麦)

ROOM\_TYPE\_FREE = 2, /// 自由模式(最多4个人)

ROOM\_TYPE\_TWO = 3, /// 二人模式

ROOM\_TYPE\_ONLY = 4, /// 抢麦模式

ROOM\_TYPE\_MAX = 5,

};

/// 创建房间的理由

enum enRoomReason

{

ROOM\_REASON\_MIN = 0,

ROOM\_REASON\_NORMAL = 1,

ROOM\_REASON\_RANDCHAT = 2, //随机聊天参数

ROOM\_REASON\_MAX = 3,

};

注：随机聊天时，参数使用ROOM\_TYPE\_FREE，ROOM\_REASON\_RANDCHAT两个。

### 1.4.5 申请加入房间

[[RTChatSdkIosHelper sharedInstance] joinRoom:roomid];

参数说明：roomid 房间id

### 1.4.6 退出房间

[[RTChatSdkIosHelper sharedInstance] leaveRoom];

### 1.4.7 最小化时清理数据

[[RTChatSdkIosHelper sharedInstance] deActivateSDK];

### 1.4.8 激活SDK

//应用从最小化回来时，需要做激活SDK操作

[[RTChatSdkIosHelper sharedInstance] activateSDK];

### 1.4.9 加入麦序及离开麦序

加入麦序

[[RTChatSdkIosHelper sharedInstance] requestInsertMicQueue];

离开麦序

[[RTChatSdkIosHelper sharedInstance] leaveMicQueue];

### 1.4.10 设置是否接受随机聊天

[[RTChatSdkIosHelper sharedInstance] returnRandChatRes: isAccept srctempid: srctempid];

参数说明：isAccept 是否接受随机聊天

Srctempid 指定是否接受随机聊天的用户

### 1.4.11 设置本人静音

[[RTChatSdkIosHelper sharedInstance] setMuteSelf:bool];

参数说明：

isMute 是否静音

详细功能说明，请参照RTChatSDK 的API说明。

# 2 FAQ