

巨人网络集团有限公司

Giant Interactive Group Inc.

**RTChatSDK接入（cocos2dx）流程**

版 本 号: V 1.0

生效日期: 2014-08-28

维护部门: 运营平台中心

手机版本要求：android2.2及以上，IOS6.0以上

编写人 : \_\_\_\_\_\_\_张海\_\_\_\_\_\_\_\_\_ \_\_\_

审核人 : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

批准人 : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

目录

[Giant Interactive Group Inc. 1](#_Toc396996423)

[1 RTChatSDK接入说明 3](#_Toc396996424)

[1.1 Cocos2dx项目中引入JNI调用RTChat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)

[1.5 Android平台上运行 9](#_Toc396996440)

[1.5.1 配置AndroidManifest 9](#_Toc396996441)

[1.5.2 在Activity中添加创建实例 9](#_Toc396996442)

[2 FAQ 10](#_Toc396996443)

[2.1 Cocos2dx中引入RTChat中文件相关问题 10](#_Toc396996444)

[2.1.1找不到android/Log.h头文件 10](#_Toc396996445)

[2.1.2找不到strings.h头文件 10](#_Toc396996446)

[2.1.3找不到jin.h头文件 10](#_Toc396996447)

[2.2 Android中相关问题 11](#_Toc396996448)

[2.2.1在初始化RTChatSDK时黑屏，应用退出 11](#_Toc396996449)

[2.2.2 创建房间、加入房间时，应用退出 11](#_Toc396996450)

# 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：**cocos2dx中引入JNI调用类；

**Step2**：设置回调代理类及初始化SDK;

**Step3**：实现SDK中OpenAPI的接口回调函数；

**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 Cocos2dx项目中引入JNI调用RTChat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 //定义了开放Jni调用RTChatSDK接口函数

RTChat/RTChatSdkAndroidIpml.cpp //实现了Jni调用开放接口函数

RTChat/RTChatSdkAndroidIpml.h //定义了jni调用的接口函数

RTChat/RTChatSdkIosHelper.h //定义了Jni函数的回调函数以及将更新ui操作放在主线程

RTChat/RTChatSdkIosHelper.cpp //实现了在主线程中对ui的操作

将以上几个类添加到Cocos2dx项目中。这几个类实现了jni调用java操作。

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

打开Cocos2dx项目，在需要使用RTChatSDK处初始化SDK，在初始化之前需要注册一个回调并且为这个回调注册一个代理类，示例代码如下：

Class voicedatamanager : public RTChatSdkIosHelperDelegate {

Public:

void init() {

//设置代理

RTChatSdkIosHelper::instance().setDelegate(this);

};

virtual void onNotifyLoginResult(SdkErrorCode code, uint64\_t tempid);

.....

}

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();

## 1.3 实现SDK中OpenAPI的接口回调函数

主要实现SdkIosHelper.h中代理类RTChatSdkIosHelperDelegate中函数，主要实现的回调有登陆、创建房间、离开房间、请求麦序和离开麦序以及发起随机聊天等函数的回调，示例代码如下：

//请求登录的回调

void RTChatOverride::onNotifyLoginResult(SdkErrorCode error, uint64\_t tempid)

{

//请求登陆成功返回的结果

if (error == OPERATION\_OK)

{//登陆成功需要做的操作

CCLOG("the login result === userid = %lld",tempid);

}else

{//登陆失败的操作

CCLOG("Login failed.=====\n");

}

}

//向服务器请求房间列表时的回调

//error 错误码

//infoVec 房间信息集合

void RTChatOverride::onNotifyRoomList(SdkErrorCode error, const std::vector<StRoomInfo> infoVec)

{

if(error==OPERATION\_OK)

{

}else

{

CCLOG("Failed to request room list.===== \n");

}

}

//执行创建房间操作后的回调函数

//error 错误码

//roomid 房间id

//roomtype 房间类型

void RTChatOverride::onNotifyCreateResult(SdkErrorCode error, uint64\_t roomid, enRoomType roomtype)

{

if(error == OPERATION\_OK)

{

CCLOG("created roomInfo======:roomid=%lld,roomtype=%d",roomid,roomtype);

}

else

{

CCLOG("Failed to create room.======\n");

}

}

//请求进入房间时的回调函数

//error 错误码，成功返回OPERATION\_OK

//roomid 进入的房间的id

//roomtype 进入的房间类型

void RTChatOverride::onNotifyEnterResult(SdkErrorCode error, uint64\_t roomid, enRoomType roomtype)

{

if (error == OPERATION\_OK)

{

CCLOG("Successed join in room %lld",roomid);

}else

{

CCLOG("Failed to join in room %lld",roomid);

}

}

//当有人进入房间时通知有人进入房间的回调,申请加入房间成功后通知

//error 错误码

//infoVec 进入房间的人集合，可以多人同时进入

void RTChatOverride::onNotifySomeEnterRoom(SdkErrorCode error, const std::vector<stRoomUserInfo> infoVec)

{

if (error == OPERATION\_OK)

{

CCLog("Some user %lld enter room success.",infoVec[0].uniqueid);

}else

{

CCLog("Some user %lld enter room failed.",infoVec[0].uniqueid);

}

}

//通知有人离开房间的回调，调用leaveRoom()时的回调

//error 错误码

//tempid 进入房间的人的id

void RTChatOverride::onNotifySomeLeaveRoom(SdkErrorCode error, uint64\_t tempid)

{

if (error == OPERATION\_OK)

{

CCLog("user %lld leave room success.",tempid);

}else

{

CCLog("user %lld leaves room failed.",tempid);

}

}

//服务器通知当前房间麦序(在用户申请加入麦序或用户离开麦序均会执行该回调函数)

//error 结果码

//infoVec 用户麦序列表

void RTChatOverride::onNotifyMicQueue(SdkErrorCode error, const std::vector<StMicInfo> infoVec)

{

CCLOG("RTChatOverride::onNotifyMicQueue()\n");

}

//服务器通知当前获取说话权限的用户

//error结果码

//tempid用户id

//mtime 时间

void RTChatOverride::onNotifyTakeMic(SdkErrorCode error, uint64\_t tempid, uint32\_t mtime)

{

CCLOG("RTChatOverride::onNotifyTakeMic()\n");

}

//通知有人想和你随机聊天 ，调用createRoom()时，roomType为ROOM\_TYPE\_FREE，并且//reason为ROOM\_REASON\_NORMAL时，会执行该回调

//error 结果码

//srctempid 用户id，发起聊天的用户id

//uniqueidstr 用户名字，发起聊天的用户的名字

//roomid 房间id，发起聊天的用户所在的房间id

void RTChatOverride::onNotifyRandChat(SdkErrorCode error, uint64\_t srctempid, const std::string uniqueidstr, uint64\_t roomid)

{

CCLOG("RTChatOverride::onNotifyRandChat()\n");

}

//告知发起随机聊天结果

//error 结果码

void RTChatOverride::onReturnRandChat(SdkErrorCode error)

{

CCLOG("RTChatOverride::onReturnRandChat()\n");

}

在以上回调的结果中，可实现自己需要做的操作，比如更新ui等。

注：要实现能够接收到回调，需要在初始化SDK时，增加回调代理类，参照1.4

## 1.4 调用OpenAPI

SDK中VoiceChannelEngine类是对外功能接口的主要实现类，所有接口都通过VoiceChannelEngine类来调用。

RTChatSdkAndroidIpml.cpp中定义了一个RTChatSDKMain类，该类实现了一些函数，这些函数通过JNI调用VoiceChannelEngine类中接口方法，在使用的时候，可以直接调用RTChatSDKMain类中的函数。调用方式，示例如下：

### 1.4.1 初始化SDK

RTChatSDKMain::sharedInstance().initSDK(const std::string& appid, const std::string& key, const char\* uniqueid = NULL);

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

RTChatSDKMain::sharedInstance().requestLogin("test2233");

### 1.4.3 请求房间列表

RTChatSDKMain::sharedInstance().requestRoomList();

### 1.4.4 创建房间

RTChatSDKMain::sharedInstance().createRoom(enRoomType roomType, enRoomReason reason);

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

RTChatSDKMain::sharedInstance().joinRoom(uint64\_t roomid);

参数说明：roomid 房间id

### 1.4.6 退出房间

RTChatSDKMain::sharedInstance().leaveRoom();

### 1.4.7 最小化时清理数据

RTChatSDKMain::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

RTChatSDKMain::sharedInstance().activateSDK();

### 1.4.9 加入麦序及离开麦序

加入麦序

RTChatSDKMain::sharedInstance().requestInsertMicQueue();

离开麦序

RTChatSDKMain::sharedInstance().leaveMicQueue();

### 1.4.10 设置是否接受随机聊天

RTChatSDKMain::sharedInstance().returnRandChatRes(bool isAccept, uint64\_t 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

RTChatSDKMain::sharedInstance().setMuteSelf(bool isMute);

参数说明：

isMute 是否静音

VoiceChannelEngine类中开放接口的详细功能说明，请参照RTChatSDK 的API说明。

## 1.5 Android平台上运行

### 1.5.1 配置AndroidManifest

在cocos2d-x项目对应的android应用的AndroidManifest.xml增加以下权限配置（注：如果不加会导致无法使用SDK中的部分openAPI）

<uses-permission android:name=*"android.permission.MODIFY\_AUDIO\_SETTINGS"* />

<uses-permission android:name=*"android.permission.RECORD\_AUDIO"* />

<uses-permission android:name=*"android.permission.INTERNET"* />

<uses-permission android:name=*"android.permission.WAKE\_LOCK"* />

<uses-permission android:name=*"android.permission.WRITE\_EXTERNAL\_STORAGE"*/>

<uses-permission android:name=*"android.permission.ACCESS\_WIFI\_STATE"*/>

其中，如果有权限已配置，则无需重复添加。

### 1.5.2 在Activity中添加创建实例

创建SDK的实例主要调用类VoiceChannelEngine类，方式如下，

打开cocos2d的android项目下自动生成的Activity类中（注:不同的cocos2dx版本不同，可能生成的activity类名称不同）添加下代码：

**public** **class** AppActivity **extends** Cocos2dxActivity {

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

//加上这句话 可以初始化instance，否则后面调用jni时不会成功

VoiceChannelEngine.*Create*(**this**, VoiceChannelEngine.enInterfaceType.***INTERFACE\_NATIVE***);

}

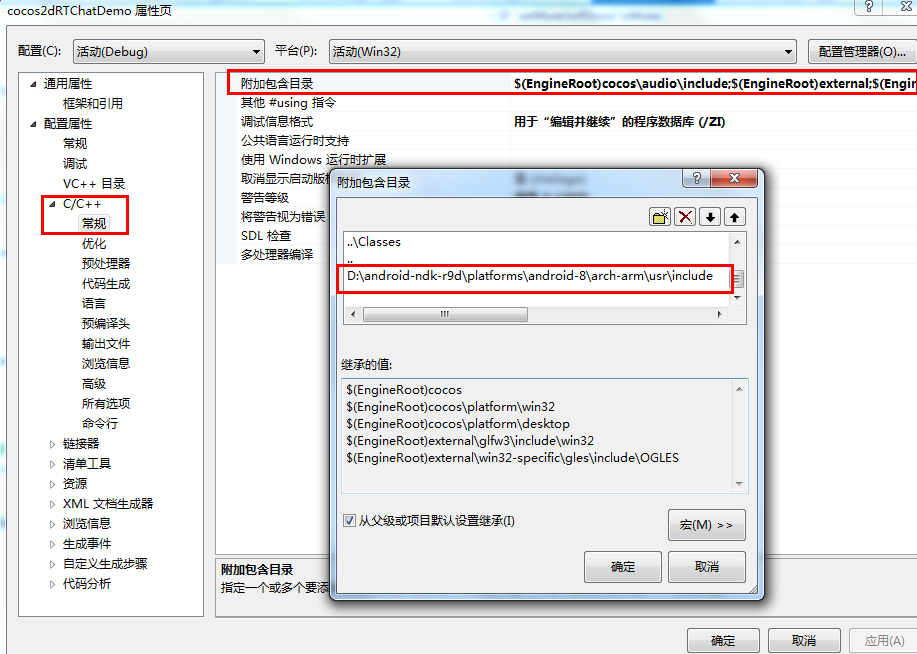
}

# 2 FAQ

## 2.1 Cocos2dx中引入RTChat中文件相关问题

### 2.1.1找不到android/Log.h头文件

在visual studio下 【项目】→【属性】→C/C++→常规，在附加包含目录中添加下目录：android ndk目录/platforms/android-8/arch-arm/usr/include，如下图：



### 2.1.2找不到strings.h头文件

同2.1.1方法

### 2.1.3找不到jin.h头文件

在%JAVA\_HOME%/include/下找到 jni.h，复制到“Visual Studio目录/VC/include/”下，即可解决问题

## 2.2 Android中相关问题

### 2.2.1在初始化RTChatSDK时黑屏，应用退出

解决方式：

可能是忘记在Activity中调用VoiceChannelEngine类中的create()方法去初始化instance变量。详情[参照1.5.2](#_1.5.2_在Activity中添加创建实例).

### 2.2.2 创建房间、加入房间时，应用退出

解决方式：

可能是没有在AndroidManifest.xml中添加相应权限，添加权限方式，详情[参照1.5.1](#_1.5.1_配置AndroidManifest).