AnyChat for Android SDK

开发手册

（版本：V8.0）

广 州 佰 锐 网 络 科 技 有 限 公 司

Guangzhou BaiRui Network Technology Co.,Ltd.

http://www.bairuitech.com http://www.anychat.cn

2020 年 01 月

目 录

1 系统概述

非常感谢您使用佰锐科技的产品，我们将为您提供最好的服务。

本手册可能包含技术上不准确的地方或排版错误。本手册的内容将做定期的更新，恕不另行通知；更新的内容将会在本手册的新版本中加入。我们随时会改进或更新本手册中描述的产品或程序。

1.1 系统介绍

AnyChat for Android SDK 是一套即时通讯开发平台（SDK），包含了音视频处理模块、P2P 网络模块、音视频录制、文件传输、数据通信以及常见业务流程封装等模块，是 AnyChat Platform Core SDK 的重要组成部分，专为 Android 平台设计，并针对 ARM 系列 CPU 进行了汇编优化，支持 ARM、arm64-v8a 等常见 Android 平台的 CPU 架构，可以做为 Android 平台上的即时通讯内核引擎，也可以做为视频会议、网络教育、即拍即传系统等互动平台的核心库。整个平台由广州佰锐网络科技有限公司独立研发，具有自主知识产权。

AnyChat SDK 分为客户端 SDK 和服务器 SDK 两大部分，其中客户端 SDK 用于实现语音、视频的交互以及其它客户端相关的功能，而服务器 SDK 主要实现业务层逻辑控制，以及与第三方平台的互联等。AnyChat for Android SDK 提供 Java 语言接口。

1.2 系统特性

“AnyChat for Android SDK”采用增强的 H.264 视频编码算法和 AMR 语音编码算法，具有高画质、语音清晰、流畅的特点，支持 P2P 技术进行网络传输，服务器采用完成端口模型的重叠 IO，具有极高的并发处理能力。

服务器支持“SDK Filter Plus”和“AnyChat Server SDK”两种可扩展编程接口，可方便实现与其它系统进行集成，增强 AnyChat 的可扩展性。上层应用也可利用服务器 SDK 来实现更复杂的业务逻辑处理。

1.2.1 视频技术

视频制式：PAL-B

分辨率： 160×120 —1920×1080（可调节）

帧 率： 1~25（可调节）

视频编码器：H.264

视频流码率：10kbps ~ 1000kbps（VBR）

支持硬件编码、解码（可定制，需提供对应平台的接口）

*高分辨率需硬件支持

1.2.2 音频技术

采样率：16000 Hz、22050Hz、44100Hz、48000Hz（可设置）

量化值：16 bit

声 道：Mono、Stereo

音频编码器：AMR_WB、AAC

音频流码率：6kbps ~ 128kbps

音效处理：回音抑制（AES）、噪音抑制（NS）、自动增益控制（AGC）、静音检测（VAD）

1.2.3 P2P 技术

传输方式：UDP、TCP

支持的 NAT 类型：

Cone NAPT <——> Cone NAPT

Cone NAPT <——> Symmetric NAT

支持 UPNP 协议（LAN、WiFi 网络中有效）

1.3 关于佰锐科技

广州佰锐网络科技有限公司（简称“佰锐”，英文：GuangZhou BaiRui Network

Technology Co., Ltd.）始创于 2005 年，是一家专业从事网络语音视频技术研究与系统开发的国家高新技术企业。成立至今，已为国内外上千家客户提供优质的产品与服务。

14 年来佰锐一直专注于音视频底层技术研究和基础数据通信能力的研究，打造了一款具有独立知识产权、世界一流的跨平台音视频解决方案产品“AnyChat”，该产品家族包括“AnyChat SDK”、“AnyChat 服务集群”和“AnyChat 视频云平台”。实现了线下线上音视频能力的全部贯通，一站式解决高清视频、全景录像、音视频呼叫、智能排队等技术难题，满足金融、证券、医疗、教育、智能设备等行业客户在不同应用领域中的音视频实时通信需求，成为首屈一指的音视频能力解决方案提供商，AnyChat 更是成为金融领域音视频应用第一品牌。

更多信息请参考佰锐科技官网：http://www.bairuitech.com

1.4 技术支持

在您使用 AnyChat SDK 的过程中，遇到任何困难，请与我们联系，我们将热忱为您提供帮助。

您可以通过如下方式与我们取得联系：

1、在线论坛：http://bbs.anychat.cn

2、公司官网：http://www.bairuitech.com

3、AnyChat 产品网站：http://www.anychat.cn

4、电子邮件：service@bairuitech.com

5、24 小时客服电话：+86 （020） 85276986、38109065、38103410

1.5 版权申明

“AnyChat for Android SDK”是由广州佰锐网络科技有限公司开发，拥有自主知识产权（软著登字第 066348 号）的系统平台，

广州佰锐网络科技有限公司拥有与本产品所用技术相关的知识产权。这些知识产权包括但不限于一项或多项发明专利或者正在进行申请的专利（2006101239829、2006101241176）。

第三方组件：AnyChat 使用了开源的第三方组件，包括：FFmpeg（LGPL 2.1）、libvpx（BSD）、libspeex（BSD）、WebRTC（BSD）。其中 FFmpeg 的版权所有者信息、源代码以及其它信息均可在其官方网站：http://www.ffmpeg.org 找到。

2 编程指南

2.1 客户端 SDK 概述

“AnyChat for Android”属于客户端组件（简称“客户端”），对上层应用提供纯 Java 语言的调用接口，内核是由一系列的.so 库（相当于 Win32 平台的 DLL）组合而成，采用 JNI 技术实现 Java 层与内核层的通信。

系统采用模块化设计，每个模块都独立完成特定的任务，模块之间采用弱关联设计，今后系统某部分功能的升级，如音频、视频编码算法的改进，只需要替换相关的模块即可，不影响系统的接口。

AnyChat for Android 与服务器有一系列的交互过程，包括：连接服务器、登录系统、进入房间，交互过程的结果（如连接服务器是否成功）SDK 内部将会采用 Java 接口技术（Windows 平台是采用消息机制）通知上层应用。只有进入同一房间的两个用户之间才能进行语音、视频、文字的交互，当某用户打开了本地设备后，其它用户请求该用户的数据时，便能收到该用户的数据。

AnyChat for Android 客户端在房间中，收到其它用户的流媒体数据后，上层应用只需要提供一个 SurfaceView 控件的句柄，内核便可自动显示视频到该 SurfaceView 控件上，并自动播放声音。

AnyChat for Android 的工作流程与 Windows 平台的 SDK 一致，熟悉 Windows 平台的 SDK 工作机制将更有助于了解 AnyChat for Android 平台的工作机制。

2.2 函数调用顺序

调用顺序	函数	功能	备注
A	InitSDK	初始化系统	初始化
	SetSDKOptionInt	设置 SDK 整形值相关参数	
	SetSDKOptionString	设置 SDK 字符串相关参数	
B	Connect	连接服务器	进入系统
	Login	用户登录系统	
	LoginEx	用户登录视频云平台	
	EnterRoom	进入房间	
C	GetOnlineUser	获取当前房间在线用户列表	其它功能
	GetRoomOnlineUsers	获取指定房间在线用户列表	
	GetCameraState	查询用户摄像头的状态	
	GetSpeakState	查询用户发言状态	
	ShowLVProperty	显示本地视频画面调节对话框	
	EnumVideoCapture	枚举本地视频采集设备	
	SelectVideoCapture	选择指定的视频采集设备	
	GetCurVideoCapture	获取当前使用的视频采集设备	
	EnumAudioCapture	枚举本地音频采集设备	
	SelectAudioCapture	选择指定的音频采集设备	
	GetCurAudioCapture	获取当前使用的音频采集设备	
	AudioGetVolume	获取指定音频设备的当前音量	
	AudioSetVolume	设置指定音频设备的音量	
	SetVideoPos	设置视频显示位置	
	UserCameraControl	操作用户视频	
	UserSpeakControl	操作用户语音	
	SendTextMessage	发送文本消息	
	ChangeChatMode	更改当前的聊天模式	
	GetUserChatMode	获取指定用户当前的聊天模式	
	PrivateChatRequest	请求与对方私聊，发起私聊请求	
	PrivateChatEcho	回复对方的私聊请求	
	PrivateChatExit	退出与某用户的私聊	
	……	……	
D	LeaveRoom	离开房间	退出系统

	Logout	用户注销	
	Release	释放资源	

2.3 编程要点

2.3.1 SDK 的 Java Package

AnyChat for Android 的 package 路径是：com.bairuitech.anychat，由于内核采用了 JNI 技术，需要与该包进行交互，所以上层不能修改包的名称。

2.3.2 SDK 与 Activity

Android 应用是由一个或多个 Activity 组合而成，每个 Activity 都有其生命周期（可参考 Android 相关开发文档），而 AnyChat for Android 内核采用 JNI 技术，并进行了特殊的设计，使得 AnyChat for Android 可以在多个 Activity 中共享，即 AnyChat for Android 的生命周期是从第一个 Activity 调用 InitSDK 方法开始，到最后一个 Activity 调用 Release 方法结束，中间的 Activity 不需要再初始化 SDK。这样便可实现在第一个 Activity 中登录之后，便于立即切换第二个 Activity 继续操作，而不需要在第二个 Activity 中再进行登录的操作。

2.3.3 SurfaceView 视频显示

AnyChat for Android 采用 SurfaceView 进行视频显示，上层应用只需要在 Activity 中创建一个 SurfaceView 控件，然后将控件句柄通过 SetVideoPos 方法传递给 SDK，则当有视频数据到达时，内核将会自动将视频显示到该 SurfaceView 控件上，不需要上层应用来处理视频的显示。

需要注意的是，部分 Android 设备显示本地视频时，需要启动 Overlay 模式，否则不能进行本地视频的预览和采集，当启动 Overlay 模式时，需要对 SurfaceView 控件进行相关属性的设置，详细信息可参考随 AnyChat for Android 一同发布的 Demo 程序源代码。

2.3.4 字符编码

Android 平台上层是 Java 虚拟机，采用 Unicode 编码，AnyChat for Android 内核在处理 Android 与 Windows 平台的通信时，会自动将字符串（如文字聊天数据）转换为上层平台所对应的编码，不需要应用层进行转换，但是当上层应用使用透明通道在客户端与服务器，或是客户端之间传输数据，而需要显示时，就需要上层应用手动来处理编码转换的任务，否则显示将会出现乱码。

2.4 服务器 SDK 概述

“AnyChat Server SDK”与“SDK Filter Plus”均是服务器扩展编程接口，均为动态连接库（DLL）形式，两者的主要区别是：（1）、“SDK Filter Plus”的 DLL 被 AnyChat 核心服务器程序（AnyChatCoreServer.exe）调用，与 AnyChat 核心服务器程序属同一个进程；（2）、“AnyChat Server SDK”被业务层服务器程序（需要用户编写）调用，与 AnyChat 核心服务器程序属不同的进程，与 AnyChat 核心服务器采用 IPC 的方式进行通信。

“AnyChat Server SDK”与“SDK Filter Plus”两者可以实现相同的功能，通常来说，“SDK Filter Plus”适合业务逻辑较简单的应用，而“AnyChat Server SDK”则适合业务逻辑较复杂的应用，实现独立的业务层服务器，有对应的界面显示。

有关“SDK Filter Plus”的详细介绍可参考相关的开发文档（《AnyChat SDK Filter Plus 开发指南》）及 SDK 包中所附带的相关源代码。

有关“AnyChat Server SDK”的详细介绍可参考相关的开发文档（《AnyChat Server SDK 开发指南》）及 SDK 包中所附带的相关源代码。

有关 AnyChat 平台用户身份验证与第三方平台集成的问题可参考技术论坛相关介绍：

http://bbs.anychat.cn/forum.php?mod=viewthread&tid=12&extra=page%3D1

服务器与客户端之间可以传输缓冲区、文件等数据，详情可参考在线文档：

http://www.anychat.cn/faq/index.php?action=artikel&cat=2&id=206&artlang=zh

3 数据结构及常量定义

3.1 SDK 内核参数定义

在使用 API“SetSDKOption”、“SetSDKOptionString” 、“GetSDKOptionInt” 、“GetSDKOptionString”来设置、获取 SDK 内核心参数时，可根据不同的参数类型设置、获取不同的值，目前可用参数类型定义如下：

分类	参数	说明
基础	BRAC_SO_CORESDK_TMPDIR	设置 AnyChat Core SDK 临时目录
	BRAC_SO_CORESDK_MAGICADJUST	内核调试参数
	BRAC_SO_CORESDK_LOADCODEC	加载外部编解码器
	BRAC_SO_CORESDK_USEARMV6LIB	强制使用 ARMv6 指令集的库，android 平台使用
	BRAC_SO_CORESDK_USEHWCODEC	使用平台内置硬件编解码器
	BRAC_SO_CORESDK_REMOTEDEBUG	远程调试
	BRAC_SO_CORESDK_PATH	设置 AnyChat Core SDK 相关组件路径
	BRAC_SO_CORESDK_DUMPCOREINFO	输出内核信息到日志文件中，便于分析故障原因
	BRAC_SO_CORESDK_MAINVERSION	查询 SDK 主版本号
	BRAC_SO_CORESDK_SUBVERSION	查询 SDK 从版本号
	BRAC_SO_CORESDK_BUILDTIME	查询 SDK 编译时间
	BRAC_SO_CORESDK_ACTIVESTATE	应用程序活动状态控制
	BRAC_SO_CORESDK_EXTVIDEOINPUT	外部扩展视频输入控制
	BRAC_SO_CORESDK_EXTAUDIOINPUT	外部扩展音频输入控制
	BRAC_SO_CORESDK_LOWDELAYCTRL	低延迟模式控制
	BRAC_SO_PROXY_FUNCTIONCTRL	本地用户代理功能控制
音频	BRAC_SO_AUDIO_VADCTRL	音频静音检测控制
	BRAC_SO_AUDIO_NSCTRL	音频噪音抑制控制
	BRAC_SO_AUDIO_ECHOCTRL	音频回音消除控制
	BRAC_SO_AUDIO_AGCCTRL	音频自动增益控制
	BRAC_SO_AUDIO_CAPTUREMODE	音频采集模式设置
	BRAC_SO_AUDIO_MICBOOST	音频采集 Mic 增强控制
	BRAC_SO_AUDIO_AUTOPARAM	根据音频采集模式，自动选择合适的相关参数
	BRAC_SO_AUDIO_MONOBITRATE	设置单声道模式下音频编码目标码率
	BRAC_SO_AUDIO_STEREOBITRATE	设置双声道模式下音频编码目标码率

	BRAC_SO_AUDIO_PLAYDRVCTRL	音频播放驱动选择
	BRAC_SO_AUDIO_CNGCTRL	舒适噪音生成控制
	BRAC_SO_AUDIO_CODECID	本地音频编码器 ID 设置
	BRAC_SO_AUDIO_SOFTVOLMODE	设置软件音量模式控制
	BRAC_SO_AUDIO_RECORDDRVCTRL	音频采集驱动控制
	BRAC_SO_PROXY_AUDIOCTRL	本地用户代理音频控制，将本地音频变为指定用户的音频对外发布
视频	BRAC_SO_LOCALVIDEO_BITRATECTRL	本地视频编码码率设置
	BRAC_SO_LOCALVIDEO_QUALITYCTRL	本地视频编码质量因子控制
	BRAC_SO_LOCALVIDEO_GOPCTRL	本地视频编码关键帧间隔控制
	BRAC_SO_LOCALVIDEO_FPSCTRL	本地视频编码帧率控制
	BRAC_SO_LOCALVIDEO_PRESETCTRL	本地视频编码预设参数控制
	BRAC_SO_LOCALVIDEO_APPLYPARAM	应用本地视频编码参数，使得前述修改即时生效
	BRAC_SO_LOCALVIDEO_VIDEOSIZEPOLITIC	本地视频采集分辨率控制策略
	BRAC_SO_LOCALVIDEO_DEINTERLACE	本地视频反交织参数控制
	BRAC_SO_LOCALVIDEO_WIDTHCTRL	本地视频采集分辨率宽度控制
	BRAC_SO_LOCALVIDEO_HEIGHTCTRL	本地视频采集分辨率高度控制
	BRAC_SO_LOCALVIDEO_FOCUSCTRL	本地视频摄像头对焦控制
	BRAC_SO_LOCALVIDEO_PIXFMTCTRL	本地视频采集优先格式控制
	BRAC_SO_LOCALVIDEO_OVERLAY	本地视频采用 Overlay 模式
	BRAC_SO_LOCALVIDEO_CODECID	本地视频编码器 ID 设置
	BRAC_SO_LOCALVIDEO_ROTATECTRL	本地视频旋转控制
	BRAC_SO_LOCALVIDEO_CAPDRIVER	本地视频采集驱动设置
	BRAC_SO_LOCALVIDEO_FIXCOLORDEVIA	修正视频采集颜色偏色
	BRAC_SO_LOCALVIDEO_TVFORMAT	视频采集制式设置
	BRAC_SO_LOCALVIDEO_OVERLAYTIMESTAMP	迭加时间戳到本地视频
	BRAC_SO_LOCALVIDEO_DEVICENAME	本地视频采集设备名称
	BRAC_SO_PROXY_VIDEOCTRL	本地用户代理视频控制，将本地视频变为指定用户的视频对外发布
录制	BRAC_SO_RECORD_VIDEOBR	录像视频码率设置
	BRAC_SO_RECORD_AUDIOBR	录像音频码率设置
	BRAC_SO_RECORD_TMPDIR	录像文件临时目录设置
	BRAC_SO_SNAPSHOT_TMPDIR	快照文件临时目录设置
	BRAC_SO_RECORD_FILETYPE	录制文件类型设置
	BRAC_SO_RECORD_WIDTH	录制视频宽度设置
	BRAC_SO_RECORD_HEIGHT	录制文件高度设置
	BRAC_SO_RECORD_FILENAMERULE	录制文件名命名规则
	BRAC_SO_RECORD_CLIPMODE	录制视频裁剪模式
	BRAC_SO_RECORD_DISABLEDATEDIR	录制文件不按日期分目录保存，全部生成在指定文件夹中
	BRAC_SO_STREAM_MAXBUFFERTIME	最大流缓冲时间

	BRAC_SO_STREAM_SMOOTHPLAYMODE	平滑播放模式
网络	BRAC_SO_NETWORK_P2PPOLITIC	本地网络 P2P 策略控制
	BRAC_SO_NETWORK_P2PCONNECT	尝试与指定用户建立 P2P 连接
	BRAC_SO_NETWORK_P2PBREAK	断开与指定用户的 P2P 连接
	BRAC_SO_NETWORK_TCPSERVICEPORT	设置本地 TCP 服务端口
	BRAC_SO_NETWORK_UDPSERVICEPORT	设置本地 UDP 服务端口
	BRAC_SO_NETWORK_MULTICASTPOLITIC	组播策略控制
	BRAC_SO_NETWORK_TRANSBUFMAXBITRATE	传输缓冲区、文件最大码率控制
	BRAC_SO_NETWORK_AUTORECONNECT	网络掉线自动重连功能控制
	BRAC_SO_UDPTRACE_MODE	UDP 数据包跟踪模式
	BRAC_SO_UDPTRACE_PACKSIZE	UDP 数据包跟踪的大小，单位：BYTE
	BRAC_SO_UDPTRACE_BITRATE	UDP 数据包跟踪的包速率，单位：bps
	BRAC_SO_UDPTRACE_START	UDP 数据包跟踪控制
	BRAC_SO_UDPTRACE_LOCALRECVNUM	UDP 数据包跟踪本地接收包数量
	BRAC_SO_UDPTRACE_SERVERRECVNUM	UDP 数据包跟踪服务器接收包数量
	BRAC_SO_UDPTRACE_SOURCESENDNUM	UDP 数据包跟踪源发包数量
	BRAC_SO_UDPTRACE_SENDUSERID	UDP 数据包跟踪源用户 ID
视频显示	BRAC_SO_VIDEOSHOW_MODECTRL	视频显示模式控制
	BRAC_SO_VIDEOSHOW_SETPRIMARYUSER	设置主显示用户编号
	BRAC_SO_VIDEOSHOW_GPUDIRECTRENDER	视频数据经过 GPU 直接渲染
	BRAC_SO_VIDEOSHOW_AUTOROTATION	远程视频显示自动旋转控制（
	BRAC_SO_VIDEOSHOW_SETOVERLAYUSER	设置迭加显示用户编号
	BRAC_SO_VIDEOSHOW_DRIVERCTRL	视频显示驱动控制
	BRAC_SO_VIDEOSHOW_CLIPMODE	远程视频显示旋转裁剪模式
其他	BRAC_SO_OBJECT_INITFLAGS	业务对象身份初始化
	BRAC_SO_CLOUD_APPGUID	云平台应用 GUID

3.2 视频采集驱动定义

序号	参数	说明
1	VIDEOCAP_DRIVER_JAVA	Java 视频采集驱动
2	VIDEOSHOW_DRIVER_JAVA	Java 视频显示驱动
3	AUDIOREC_DRIVER_JAVA	Java 音频采集驱动
4	AUDIOPLAY_DRIVER_JAVA	Java 音频播放驱动

3.3 用户状态标志常量定义

在使用 API“QueryUserStateInt”、“QueryUserStateString”来获取指定用户相关的参数信息，目前可用参数定义如下：

序号	参数	说明
1	BRAC_USERSTATE_CAMERA	用户摄像头状态
2	BRAC_USERSTATE_HOLDMIC	用户音频设备状态
3	BRAC_USERSTATE_SPEAKVOLUME	用户当前说话音量
4	BRAC_USERSTATE_RECORDING	用户录像（音）状态
5	BRAC_USERSTATE_LEVEL	用户级别
6	BRAC_USERSTATE_NICKNAME	用户昵称
7	BRAC_USERSTATE_LOCALIP	用户本地 IP 地址
8	BRAC_USERSTATE_INTERNETIP	用户互联网 IP 地址
9	BRAC_USERSTATE_VIDEOBITRATE	用户当前的视频码率
10	BRAC_USERSTATE_AUDIOBITRATE	用户当前的音频码率
11	BRAC_USERSTATE_P2PCONNECT	查询本地用户与指定用户的当前 P2P 连接状态
12	BRAC_USERSTATE_NETWORKSTATUS	查询指定用户的网络状态
13	BRAC_USERSTATE_VIDEOSIZE	查询指定用户的视频分辨率
14	BRAC_USERSTATE_PACKLOSSRATE	查询指定用户的网络流媒体数据丢包率
15	BRAC_USERSTATE_DEVICETYPE	查询指定用户的终端类型

3.4 录像相关常量定义

在使用 API“BRAC_StreamRecordCtrl”接口进行录制时，需要传入不同的参数或定义常量。目前可用参数定义如下：

序号	参数	说明

1	BRAC_RECORD_FLAGS_VIDEO	录制视频
2	BRAC_RECORD_FLAGS_AUDIO	录制音频
3	BRAC_RECORD_FLAGS_SERVER	服务器端录制
4	BRAC_RECORD_FLAGS_MIXAUDIO	录制音频时，将其它人的声音混音后录制
5	BRAC_RECORD_FLAGS_MIXVIDEO	录制视频时，将其它人的视频迭加后录制
6	BRAC_RECORD_FLAGS_ABREAST	录制视频时，将其它人的视频并列录制
7	BRAC_RECORD_FLAGS_STEREO	录制音频时，将其它人的声音混合为立体声后录制
8	BRAC_RECORD_FLAGS_SNAPSHOT	拍照
9	BRAC_RECORD_FLAGS_LOCALCB	触发本地回调
10	BRAC_RECORD_FLAGS_STREAM	对视频流进行录制
11	BRAC_RECORD_FLAGS_USERFILENAME	用户自定义文件名

3.5 业务对象常量定义

AnyChat SDK 业务队列功能新定义了业务队列所涉及到的业务对象及业务对象相关属性及方法、事件常量。

3.5.1 对象类型定义

3.5.1.1 对象类型定义

```
ANYCHAT_OBJECT_TYPE_AREA        =   4;      ///< 服务区域
ANYCHAT_OBJECT_TYPE_QUEUE       =   5;      ///< 队列对象
ANYCHAT_OBJECT_TYPE_AGENT       =   6;      ///< 客服对象
ANYCHAT_OBJECT_TYPE_CLIENTUSER=   8;      ///< 客户端用户对象，用于与服务器交换数据用户对象，用于与服务器交换数据
```

3.5.1.2 通用标识定义

```
ANYCHAT_OBJECT_FLAGS_CLIENT     =   0;      ///< 普通客户
ANYCHAT_OBJECT_FLAGS_AGENT      =   2;      ///< 坐席用户
ANYCHAT_OBJECT_FLAGS_MANANGER   =   4;      ///< 管理用户
ANYCHAT_INVALID_OBJECT_ID       =   -1;     ///< 无效的对象 ID
```

3.5.2 对象属性定义

3.5.2.1 对象公共属性定义

```
ANYCHAT_OBJECT_INFO_FLAGS       =   7;      ///< 对象属性标志
ANYCHAT_OBJECT_INFO_NAME        =   8;      ///< 对象名称
ANYCHAT_OBJECT_INFO_PRIORITY    =   9;      ///< 对象优先级
ANYCHAT_OBJECT_INFO_ATTRIBUTE   =   10;     ///< 对象业务属性
ANYCHAT_OBJECT_INFO_DESCRIPTION     =   11;     ///< 对象描述
ANYCHAT_OBJECT_INFO_INTTAG      =   12;     ///< 对象标签，整型，上层应用自定义
ANYCHAT_OBJECT_INFO_STRINGTAG   =   13;     ///< 对象标签，字符串，上层应用自定义
```

3.5.2.2 服务区域属性定义

```
ANYCHAT_AREA_INFO_AGENTCOUNT =  401;    ///< 服务区域客服用户数
ANYCHAT_AREA_INFO_GUESTCOUNT =  402;    ///< 服务区域内访客的用户数（没有排入队列的用户）
ANYCHAT_AREA_INFO_QUEUEUSERCOUNT =  403; ///< 服务区域内排队的用户数
ANYCHAT_AREA_INFO_QUEUECOUNT =  404;    ///< 服务区域内队列的数量
```

3.5.2.3 队列属性定义

```
ANYCHAT_QUEUE_INFO_MYSEQUENCENO  =  501;    ///< 自己在该队列中的序号
ANYCHAT_QUEUE_INFO_BEFOREUSERNUM =  502;    ///< 排在自己前面的用户数
ANYCHAT_QUEUE_INFO_MYENTERQUEUETIME= 503    ///< 进入队列的时间
ANYCHAT_QUEUE_INFO_LENGTH        =  504;    ///< 队列长度（有多少人在排队），整型
ANYCHAT_QUEUE_INFO_WAITTIMESECOND    =  508; ///< 自己在队列中的等待时间（排队时长），单位：秒
```

3.5.2.4 客服属性定义

```
ANYCHAT_AGENT_INFO_SERVICESTATUS        =   601; ///< 服务状态，整型
ANYCHAT_AGENT_INFO_SERVICEUSERID        =   602; ///< 当前服务的用户ID，整型
ANYCHAT_AGENT_INFO_SERVICEBEGINTIME     =   603; ///< 当前服务的开始时间，整型
ANYCHAT_AGENT_INFO_SERVICETOTALTIME     =   604; ///< 累计服务时间，整型，单位：秒
```

```
ANYCHAT_AGENT_INFO_SERVICETOTALNUM=  605;          ///< 累计服务的用户数，整型
```

3.5.2.5 客服服务状态定义

```
ANYCHAT_AGENT_STATUS_CLOSEED  =  0;        ///< 关闭，不对外提供服务
ANYCHAT_AGENT_STATUS_WAITTING =  1;        ///< 等待中，可随时接受用户服务
ANYCHAT_AGENT_STATUS_WORKING  =  2;        ///< 工作中，正在为用户服务
ANYCHAT_AGENT_STATUS_PAUSED   =  3;        ///< 暂停服务
```

3.5.3 对象方法定义

3.5.3.1 对象公共参数控制方法

```
ANYCHAT_OBJECT_CTRL_CREATE       =  2;      ///< 创建一个对象
ANYCHAT_OBJECT_CTRL_SYNCDATA     =  3;      ///< 同步对象数据给指定用户，
dwObjectId=-1，表示同步该类型的所有对象
ANYCHAT_OBJECT_CTRL_DEBUGOUTPUT  =  4;      ///< 对象调试信息输出
```

3.5.3.2 服务区域控制方法

```
ANYCHAT_AREA_CTRL_USERENTER      =  401; ///< 进入服务区域
ANYCHAT_AREA_CTRL_USERLEAVE      =  402; ///< 离开服务区域
```

3.5.3.3 队列控制方法

```
ANYCHAT_QUEUE_CTRL_USERENTER  =  501; ///< 进入队列
ANYCHAT_QUEUE_CTRL_USERLEAVE  =  502; ///< 离开队列
```

3.5.3.4 客服服务方法

```
ANYCHAT_AGENT_CTRL_SERVICESTATUS   =  601; ///< 坐席服务状态控制（暂停服务、工作
```

中、关闭）

ANYCHAT_AGENT_CTRL_SERVICEREQUEST = 602; ///< 服务请求

ANYCHAT_AGENT_CTRL_FINISHSERVICE = 604; ///< 结束服务

ANYCHAT_AGENT_CTRL_EVALUATION = 605;///< 服务评价，wParam为客服userid，lParam为评分，lpStrValue为留言

3.5.4 对象异步事件定义

3.5.4.1 对象公共事件定义

ANYCHAT_OBJECT_EVENT_UPDATE = 1; ///< 对象数据更新

ANYCHAT_OBJECT_EVENT_SYNCDATAFINISH = 2; ///< 对象数据同步结束

3.5.4.2 服务区域事件定义

ANYCHAT_AREA_EVENT_STATUSCHANGE = 401; ///< 服务区域状态变化

ANYCHAT_AREA_EVENT_ENTERRESULT = 402; ///< 进入服务区域结果

ANYCHAT_AREA_EVENT_USERENTER = 403; ///< 用户进入服务区域

ANYCHAT_AREA_EVENT_USERLEAVE = 404; ///< 用户离开服务区域

ANYCHAT_AREA_EVENT_LEAVERESULT = 405; ///< 离开服务区域结果

3.5.4.3 队列事件定义

ANYCHAT_QUEUE_EVENT_STATUSCHANGE= 501; ///< 队列状态变化

ANYCHAT_QUEUE_EVENT_ENTERRESULT = 502; ///< 进入队列结果

ANYCHAT_QUEUE_EVENT_USERENTER = 503; ///< 用户进入队列

ANYCHAT_QUEUE_EVENT_USERLEAVE = 504; ///< 用户离开队列

ANYCHAT_QUEUE_EVENT_LEAVERESULT = 505; ///< 离开队列结果

3.5.4.4 座席事件定义

ANYCHAT_AGENT_EVENT_STATUSCHANGE= 601; ///< 坐席状态变化

ANYCHAT_AGENT_EVENT_SERVICENOTIFY= 602; ///< 坐席服务通知（哪个用户到哪个客服办理业务）

ANYCHAT_AGENT_EVENT_WAITINGUSER = 603; ///< 暂时没有客户，请等待

ANYCHAT_AGENT_EVENT_ISREADY = 604 ///< 坐席准备好，可以发起呼叫

3.5.5 服务器信息查询常量定义

ANYCHAT_SERVERQUERY_USERIDBYNAME = 1; ///< 根据用户昵称查询用户 ID
ANYCHAT_SERVERQUERY_USERIDBYSTRID = 2; ///< 根据字符串 ID 查询用户 ID
ANYCHAT_SERVERQUERY_STRIDBYUSERID = 3; ///< 根据用户 ID 查询字符串 ID
ANYCHAT_SERVERQUERY_PPTFILEINFO = 10; ///< PPT 文件信息
ANYCHAT_SERVERQUERY_QUEUEAGENTINFO = 100; ///< 查询指定队列的坐席服务信息
ANYCHAT_SERVERQUERY_RUNNINGSTATUS = 200; ///< 查询服务器运行状态
ANYCHAT_SERVERQUERY_ONLINEUSERS = 201; ///< 查询服务器在线用户数

3.5.6 媒体播放常量定义

3.5.6.1 媒体播放事件类型定义

ANYCHAT_STREAMPLAY_EVENT_START =3; ///< 播放开始事件
ANYCHAT_STREAMPLAY_EVENT_FINISH = 4; ///< 播放结束事件

3.5.6.2 媒体播放标志定义

ANYCHAT_STREAMPLAY_FLAGS_REPLACEAUDIOINPUT = 1; ///< 播放音频流代替本地音频输入（Mic）
ANYCHAT_STREAMPLAY_FLAGS_REPLACEVIDEOINPUT = 2; ///< 播放视频流代替本地视频输入（Camera）

3.5.6.3 媒体播放信息类型定义

ANYCHAT_STREAMPLAY_INFO_JSONVALUE=1; ///< 包含所有播放信息的Json字符串

3.5.6.4 媒体播放控制类型定义

ANYCHAT_STREAMPLAY_CTRL_START = 1; ///< 开始播放
ANYCHAT_STREAMPLAY_CTRL_PAUSE = 2; ///< 暂停播放
ANYCHAT_STREAMPLAY_CTRL_STOP = 3; ///< 停止播放
ANYCHAT_STREAMPLAY_CTRL_OPENLOOP = 6; ///< 打开循环播放
ANYCHAT_STREAMPLAY_CTRL_CLOSELOOP = 7; ///< 关闭循环播放

3.5.7 SDK 控制常量定义

ANYCHAT_SDKCTRL_BASE = 1; ///< 基本功能控制
ANYCHAT_SDKCTRL_OBJECT = 20; ///< 对象操作
ANYCHAT_SDKCTRL_VIDEOCALL = 30; ///< 呼叫控制
ANYCHAT_SDKCTRL_USERINFO = 40; ///< 用户信息控制
ANYCHAT_SDKCTRL_STREAMPLAY = 50; ///< 流媒体播放
ANYCHAT_SDKCTRL_NETWORK = 60; ///< 网络控制
ANYCHAT_SDKCTRL_MEDIA = 70; ///< 媒体控制
ANYCHAT_SDKCTRL_FILEDELETE = 80; ///< 删除文件
ANYCHAT_SDKCTRL_FILEINFO = 81; ///< 获取文件信息
ANYCHAT_SDKCTRL_DISKSIZE = 82; ///< 获取磁盘容量
ANYCHAT_SDKCTRL_FILEENCRYPT = 83; ///< 文件加解密控制
ANYCHAT_SDKCTRL_PPTHELPERINIT = 90; ///< PPT 播报环境初始化
ANYCHAT_SDKCTRL_PPTFILECTRL = 91; ///< PPT 文件控制
ANYCHAT_SDKCTRL_PPTFILEINFO = 92; ///< PPT 文件信息
ANYCHAT_SDKCTRL_BUSINESS = 95; ///< 业务控制

4 接口(Interface)说明及常量定义

AnyChat for Android SDK 通过接口（类似于 C++的回调函数）实现与上层应用的状态更新和数据交互。SDK 的很多调用都是异步的，如登录操作，调用函数完成之后，需要等待对应接口的函数来触发是否登录成功，总体原则是：需要异步操作的地方，都采用接口来实现。

根据不同的类型，接口也分为几大类，在实际的开发过程中，可根据具体情况实现这些接口。

注意：**上层应用设置事件委托时，需要在初始化 SDK 之前完成**，否则可能造成上层应用不能正常响应异步事件，示例代码如下：

```
anychat = new AnyChatCoreSDK();
anychat.SetBaseEvent(this);
anychat.InitSDK(android.os.Build.VERSION.SDK_INT, 0);
```

4.1 基本流程事件接口

“AnyChat 在线音视频互动平台”采用消息通知的方式来告知系统的各种状态变化。

4.1.1 接口定义

```
package com.bairuitech.anychat;
//AnyChat基本事件接口
public interface AnyChatBaseEvent {
	public void OnAnyChatConnectMessage(boolean bSuccess);
	public void OnAnyChatLoginMessage(INT UserId, INT ErrorCode);
	public void OnAnyChatEnterRoomMessage(INT RoomId, INT ErrorCode);
	public void OnAnyChatOnlineUserMessage(INT UserNum, INT RoomId);
	public void OnAnyChatUserAtRoomMessage(INT UserId, boolean bEnter);
	public void OnAnyChatLinkCloseMessage(INT ErrorCode);
}
```

4.1.2 网络连接事件

方法： public void OnAnyChatConnectMessage(boolean bSuccess)

参数：bSuccess 表示是否连接成功，BOOLEAN 类型；

说明:当客户端连接服务器时被触发,等同于 WIN32 平台的 WM_GV_CONNECT 消息。

4.1.3 用户登录事件

方法： public void OnAnyChatLoginMessage(int dwUserId, int dwErrorCode)

参数：

dwUserId 表示自己的用户 ID 号,当 dwErrorCode 为 0 时有效

dwErrorCode 出错代码，可判断登录是否成功

说明：当客户端登录服务器时被触发，等同于 WIN32 平台的 WM_GV_LOGINSYSTEM 消息。

4.1.4 进入房间事件

方法： public void OnAnyChatEnterRoomMessage(int dwRoomId, int dwErrorCode)

参数：

dwRoomId 表示进入的房间 ID 号

dwErrorCode 出错代码，可判断进入房间是否成功

说明：当客户端请求进入房间时被触发，等同于 WIN32 平台的 WM_GV_ENTERROOM 消息。

4.1.5 房间在线用户事件

方法： public void OnAnyChatOnlineUserMessage(int dwUserNum, int dwRoomId)

参数：

dwUserNum 表示当前房间的在线用户数（包含自己）

dwRoomId 房间编号

说明：房间在线用户消息，进入房间后触发一次，等同于 WIN32 平台的 WM_GV_ONLINEUSER 消息。收到该消息后，便可对房间中的用户进行音视频的相关操作，如请求音频、请求视频等。

4.1.6 房间用户活动事件

方法： public void OnAnyChatUserAtRoomMessage(int dwUserId, boolean bEnter)

参数：

dwUserId	表示当前房间活动用户的 ID 号
bEnter	true 表示进入房间，false 表示离开房间

说明：当成功进入房间之后，有新的用户进入房间，或是房间用户离开房间，均会触发该接口，等同于 WIN32 平台的 WM_GV_USERATROOM 消息。

4.1.7 网络连接关闭事件

方法： public void OnAnyChatLinkCloseMessage(int dwReason)

参数：

dwReason	表示连接被关闭的原因

说明：当连接服务器成功之后，网络连接关闭时触发该接口，等同于 WIN32 平台的 WM_GV_LINKCLOSE 消息。如果已打开本地摄像头，则上层应用必须在网络连接关闭事件中关闭本地摄像头，否则可能造成异常。

4.2 用户信息事件接口

用户信息事件接口定义在“AnyChatUserInfoEvent.java”接口类中，包含了用户信息相关的所有事件，包括用户信息更新事件、用户好友上下线等异步事件的通知。

4.2.1 接口定义

```
package com.bairuitech.anychat;
//AnyChat 用户信息事件接口
public interface AnyChatUserInfoEvent {
    // 用户信息更新通知，wParam（INT）表示用户 ID 号，lParam（INT）表示更新类别
    public void OnAnyChatUserInfoUpdate(int dwUserId, int dwType);
    // 好友在线状态变化，wParam（INT）表示好友用户 ID 号，lParam（INT）表示用户的当前活动状态：0 离线， 1 上线
    public void OnAnyChatFriendStatus(int dwUserId, int dwStatus);
}
```

4.2.2 用户信息更新事件

public void OnAnyChatUserInfoUpdate(int dwUserId, int dwType)

参数：

dwUserId 用户 ID 号；

dwType 表示用户信息更新类别；

说明：

当服务器更新用户信息之后，将会触发该消息，通知客户端某个用户的信息已被更新，客户端可以通过 API 接口获取更新后的信息显示在界面上；等同于 WIN32 平台的 WM_GV_USERINFOUPDATE 消息。

4.2.3 好友在线状态变化事件

public void OnAnyChatFriendStatus(int dwUserId, int dwStatus)

参数：

dwUserId 用户 ID 号；

dwStatus 表示用户的当前活动状态：0 离线， 1 上线；

说明：

由服务器通知客户端，当好友上线，或是离线时将会收到通知。在客户端登录成功之后，客户端针对每一个在线好友均会触发一次用户上线消息；等同于 WIN32 平台的 WM_GV_FRIENDSTATUS 消息。

4.3 状态变化事件接口

状态变化事件接口定义在“AnyChatStateChgEvent.java”接口类中，包含了SDK 的用户视频设备状态、音频设备状态、用户聊天模式切换、P2P 连接状态变化等异步事件的通知。

4.3.1 接口定义

```
package com.bairuitech.anychat;
// AnyChat状态变化事件通知接口
public interface AnyChatStateChgEvent {
    public void OnAnyChatMicStateChgMessage(INT UserId, boolean bOpenMic);
    public void OnAnyChatCameraStateChgMessage(INT UserId, INT State);
    public void OnAnyChatChatModeChgMessage(INT UserId, int bPublicChat);
    public void OnAnyChatActiveStateChgMessage(INT UserId, INT State);
    public void OnAnyChatP2PConnectStateMessage(INT UserId, INT State);
}
```

4.3.2 音频设备状态改变事件

public void OnAnyChatMicStateChgMessage(int dwUserId, boolean bOpenMic)

参数：

dwUserId	表示状态变化的用户 ID
bOpenMic	表示该用户是否已打开音频采集设备

说明：当进入房间成功之后，当用户使用 API：UserSpeakControl 操作本地音频设备时将会触发该接口，等同于 WIN32 平台的 WM_GV_MICSTATECHANGE 消息。

4.3.3 视频设备状态改变事件

public void OnAnyChatCameraStateChgMessage(int dwUserId, int dwState)

参数：

dwUserId	表示状态变化的用户 ID
dwState	表示该用户当前的视频设备状态：

0 没有摄像头设备

1 有摄像头设备，但没有打开

2 已打开摄像头设备

说明：当进入房间成功之后，当用户使用 API：UserCameraControl 操作本地视频设备时将会触发该接口，等同于 WIN32 平台的 WM_GV_CAMERASTATE 消息。

4.3.4 用户聊天模式改变事件

public void OnAnyChatChatModeChgMessage(int dwUserId, boolean bPublicChat)

参数：

dwUserId 表示状态变化的用户 ID

bPublicChat 表示该用户当前是否为公聊状态，否则为私聊状态：

说明：当进入房间成功之后，当用户改变聊天模式时将会触发该接口，等同于 WIN32 平台的 WM_GV_CHATMODECHG 消息。

4.3.5 用户 P2P 连接改变事件

public void OnAnyChatP2PConnectStateMessage(int dwUserId, int dwState)

参数：

dwUserId 表示其它用户 ID 号

dwState 表示本地用户与其它用户的当前 P2P 网络连接状态：

0 没有任何连接

1 P2P 连接成功，TCP 连接

2 P2P 连接成功，UDP 连接

3 P2P 连接成功，TCP 与 UDP 连接

说明：当进入房间成功之后，与其它用户建立 P2P 连接，或是 P2P 连接被断开时触发该接口，等同于 WIN32 平台的 WM_GV_P2PCONNECTSTATE 消息。

4.3.6 用户视频大小改变事件

public void OnAnyChatVideoSizeChgMessage(int dwUserId, int dwWidth, int dwHeight)

参数：

dwUserId	表示状态变化的用户 ID
dwWidth	表示该用户当前的视频宽度
dwHeight	表示该用户当前的视频高度

说明：当进入房间成功之后，成功打开本地视频设备，或是修改视频设备采集分辨率之后将触发该接口，等同于 WIN32 平台的 WM_GV_VIDEOSIZECHG 消息。

4.4 数据传输事件接口

4.4.1 接口定义

```
package com.bairuitech.anychat;
// 数据传输通知接口
public interface AnyChatTransDataEvent {
    public void OnAnyChatTransFile(int dwUserid, String FileName, String TempFilePath, int dwFileLength, int wParam, int lParam, int dwTaskId);
    public void OnAnyChatTransBuffer(int dwUserid, byte[] lpBuf, int dwLen);
    public void OnAnyChatTransBufferEx(int dwUserid, byte[] lpBuf, int dwLen, int wparam, int lparam, int taskid);
    public void OnAnyChatSDKFilterData(byte[] lpBuf, int dwLen);
}
```

4.4.2 收到文件传输数据事件

public void OnAnyChatTransFile(int dwUserid, String FileName, String TempFilePath, int dwFileLength, int wParam, int lParam, int dwTaskId);

参数：

dwUserid：	用户 ID，指示发送用户
FileName：	文件名（含扩展名，不含路径）

TempFilePath： 接收完成后，SDK 保存在本地的临时文件（包含完整路径）

dwFileLength： 文件总长度

wParam： 附带参数 1

lParam： 附带参数 2

dwTaskId： 该文件所对应的任务编号

说明：

当收到其它用户使用“TransFile”方法发送的文件时，将会触发该接口，等同于回调函数：BRAC_TransFile_CallBack。

特别提示：本 SDK 不会删除“lpTempFilePath”所指示的临时文件，上层应用在处理完毕后，需要主动删除该临时文件。

4.4.3 收到透明通道数据事件

public void OnAnyChatTransBuffer(int dwUserid, byte[] lpBuf, int dwLen);

参数：

dwUserid： 用户 ID，指示发送用户

lpBuf： 缓冲区地址

dwLen： 缓冲区大小

说明：

当收到其它用户使用“TransBuffer””方法发送的缓冲区数据时，将会触发该接口，等同于回调函数：BRAC_TransBuffer_CallBack。

由于该函数传递的数据是一个与本 SDK 无关的缓冲区（由上层应用自己填充内容），相对于本 SDK 来说是透明的，故称为透明通道，利用该通道，可以向当前房间内的任何用户传输上层应用自定义的数据。

4.4.4 收到扩展透明通道数据事件

public void OnAnyChatTransBufferEx(int dwUserid, byte[] lpBuf, int dwLen, int wparam, int lparam, int taskid);

参数：

dwUserid： 用户 ID，指示发送用户

lpBuf： 缓冲区地址

dwLen： 缓冲区大小

wParam： 缓冲区附带参数（由发送者设置，上层应用可自定义用途）

lParam： 缓冲区附带参数 2

dwTaskId： 该缓冲区所对应的传输任务编号

说明：

当收到其它用户使用“TransBufferEx”方法发送的缓冲区数据时，将会触发该接口，等同于回调函数：BRAC_TransBufferEx_CallBack。

4.4.5 收到 SDK Filter 数据事件

public void OnAnyChatSDKFilterData(byte[] lpBuf, int dwLen);

参数：

lpBuf： 缓冲区地址

dwLen： 缓冲区大小

说明：

当收到服务器“SDK Filter”或是“Server SDK”相关接口发送的缓冲区数据时，将会触发该接口，等同于回调函数：BRAC_SDKFilterData_CallBack。

建议采用“透明通道”API 接口来处理服务器与客户端之间的缓冲区数据传输，参考：如何使用缓冲区及文件传输功能？

4.5 文字消息事件接口

4.5.1 接口定义

```
package com.bairuitech.anychat;
// 文字聊天通知接口
public interface AnyChatTextMsgEvent {
    public void OnAnyChatTextMessage(int dwFromUserid, int dwToUserid, int bSecret, String
message);
}
```

4.5.2 收到文字聊天消息事件

public void OnAnyChatTextMessage(int dwFromUserid, int dwToUserid, boolean bSecret, String message);

参数：

dwFromUserid	消息发送者用户 ID
dwToUserid	目标用户，-1 表示发送给大家，即房间所有人
bSecret	是否为悄悄话，当目标用户不为-1 时有效
message	消息字符串

说明：当进入房间成功之后，收到其他用户发送的文字聊天信息时将触发该接口，等同于 WIN32 平台的回调函数：BRAC_TextMessage_CallBack。本地用户向其它用户发送文字消息时，将不会触发该接口。

4.6 视频呼叫事件接口

4.6.1 接口定义

```
package com.bairuitech.anychat;
// AnyChat 视频呼叫事件通知接口
public interface AnyChatVideoCallEvent {
    public void OnAnyChatVideoCallEvent(int dwEventType, int dwUserId, int dwErrorCode, int dwFlags, int dwParam, String userStr);
}
```

4.6.2 视频呼叫事件

public void OnAnyChatVideoCallEvent(int dwEventType, int dwUserId, int dwErrorCode, int dwFlags, int dwParam, String userStr)

参数：

dwEventType 呼叫事件类型，详见函数 BRAC_VideoCallControl 中的定义

dwUserId： 视频呼叫事件发起方用户 ID

dwErrorCode： 错误代码，当事件类型为“Reply”和“Finish”时有效

dwFlags： 视频呼叫标志

dwParam： 事件附带参数（整型）

userStr： 事件附带参数（字符串）

备注：

当实现该接口事件后，其它用户通过 API：BRAC_VideoCallControl 发起视频呼叫时，将触发该事件接口。

用户 A 向用户 B 发送（Request）请求，用户 B 回复（Reply）同意通话之后，服务器会自动向 A、B 同时发送（Start）指令，表示会话开始， 当客户端在回调函数中收到 dwEventType= BRAC_VIDEOCALL_EVENT_START 事件时，dwParam 表示 RoomId，由服务器自动分配，这时用户 A、B 均需要主动进入分配的房间，打开本地音频、视频，同时请求对方的音频、视频才能完成整个视频呼叫过程。

更多关于视频呼叫事件的信息可参考技术论坛相关内容：http://bbs.anychat.cn/forum.php?mod=viewthread&tid=150&extra=page%3D1

4.7 数据加密、解密事件接口

4.7.1 接口定义

```
package com.bairuitech.anychat;
import com.bairuitech.anychat.AnyChatOutParam;
// 数据加密、解密接口
public interface AnyChatDataEncDecEvent {
    public int OnAnyChatDataEncDec(int userid, int flags, byte[] lpBuf, int dwLen, AnyChatOutParam outParam);
}
```

4.7.2 数据加密、解密事件

public int OnAnyChatDataEncDec(int userid, int flags, byte[] lpBuf, int dwLen, AnyChatOutParam outParam)

参数：

userid： 数据来源用户 ID

flags： 数据标志，包括加密、解密以及数据类型等信息

lpBuf： 输入缓冲区（字节数组）

dwLen： 输入缓冲区数据大小

outParam： 输出缓冲区

备注：

内核默认情况下没有开始数据加密、解密功能，若需要触发该回调，则需要调用如下 API 接口开启：

```
AnyChatCoreSDK.SetSDKOptionInt(AnyChatDefine.BRAC_SO_CORESDK_DATAENCRYPTION, 1);
```

需要对数据进行加密或是解密处理，可通过 dwFlags 中的标志位区分，同时还可以区分数据类型，是视频数据，或是音频数据等；

在数据加密（或解密）完成之后，通过输出缓冲区（OutParam）直接将处理后的数据拷贝到输出缓冲区；

更多关于数据加密、解密的信息可参考技术论坛相关内容：

http://bbs.anychat.cn/forum.php?mod=viewthread&tid=674&extra=page%3D1

4.8 录像、拍照事件接口

4.8.1 接口定义

```
package com.bairuitech.anychat;
// AnyChat 视频录制事件通知接口
public interface AnyChatRecordEvent {
    // 视频录制完成事件
    public void OnAnyChatRecordEvent(int dwUserId, String lpFileName, int dwElapse, int dwFlags, int dwParam, String lpUserStr);
    // 图像抓拍完成事件
```

public void OnAnyChatSnapShotEvent(int dwUserId, String lpFileName, int dwFlags, int dwParam, String lpUserStr);
}

4.8.2 视频录制完成事件

public void OnAnyChatRecordEvent(int dwUserId, String lpFileName, int dwElapse, int dwFlags, int dwParam, String lpUserStr);

参数：

dwUserId： 被录制用户 ID

lpFileName： 文件保存路径

dwElapse： 录像时长，单位：秒

dwFlags： 录像标志

dwParam： 用户自定义参数，整型

lpUserStr： 用户自定义参数，字符串类型

备注：

客户端调用 API：StreamRecordCtrlEx 录像完成之后，将会触发该回调事件，其中 lpFileName 为本地文件路径；

用户自定义参数包括整型（dwParam）、字符串类型（lpUserStr）与 API：StreamRecordCtrlEx 的传入参数对应；

进行中心服务器录像时，也可以触发客户端的本地回调函数，更多信息可参考：中心服务器录像支持触发客户端回调事件、AnyChat 音视频录制整体解决方案

4.8.3 图像抓拍完成事件

public void OnAnyChatSnapShotEvent(int dwUserId, String lpFileName, int dwFlags, int dwParam, String lpUserStr);

参数：

dwUserId： 被拍照用户 ID

lpFileName： 文件保存路径

dwFlags： 拍照标志

dwParam： 用户自定义参数，整型

lpUserStr： 用户自定义参数，字符串类型

备注：

客户端调用 API：SnapShot 完成图像抓拍之后，将会触发该回调事件，其中 lpFileName 为本地文件路径。

4.9 业务对象事件的接口

```
package com.bairuitech.anychat;

// AnyChat业务对象事件通知接口
public interface AnyChatObjectEvent {
//业务对象回调事件
    public void OnAnyChatObjectEvent(int dwObjectType, int dwObjectId, int dwEventType, int dwParam1, int dwParam2, int dwParam3, int dwParam4, String strParam);
}
```

4.9.1 业务对象事件接口

public void OnAnyChatObjectEvent(int dwObjectType, int dwObjectId, int dwEventType, int dwParam1, int dwParam2, int dwParam3, int dwParam4, String strParam);

参数：

dwObjectType：业务对象类型，见 3.5.1.1 章节描述内容

dwObjectId： 业务对象的 ID

dwEventType： 业务对象事件回调类型，见3.5.4章节描述内容

dwParam1： 用户自定义参数，整型

dwParam2： 用户自定义参数，整型

dwParam3： 用户自定义参数，整型

dwParam4： 用户自定义参数，整型

strParam： 用户自定义参数，字符串型

备注：

当注册该回调函数后，其它用户通过 API：ObjectControl 发起操作业务对象方法时，将触发该回调函数。

5 函数说明

5.1 初始化与资源释放

5.1.1 初始化 SDK

int InitSDK(int osver, int flags)

功能：初始化 SDK

返回值：0 表示成功，否则为出错代码

参数：

osver Android 平台 API 版本，系统常量：android.os.Build.VERSION.SDK_INT

flags 功能模式组合，默认为 0，参考 WIN32 平台 SDK 的 InitSDK 的参数 dwFuncMode 定义

备注：

功能模式组合可根据实际的需求灵活定义，如果在后续的方法调用中失败，则很有可能是某一项功能没有被定义，默认为 0，SDK 内部会自动设置常用的标志。

该方法必须第一个被调用（SetSDKOptionString 方法除外），否则后续的其它方法调用将会返回没有初始化错误。

5.1.2 设置基本事件通知接口

public void SetBaseEvent(AnyChatBaseEvent e);

功能：只有设置基本事件通知接口后，AnyChat 内核产生的异步事件才能通知到 Activity，并触发 Activity 中对应的接口方法。

返回值：无

参数：

E 实现 AnyChatBaseEvent 接口的对象

备注：

如有多个 Activity 需要响应事件，则每个 Activity 都必须实现对应的接口，并调用该方法向 AnyChat 内核注册该接口，只有当前活动的 Activity 才能接收到异步消息，触发接口方法。

5.1.3 设置状态变化事件接口

public void SetStateChgEvent(AnyChatStateChgEvent e);

功能：只有设置状态变化事件通知接口后，AnyChat 内核产生的状态变化事件才能通知到 Activity，并触发 Activity 中对应的接口方法。

返回值：无

参数：

E 实现 AnyChatStateChgEvent 接口的对象

备注：

如有多个 Activity 需要响应事件，则每个 Activity 都必须实现对应的接口，并调用该方法向 AnyChat 内核注册该接口，只有当前活动的 Activity 才能接收到异步消息，触发接口方法。

5.1.4 设置私聊消息通知接口

public void SetPrivateChatEvent(AnyChatPrivateChatEvent e);

功能：只有设置私聊事件通知接口后，AnyChat 内核产生的私聊事件才能通知到 Activity，并触发 Activity 中对应的接口方法。

返回值：无

参数：

E 实现 AnyChatPrivateChatEvent 接口的对象

备注：

如有多个 Activity 需要响应事件，则每个 Activity 都必须实现对应的接口，并调用该方法向 AnyChat 内核注册该接口，只有当前活动的 Activity 才能接收到

异步消息，触发接口方法。

5.1.5 设置文字聊天消息接口

public void SetTextMessageEvent(AnyChatTextMsgEvent e);

功能：只有设置文字聊天消息接口后，AnyChat 内核产生的文字聊天事件才能通知到 Activity，并触发 Activity 中对应的接口方法。

返回值：无

参数：

E 实现 AnyChatTextMsgEvent 接口的对象

备注：

如有多个 Activity 需要响应事件，则每个 Activity 都必须实现对应的接口，并调用该方法向 AnyChat 内核注册该接口，只有当前活动的 Activity 才能接收到异步消息，触发接口方法。

5.1.6 设置数据传输消息接口

public void SetTransDataEvent(AnyChatTransDataEvent e);

功能：只有设置数据传输消息接口后，AnyChat 内核产生的数据传输事件才能通知到 Activity，并触发 Activity 中对应的接口方法。

返回值：无

参数：

E 实现 AnyChatTransDataEvent 接口的对象

备注：

如有多个 Activity 需要响应事件，则每个 Activity 都必须实现对应的接口，并调用该方法向 AnyChat 内核注册该接口，只有当前活动的 Activity 才能接收到异步消息，触发接口方法。

数据传输接口内部实现了缓冲区传输、扩展缓冲区传输、文件传输等传输数据的异步事件通知接口。

5.1.7 设置业务对象接口

public void SetObjectEvent(AnyChatObjectEvent e)

功能：只有设置业务对象接口后，AnyChat 内核产生的业务对象事件才能通知到 Activity，并触发 Activity 中对应的接口方法。

返回值：无

参数：

E 实现 AnyChatTransDataEvent 接口的对象

备注：

如有多个 Activity 需要响应事件，则每个 Activity 都必须实现对应的接口，并调用该方法向 AnyChat 内核注册该接口，只有当前活动的 Activity 才能接收到异步消息，触发接口方法。

5.1.8 释放 SDK 资源

int Release();

功能：释放 SDK 占用的所有资源

返回值：0 表示成功，否则为出错代码

参数：

无。

备注：

该方法必须最后一个被调用，调用该方法后，SDK 内部所占用的资源将被释放，如果在其后面再调用其它的方法，将会返回没有初始化的错误。

该方法通常在上层应用退出系统时被调用，即最后一个 Activity 销毁时调用，而中间 Activity 销毁时不需要，也不能调用该方法。

5.2 登录流程

5.2.1 设置服务器认证密码

int SetServerAuthPass(String Password);

功能：设置服务器连接认证密码，确保 SDK 能正常连接到服务器。

返回值：0 表示成功，否则为出错代码

参数：

lpPassword 认证密码（大小写敏感）；

备注：

为了防止未授权 SDK 连接服务器，在服务器配置文件（AnyChatCoreServer.ini）中可设置“SDKAuthPass”，如果该配置项被设置，当 SDK 连接服务器时，会将该方法所传入的密码加密后传输到服务器，服务器再比较是否合法，如果密码不正确，则连接将被断开。如果该配置项未被设置（配置文件默认），则无论该方法是否被调用，SDK 均可正常连接到服务器。

5.2.2 连接服务器

int Connect(String ServerAddr, int Port);

功能：用于与服务器建立连接。

返回值：0 表示成功，否则为出错代码

参数：

lpServerAddr 服务器 IP 地址，或是网站域名（URL）地址；

dwPort 服务端口号（默认为 8906）

备注：

返回值为 0 并不表示连接服务器成功，仅表示 SDK 已成功收到连接服务器的指令，如果连接成功，或是失败，都将会通过相应的接口通知上层应用，这里是一个异步的过程。

5.2.3 登录系统

int Login(String UserName, String Password)

功能：登录服务器，请求身份认证。

返回值：0 表示成功，否则为出错代码

参数：

lpUserName 注册用户名；

lpPassword 登录密码（为空表示游客）；

备注：

该方法可以连接系统之后立即调用，而不用关心连接系统是否成功，当 SDK 连接系统成功之后，如果之前调用过该方法，则 SDK 将会自动向服务器发出登录系统的申请。

返回值为 0 并不表示登录服务器成功，仅表示 SDK 已成功收到登录服务器的指令，如果登录成功，或是失败，都将会通过相应的接口通知上层应用，这里是一个异步的过程。

如果服务器配置了“SDK Filter Plus”插件，则客户端调用该方法后，将会触发其 API 接口：BRFP_VerifyUser，用户名、密码参数将会作为参数传递给该 API 函数，由“SDK Filter Plus”完成用户的身份验证工作，服务器根据该 API 接口的返回值来判定是否通过身份验证，详细信息可参考文档《AnyChat SDK Filter Plus 开发指南》。

如果在服务器端使用“AnyChat Server SDK”开发了业务层服务器，则客户端调用该方法后，将会触发业务层服务器的回调函数“BRAS_VerifyUser_CallBack”，由业务层服务器完成用户的身份验证工作，服务器根据回调函数的返回值来判定是否通过身份验证，详细信息可参考文档《AnyChat Server SDK 开发指南》。

int LoginEx(String lpNickName,int dwUserId, String lpStrUserId, String lpStrAppId, int dwTimeStamp, String lpSigStr, String lpStrParam)

功能： 登录云平台的接口，用户进行签名后调用此接口登录到云平台；

参数：

lpNickName 字符串值，用户显示名称

dwUserId 整形值，用户 Id 值，如果应用没有此参数，则传入-1

lpStrUserId 字符串值，用户编号，如果 dwUserId 参数有值，则此参数值可为传空字符串；如果 dwUserId 为-1，则需要传此参数值

dwTimeStamp 整型值，签名的时间戳，由签名工具返回

lpStrAppId 字符串值，在集群版本、视频云平台申请的应用 Id

lpSigStr 字符串值，使用应用的公钥和私钥进行签名后生成的签名字符串

lpStrParam 字符串值，预留参数，传空字符串

返回值：0 表示成功，否则为出错代码；

说明：

此接口用于用户在自有的或第三方的身份验证系统验证之后，根据获取的应用公钥和设置的私钥数据，再调用 AnyChat 提供的或自己编写的身份签名工具对用户进行身份签名。签名后再调用此接口进行登陆；通过该接口后则不需要再由 AnyChat 业务服务器进行身份验证了。

如果在系统或应用中设置了允许用户以游客身份进行登陆，则该接口也可以不用验证用户身份签名，允许用户登录系统。

5.2.4 视频呼叫控制

int VideoCallControl(int dwEventType, int dwUserId, int dwErrorCode, int dwFlags, int dwParam, String lpUserStr);

功能：对视频呼叫业务流程进行控制，发起视频呼叫，或是结束视频通话等。

返回值：0 表示成功，否则为出错代码

参数：

dwEventType 事件类型，定义为：

```
#define BRAC_VIDEOCALL_EVENT_REQUEST1     ///< 呼叫请求
#define BRAC_VIDEOCALL_EVENT_REPLY    2   ///< 呼叫请求回复
```

```
#define BRAC_VIDEOCALL_EVENT_START    3    ///< 视频呼叫会话开始事件
#define BRAC_VIDEOCALL_EVENT_FINISH   4    ///< 挂断（结束）呼叫会话
```

dwUserId	目标用户 ID
dwErrorCode	出错代码，与事件类型相关，默认为 0
dwFlags	视频呼叫标志，默认为 0
dwParam	用户自定义参数（整型），默认为 0
lpUserStr	用户自定义参数（字符串），默认为空

备注：

视频呼叫业务逻辑主要实现两个终端（PC、手机、Pad 等）之间的通话请求流程控制，包括请求（Request）、回复（Reply）、开始（Start）以及结束（Finish）等过程，可以形象理解为打电话的流程：拨号、等待、通话、挂断。为 V1.9 版本新增接口。

该 API 接口和回调函数（BRAC_VideoCallEvent_CallBack）中的 dwUserId 均为对方（被呼叫方）的用户 ID；

被呼叫方拒绝通话时，可通过发送发送回复（Reply）指令，其中 dwErrorCode=100104 来拒绝；

被呼叫方同意通话时，发送回复（Reply）指令，其中 dwErrorCode=0，然后服务器会向双方发送通话开始（Start）指令；

用户 A 向用户 B 发送（Request）请求，用户 B 回复（Reply）同意通话之后，服务器会自动向 A、B 同时发送（Start）指令，表示会话开始，当客户端在回调函数中收到 dwEventType= BRAC_VIDEOCALL_EVENT_START 事件时，dwParam 表示 RoomId，由服务器自动分配，这时用户 A、B 均需要主动进入分配的房间，打开本地音频、视频，同时请求对方的音频、视频才能完成整个视频呼叫过程。

结束通话时，任何一方（包括业务服务器）均可以发送结束（Finish）指令，然后服务器会向双方发送通话结束（Finish）指令；

业务服务器可干预呼叫流程：在业务服务器端回调函数（BRAS_OnVideoCallEvent_CallBack）收到呼叫请求指令后，返回 0 表示允许呼叫，否则为出错代码，不允许呼叫；在会话过程中业务服务器可以发送结束（Finish）指令，强制挂断指定用户的通话；

API 接口中的 dwParam（整型）、lpUserStr（字符串）均为用户自定义用途；

一个用户同时只能发起一路呼叫请求，也同时只能被一个用户呼叫；

更多关于视频呼叫事件的信息可参考技术论坛相关内容：

http://bbs.anychat.cn/forum.php?mod=viewthread&tid=150&extra=page%3D1

5.3 进入房间（根据房间编号）

int EnterRoom(int Roomid, String RoomPass);

功能：根据房间编号进入房间

返回值：0 表示成功，否则为出错代码

参数：

dwRoomid 房间编号，系统唯一；

lpRoomPass 房间密码（当房间需要密码时有效，如果没有可为空）；

备注：

该方法可以登录系统之后立即调用，而不用关心登录系统是否成功，当 SDK 登录系统成功之后，如果之前调用过该方法，则 SDK 将会自动向服务器发出进入房间的申请。

返回值为 0 并不表示进入房间成功，仅表示 SDK 已成功收到进入房间的指令，不论成功，或是失败，都将会通过相应的接口通知上层应用，这里是一个异步的过程。

用户必须进入一个房间，否则无法进行相关的操作，后续用户所有的操作都是在房间内操作，针对游戏，房间可以理解为游戏桌（一桌游戏对应一个房间），针对视频会议，房间可以理解为会议室。

如果服务器配置了“SDK Filter Plus”插件，则客户端调用该方法后，将会触发其 API 接口：BRFP_PrepareEnterRoom，用户 ID、房间 ID、房间密码将会作为参数传递给该 API 函数，由“SDK Filter Plus”完成用户进入房间的验证工作，服务器根据该 API 接口的返回值来判定是否允许进入房间，详细信息可参考文档

《AnyChat SDK Filter Plus 开发指南》。

如果在服务器端使用“AnyChat Server SDK”开发了业务层服务器，则客户端调用该方法后，将会触发业务层服务器的回调函数“BRAS_PrepareEnterRoom_CallBack”，由业务层服务器完成用户进入房间的验证工作，服务器根据该 API 接口的返回值来判定是否允许进入房间，详细信息可参考文档《AnyChat Server SDK 开发指南》。

5.3.1 进入房间（根据房间名称）

int EnterRoomEx(String RoomName, String RoomPass)

功能：根据房间名称进入房间

返回值：0 表示成功，否则为出错代码

参数：

lpRoomName 房间名称；

lpRoomPass 房间密码（当房间需要密码时有效，如果没有可为空）；

备注：

该方法与“EnterRoom”功能相同，区别在于房间的标识不同，其中“EnterRoom”是用房间 ID 进入房间，而该方法是用房间名称进入房间，如果房间不存在，而且系统配置为自动创建房间时，将会由系统分配一个唯一的房间编号，通过基本事件接口（AnyChatBaseEvent）返回给上层应用，上层应用可以通过方法“GetRoomName”来获取房间名称。

5.3.2 离开房间

int LeaveRoom(int Roomid);

功能：离开房间。

返回值：0 表示成功，否则为出错代码

参数：

Roomid 房间编号，为-1 表示退出当前房间

备注：

在用户变换房间之前，需要调用该方法离开房间，然后才能进入新的房间。

5.3.3 注销系统

int Logout();

功能：将用户从系统中注销。

返回值：0 表示成功，否则为出错代码

参数：

无

备注：

在切换用户（如用户换用其它的用户名登录系统）时需要先调用该方法，或是在退出系统前需要调用该方法

5.4 音视频操作

5.4.1 用户视频控制

int UserCameraControl(int Userid, int bOpen);

功能：用户视频控制，打开或关闭本地摄像头，或请求对方的视频

返回值：0 表示成功，否则为出错代码

参数：

dwUserid： 用户编号，为-1 表示对本地视频进行控制

bOpen 1 是打开视频,0 是关闭

备注：

对于本地用户，该方法是直接操作用户的摄像头，而对于其它用户，该方法只是向对方发送一个请求（取消）视频流的申请，并不会直接操作对方的摄像头。

5.4.2 用户语音控制

int UserSpeakControl(int Userid, int bOpen);

功能：用户发言控制

返回值：0 表示成功，否则为出错代码

参数：

dwUserid 用户编号，为-1 表示对本地发言进行控制

bOpen 是否允许用户发言，当 dwUserid=-1 时，1 表示请求发言（拿 Mic），0 表示停止发言（放 Mic）

备注：

对于本地用户，该方法是直接操作用户的 Mic，而对于其它用户，该方法只是向对方发送一个请求（取消）音频流的申请，并不会直接操作对方的 Mic。

5.4.3 设置视频显示位置

int SetVideoPos(int userid, Surface s, int Left, int Top, int Right, int Bottom);

功能：设置视频显示位置，或是刷新视频显示

返回值：0 表示成功，否则为出错代码

参数：

userid 用户编号，为-1 表示操作自己的视频显示位置

s 视频显示表面

dwLeft、dwTop、dwRight、dwBottom 位置信息，默认为 0

备注：

该方法在打开用户视频之前调用，调用之前需在 java 层设置该表面的视频大小（SetFixedSize(w,h)），用户视频大小可通过方法 GetUserVideoWidth、GetUserVideoHeight 获取。

5.5 查询状态

5.5.1 查询摄像头的状态

int GetCameraState(int Userid);

功能：查询用户摄像头的状态

返回值：返回指定用户的摄像头状态，定义为：

0 没有摄像头

1 有摄像头但没有打开

2 摄像头已打开

参数：

userid 用户编号，为-1 时表示获取自己的摄像头状态；

备注：

该方法必须在登录系统之后调用方才有效，根据返回参数的不同，可以判别用户当前摄像头的状态，以及判断用户是否有摄像头。

5.5.2 查询用户音频设备采集状态

int GetSpeakState(int userid);

功能：查询用户音频设备采集状态

返回值：返回指定用户的音频设备状态，定义为：

0 音频采集关闭

1 音频采集开启

参数：

userid 用户编号，为-1 时表示获取自己的音频设备状态；

备注：

这里所说的“音频设备采集状态”是指在 SDK 内部是否已开始音频采集，当返回值为 1 时，表示 SDK 已经开始采集，当有其它用户请求时，才对外传输。

关于实际应用中的“公麦”、“麦序”等属于业务逻辑范畴，具体的实现方式可

参考《AnyChat Server SDK 开发指南》中“常用业务处理逻辑”的章节。

5.5.3 查询用户昵称

String GetUserName(int userid);

功能：查询用户昵称

返回值：用户昵称字符串

参数：

userid 用户编号，为-1 时表示获取自己的昵称；

备注：

这里所查询到的用户昵称，是用户在身份验证时，服务器端调用 SDK Filter 的“BRGS_VerifyUser”方法时，由 SDK Filter 返回给服务器的 lpNickName 参数值，如果 lpNickName 为空，则默认采用登录用户名替代用户昵称。

当用户离开房间之后（包括在 WM_GV_USERATROOM 消息中，状态为用户离开时）将会查询失败。

5.5.4 查询用户 IP 地址

String GetUserIPAddr (int userid);

功能：查询用户互联网 IP 地址

返回值：用户 IP 字符串

参数：

userid 用户编号，为-1 时表示获取自己的 IP 地址；

备注：

这里所查询到的用户 IP 为互联网 IP 地址，可能是用户本机的真实 IP，也可能是用户接入互联网的网关 IP 地址。

5.5.5 查询用户视频宽度

int GetUserVideoWidth (int userid);

功能：查询用户视频分辨率的宽度值

返回值：返回指定用户的视频宽度，如果用户视频没有打开，或是打开视频设备失败，则获取的值为 0

参数：

userid　用户编号，为-1 时表示获取自己的视频宽度；

备注：

如查询本地的视频宽度值，则必须在打开本地视频设备成功之后方能查询成功；如查询其它用户的视频宽度值，则必须在对方打开视频设备成功，且状态同步到本地后，才能查询成功，故上层应用可用一个定时器间隔查询。

5.5.6 查询用户视频高度

int GetUserVideoHeight (int userid);

功能：查询用户视频分辨率的高度值

返回值：返回指定用户的视频高度，如果用户视频没有打开，或是打开视频设备失败，则获取的值为 0

参数：

userid　用户编号，为-1 时表示获取自己的视频高度；

备注：

如查询本地的视频高度值，则必须在打开本地视频设备成功之后方能查询成功；如查询其它用户的视频高度值，则必须在对方打开视频设备成功，且状态同步到本地后，才能查询成功，故上层应用可用一个定时器间隔查询。

5.5.7 查询用户状态（字符串）

String QueryUserStateString(int userid, int infoname);

功能：查询指定用户状态（字符串类型）

返回值：相关状态的字符串

参数：

dwUserId	用户编号，可用-1 代表本地用户（自己）；
infoname	需要查询的信息代码（见 WIN32 平台 SDK 相关定义）

备注：

通过调用该方法，可以查询指定用户的相关状态值。

5.5.8 查询用户状态（整型）

int QueryUserStateInt(int userid, int infoname);

功能：查询指定用户状态（整型）

返回值：相关状态值

参数：

dwUserId 用户编号，可用-1 代表本地用户（自己）；

infoname 需要查询的信息代码（见 WIN32 平台 SDK 相关定义）

备注：

通过调用该方法，可以查询指定用户的相关状态值。

5.5.9 查询房间名称

String GetRoomName(int roomid);

功能：查询根据房间 ID 获取房间名称

返回值：返回房间名称，如查询失败，则返回空字符串

参数：

roomid 房间编号；

备注：

目前只能查询当前所在房间的房间名称，当用户离开房间后，查询将会失败。

5.5.10 向服务器动态查询相关信息

String QueryInfoFromServer(int dwInfoName, String strInParam, int dwFlags);

功能： 查询服务器信息

返回值： 返回要查询的服务器信息。

参数：

dwInfoName： 整型值， 需要查询的信息代码（见 3.5.5 章节描述内容）

lpInParam： 查询信息所依据的值

dwFlags 查询标志，默认为 0

5.6 普通功能

5.6.1 获取 SDK 版本信息

public int GetSDKMainVersion();

功能：获取 SDK 主版本号。

返回值：主版本号

参数：无

public int GetSDKSubVersion ();

功能：获取 SDK 从版本号。

返回值：从版本号

参数：无

public String GetSDKBuildTime ();

功能：获取 SDK 编译时间字符串。

返回值：编译时间字符串

参数：无

5.6.2 获取当前房间在线用户列表

int[] GetOnlineUser();

功能：获取当前房间在线用户列表（不包含自己）

返回值：在线用户 ID 数组

参数：

无。

备注：

获取在线用户列表，并不包含当前用户自己的 ID，自己的 ID 在登录事件（AnyChatBaseEvent）中已通知给上层应用。

5.6.3 查询指定房间在线用户列表

函数：int[] GetRoomOnlineUsers(int dwRoomid);

功能：获取指定房间在线用户列表

返回值：返回房间在线用户 ID 数组

参数：

dwRoomid 整形值，房间编号；

备注：

获取指定房间在线用户列表，接口调用后会返回在线用户 ID 数组。

5.6.4 传送文本消息

int SendTextMessage(int userid, boolean secret, String message);

功能：向指定的用户传送文本消息

返回值：0 表示成功，否则为出错代码

参数：

dwUserid： 目标用户编号，-1 表示大家（所有人）

secret： 是否为密语，只在 dwUserid 不为-1 时有效，选择密语时，其它用户看不到发送的消息

message： 消息字符串

备注：

可以利用该消息实现文字交流的功能，发送消息的对象可以是大家，也可以是指定的对象，如果是对指定的对象发送文字消息，可以选择密语。

对方收到该消息后，会触发 AnyChatTextMsgEvent 接口的接口函数的调用。

5.6.5 透明通道传送缓冲区

int TransBuffer(int userid, byte[] buf, int len);

功能：透明通道传送缓冲区

返回值：0 表示成功，否则为出错代码

参数：

dwUserid： 目标用户编号，-1 表示大家（用户当前房间所有人）

buf： 缓冲区

len： 缓冲区的大小

备注：

可以利用该方法实现自定义功能，缓冲区采用透明传输，目标对象可以是大家，也可以是指定的对象。

该方法将会触发对方的 AnyChatTransDataEvent 接口的对应接口函数。

该 API 方法支持跨房间传输缓冲区数据，目标用户为指定用户时，目标用户可以与自己在不同的房间，或是目标用户没有进入任何房间，或是源用户（自己）没有进入任何房间，只要双方都登录服务器成功，则可利用该方法传输缓冲区，当目标用户编号为-1 时，则源用户（自己）必须已经在房间中，表示向该房间的其它用户广播数据（注：自己不能发送给自己）。

5.6.6 透明通道传送缓冲区扩展

int TransBufferEx(int userid, byte[] buf, int len, int wparam, int lparam, int flags, AnyChatOutParam outParam);

功能：透明通道传送缓冲区

返回值：0 表示函数调用成功，否则为出错代码

参数：

userid： 目标用户编号，只针对某一个用户，不能为-1（所有人）

buf： 缓冲区，内部会自动分包处理

len： 缓冲区的大小

wParam： 附带参数，由上层应用自定义

lParam： 附带参数 2，由上层应用自定义

dwFlags： 特殊功能标志，当对该缓冲区有特殊要求时，可通过使用相关的功能标志，通知 SDK 进行特殊的处理，默认为 0，SDK 将自动根据网络状态选择合适的传输途径（TCP、UDP or P2P）

outParam： 若函数调用成功，则内核通过该参数输出任务 ID（TaskId）

备注：

该方法与“TransBuffer”功能相同，都是传输上层应用自定义（透明通道）数据，区别在于该方法通过设置相应的功能标识，如可选择采用 UDP 通道传输，但是只针对指定的用户传输，而“TransBuffer”方法则固定采用 TCP 通道传输，且缓冲区大小不能超过 1024 个字节，但可以针对所有用户传输。

从应用来看：

（1）、TransBuffer 适合数据量小、要求实时传输的缓冲区传递，如控制指令等；

（2）、TransBufferEx 适合数据量大、对实时性要求不高的需求；

API 调用成功之后，可通过 outParam 的 GetIntValue 方法获取任务 ID（TaskId）。

5.6.7 传送文件

int TransFile(int userid, String filepath, int wparam, int lparam, int flags, AnyChatOutParam outParam);

功能：传送文件给指定用户

返回值：0 表示函数调用成功，否则为出错代码

参数：

userid： 目标用户编号，只针对某一个用户，不能为-1（所有人）

filepath： 本地文件名，含路径

wParam： 附带参数 1，便于上层应用扩展

lParam： 附带参数 2

dwFlags： 特殊功能标志，参考：“TransBufferEx”方法

outParam： 若函数调用成功，则内核通过该参数输出任务 ID

备注：

该方法传输效率与“TransBufferEx”方法相同，只是在 SDK 内部封装了文件的

分组传输功能，实现对上层应用的透明，简化上层应用的开发难度。

API 调用成功之后，可通过 outParam 的 GetIntValue 方法获取任务 ID（TaskId）。

5.6.8 查询传输任务相关信息

int QueryTransTaskInfo(int Userid, int TaskId, int infoname, AnyChatOutParam outParam)

功能：查询与传输任务相关的信息，如传输进度、传输状态、传输码率等

返回值：0 表示查询成功，否则为出错代码

参数：

dwUserid： 任务发起者用户编号（并非传输目标用户编号）

dwTaskId： 需要查询的任务编号

infoname 需要查询的信息代码（见备注附表）

outParam： 若函数调用成功，则内核通过该参数输出查询结果

备注：

通过调用该方法，可以查询指定传输任务编号的缓冲区传输情况。用户编号与任务编号组合才具有唯一性，不同的用户可能存在相同的任务编号。

API 调用成功之后，可通过 outParam 的 GetIntValue 方法获取查询值

目前提供的查询的信息代码见下表：

<table>
<tr><th>信息代码定义</th><th>参数类型</th><th colspan="2">用途</th><th>备注</th></tr>
<tr><td>TRANSTASK_PROGRESS</td><td>int</td><td colspan="2">传输任务进度查询</td><td>0 ~ 100</td></tr>
<tr><td>TRANSTASK_BITRATE</td><td>int</td><td colspan="2">传输任务当前码率</td><td>单位：bps</td></tr>
<tr><td rowspan="5">TRANSTASK_STATUS</td><td rowspan="5">int</td><td colspan="3">传输任务当前状态：</td></tr>
<tr><td>1</td><td colspan="2">准备状态</td></tr>
<tr><td>2</td><td colspan="2">传输状态</td></tr>
<tr><td>3</td><td colspan="2">完成状态</td></tr>
<tr><td>4</td><td colspan="2">任务被发送者取消</td></tr>
</table>

		5	任务被接收方取消

5.6.9 发送 SDK Filter 通信数据

int SendSDKFilterData(byte[] buf, int len);

功能：向服务器发送 SDK Filter 通信数据

返回值：0 表示成功，否则为出错代码

参数：

buf： 缓冲区

len： 缓冲区的大小

备注：

服务器收到数据后，会将该缓冲区数据全部提交给 SDK Filter，由 SDK Filter 来解析，该缓冲区的内容对于本 SDK 和服务器来说，都是透明的。

5.6.10 音视频录制

int StreamRecordCtrlEx(int dwUserId, int bStartRecord, int dwFlags, int dwParam, String szUserStr);

功能：对指定用户的音视频流进行录制。

返回值：0 表示录制指令被 SDK 成功接收，否则为出错代码

参数：

dwUserId： 需要录制视频的用户编号，可用-1 表示本地用户（自己）；

bStartRecord： 指示当前指令是启动录像，或是停止录像；

dwFlags： 录制功能标志，参考备注；

dwParam： 录制指令附带参数（整形），录像任务结束时，该参数将通过回调函数返回给上层应用；

szUserStr： 录制指令附带参数（字符串类型），录像任务结束时，该参数将通过回调函数返回给上层应用。

备注：

该方法只是向 SDK 下达（停止）录像任务，当指令（bStartRecord）为停止录像时，而且已经录制到了数据时，SDK 将产生一次回调，通知上层应用录像文件名。

视频录制可以在本地进行，也可以在服务器端进行，通过 dwFlags 标志进行控制，视频录制完成之后， 将触发回调事件：OnAnyChatRecordEvent。

录像功能标志指示 SDK 在录制时，进行特殊的处理，0 表示默认（音视频同步录制），目前支持如下标志组合：

ANYCHAT_RECORD_FLAGS_VIDEO ///< 录制视频

ANYCHAT_RECORD_FLAGS_AUDIO ///< 录制音频

ANYCHAT_RECORD_FLAGS_SERVER ///< 服务器端录制

ANYCHAT_RECORD_FLAGS_MIXAUDIO ///< 录制音频时，将其它人的声音混音后录制

ANYCHAT_RECORD_FLAGS_MIXVIDEO ///< 录制视频时，将其它人的视频迭加后录制

ANYCHAT_RECORD_FLAGS_ABREAST ///< 录制视频时，将其它人的视频并列录制

ANYCHAT_RECORD_FLAGS_STEREO ///< 录制音频时，将其它人的声音混合为立体声后录制

ANYCHAT_RECORD_FLAGS_SNAPSHOT ///< 拍照

ANYCHAT_RECORD_FLAGS_LOCALCB ///< 触发本地回调

在服务器端录制音视频，需要单独部署中心录像服务器，参考： Windows 平台中心录像服务器部署、Linux 平台中心录像服务器部署

更多信息可参考：

AnyChat 音视频录制整体解决方案

AnyChat 支持录像文件格式设置（MP4、WMV、FLV、MP3）

中心录像服务器返回录像文件路径可配置

中心服务器录像支持触发客户端回调事件

5.6.11 图像抓拍

int SnapShot(int dwUserId, int dwFlags, int dwParam);

功能：对指定用户的视频进行抓拍。

返回值：0 表示抓拍指令被 SDK 成功接收，否则为出错代码

参数：

dwUserId： 需要抓拍视频的用户编号，可用-1 表示本地用户（自己）；

dwFlags： 功能标志；

dwParam： 抓拍指令附带参数（整形），抓拍图像成功之后，该参数将通过回调函数返回给上层应用；

备注：

该方法只是向 SDK 下达图像抓拍任务，视频抓拍完成之后， 将触发回调事件：OnAnyChatSnapShotEvent。

5.6.12 好友列表

自 AnyChat for Android SDK V1.9 版本开始，AnyChat 提供了一个轻量级的用户好友解决方案，可以实现大厅好友列表、好友分组以及好友在线状态同步等功能。

好友列表需要业务服务器的支持，详情请参考《AnyChat Server SDK 开发指南》以及示例程序（AnyChatCallCenter）源代码。

5.6.13 获取好友列表

int[] GetUserFriends();

功能：获取本地用户的好友列表；

返回值：返回好友用户 ID 列表数组；

参数：无

备注：

当客户端登录系统成功之后，会触发异步消息：WM_GV_LOGINSYSTEM，同时服务器会向客户端发送用户好友信息，当客户端接收完成之后，会触发客户端的异步消息：WM_GV_USERINFOUPDATE，且该消息的 lParam 为 0，表示好友列表有更新。所以该 API 通常在接收到 WM_GV_USERINFOUPDATE 异步消息之后调用。

5.6.14 获取好友在线状态

int GetFriendStatus (int dwFriendUserId);

功能：获取本地用户的好友在线状态，根据状态可以知道好友是否在线。

返回值：返回该好友的在线状态：0 离线， 1 在线

参数：

dwFriendUserId 好友的用户 ID；

备注：

登录成功之后调用有效。

5.6.15 获取好友分组列表

int [] GetUserGroups();

功能：获取本地用户的好友分组列表，返回好友分组 ID 列表数组。

返回值：用户 ID 数组

参数：无

备注：

登录成功之后调用有效。好友分组是指将好友归纳到某一个组别下，如“家

人”、“大学同学”以及“老师”等。每一个分组对应一个分组 ID，通过分组 ID 可以获取分组的名称。

5.6.16 获取分组名称

String GetGroupName(int dwGroupId);

功能：根据分组 ID 获取分组名称。

返回值：分组名称字符串

参数：

dwGroupId 分组 ID；

备注：

登录成功之后调用有效。分组名称由业务服务器设置。

5.6.17 获取分组所对应的用户列表

int[] GetGroupFriends(int dwGroupId);

功能：获取分组所对应的用户列表，即该分组下有多少用户。

返回值：用户 ID 数组

参数：

dwGroupId 分组 ID；

备注：

登录成功之后调用有效。通过该 API 接口，可以获得每一个分组下面的用户列表，进而可以获得该分组下每一个用户的详细信息。

5.6.18 获取好友用户信息

String GetUserInfo(int dwUserId, int dwInfoId)

功能：获取好友用户的详细信息。

返回值：用户信息字符串

参数：

dwUserId 好友用户 ID；

dwInfoId 用户信息类型 ID，业务层可自定义；

备注：

登录成功之后调用有效。当业务服务器调用 API：BRAS_SetUserInfo 设置了用户的信息之后，客户端便可以通过该 API 获取业务服务器所设置的信息，其中 dInfoId 由业务层（上层应用）自己定义。

关于好友用户信息这一部分，对于 AnyChat 来说是透明的，业务服务器设置了什么样的信息，客户端便可以获取到什么样的信息，AnyChat 只是提供了一个信息传输的中间通道，业务层可以自由扩展。

5.6.19 用户信息控制

int UserInfoControl(int dwUserId ,int dwCtrlCode, int wParam, int lParam,String lpStrValue);

功能：对用户信息进行控制。

返回值：0 表示成功，否则为出错代码

参数：

dwUserId 用户 ID；

dwCtrlCode 控制代码，业务层自定义，其中<100 的值为系统保留，业务层使用时，其值必须>100

wParam 附带参数，业务层自定义；

lParam 附带参数，业务层自定义；

lpStrValue 附带参数（字符串类型），业务层自定义，可为空；

备注：

登录成功之后调用有效。该 API 调用之后，会向业务服务器发送信息控制指令，将会触发业务服务器对应的回调函数。

5.7 系统设置

5.7.1 枚举本地视频采集设备

String[] EnumVideoCapture();

功能：枚举本地视频采集设备

返回值：String[]

备注

该方法将会在内部分配缓冲区，外部使用完成之后，必需手工释放这些缓冲区，否则会造成内存泄露，由于内部采用了“GlobalAlloc”来分配高端内存，故外部需要调用“GlobalFree”来释放，而不能是 delete 或 free 方法，具体使用方法请参考 Demo 程序中的源代码。

5.7.2 选择指定的视频采集设备

int SelectVideoCapture(String szCaptureName);

功能：选择指定的视频采集设备

返回值：0 表示成功，否则为出错代码

参数：

szCaptureName 所获取设备的名称；

备注

当用户有多个视频采集设备（USB 摄像头、虚拟摄像头、采集卡等）时，可以通过该方法选用指定的视频采集设备。

5.7.3 获取当前视频采集设备

String GetCurVideoCapture();

功能：获取当前使用的视频采集设备名称

返回值：String

5.7.4 枚举本地音频采集设备

String[] EnumAudioCapture();

功能：枚举本地音频采集设备

返回值：String[]

备注

该方法将会在内部分配缓冲区，外部使用完成之后，必需手工释放这些缓冲区，否则会造成内存泄露，由于内部采用了“GlobalAlloc”来分配高端内存，故外部需要调用“GlobalFree”来释放，而不能是 delete 或 free 方法，具体使用方法请参考 Demo 程序中的源代码。

5.7.5 选择指定的音频采集设备

int SelectAudioCapture(String szCaptureName);

功能：选择指定的音频采集设备

返回值：0 表示成功，否则为出错代码

参数：

szCaptureName 所获取设备的名称；

备注

当用户有多个音频采集设备（板载声卡、USB 声卡等）时，可以通过该方法选用指定的音频采集设备。

5.7.6 获取当前音频采集设备

String GetCurAudioCapture();

功能：获取当前使用的音频采集设备

返回值：String

5.7.7 获取音频设备的当前音量

int AudioGetVolume(int device);

功能：获取指定音频设备的当前音量

返回值：0 表示成功，否则为出错代码

参数：

device 设备类型，定义为：

AD_WAVEIN = 0, ///< 输入设备：Mic

AD_WAVEOUT = 1, ///< 输出设备：Wave

备注

根据设备类型（device）参数的不同，可以获取放音设备（WaveOut）和录音设备（WaveIn）的当前音量大小。

5.7.8 设置指定音频设备的音量

int AudioSetVolume(int device, int Volume);

功能：设置指定音频设备的音量

返回值：0 表示成功，否则为出错代码

参数：

device 设备类型，定义为：

AD_WAVEIN = 0, ///< 输入设备：Mic

AD_WAVEOUT = 1, ///< 输出设备：Wave

dwVolume 需要设置的音量，取值范围：0~100，值越大，音量越大；

备注

根据设备类型（device）参数的不同，可以调节放音设备（WaveOut）和录音设备（WaveIn）的音量大小。

5.7.9 SDK 内核参数设置（整形值）

int SetSDKOptionInt(int optname, int optvalue);

功能：SDK 内核参数设置（整形值参数）

返回值：0 表示成功，否则为出错代码

参数：

optname 内核参数名称；

optval 设置的参数值

备注

可以通过该方法对 AnyChat Core SDK 内部的参数进行设置，实现特殊的功能要求。

目前提供的可设置内核参数名称代码见 WIN32 平台 SDK 相关定义。

5.7.10 SDK 内核参数设置（字符串值）

int SetSDKOptionString(int optname, String optvalue);

功能：SDK 内核参数设置（字符串值参数）

返回值：0 表示成功，否则为出错代码

参数：

optname 内核参数名称；

optval 设置的参数值

备注

可以通过该方法对 AnyChat Core SDK 内部的参数进行设置，实现特殊的功能要求。

目前提供的可设置内核参数名称代码见 WIN32 平台 SDK 相关定义。

5.7.11 SDK 内核参数状态查询

int GetSDKOptionInt(int optname);

功能：SDK 内核参数状态查询（整形值）

返回值：返回查询结果

参数：

optname 内核参数名称；

备注

可以通过该方法对 AnyChat Core SDK 内部的参数进行状态查询，获取当前的设置。

String GetSDKOptionString(int optname);

功能：SDK 内核参数状态查询（字符串）

返回值：返回查询结果

参数：

optname 内核参数名称；

备注

可以通过该方法对 AnyChat Core SDK 内部的参数进行状态查询，获取当前的设置。

5.8 业务排队

自 AnyChat for Android SDK V2.3 版本开始，AnyChat 提供了业务排队功能，抽象出了业务排队应用场景中需要的营业厅、队列、坐席、客户等业务对象，通过调用提供的客户端 API 来操作这些对象的属性、方法及事件，如：进出营业厅、进出队列方法；获取排队人数、在队列中所排位置属性；进出队列、坐席服务响应事件等。通过响应不同的业务对象事件来实现排队业务逻辑功能，开发人员只需关注业务逻辑的实现，AnyChat 会自动的维护业务对象的数据变化。

具体可以参考：AnyChat 提供业务排队整体解决方案。

5.8.1 获取对象 ID 列表

int[] ObjectGetIdList(int dwObjectType)

功能: 获取业务对象的 Id 数组

返回值: dwObjectType 类型的Id数组

参数:

dwObjectType 业务对象类型，见3.5.1.1 章节描述内容

备注: 可以通过该方法查询业务对象的整形数组的属性，如服务区域的业务队列Id数组

5.8.2 获取对象属性值

int ObjectGetIntValue(int dwObjectType, int dwObjectId, int dwInfoName)

功能: 获取业务对象属性值(整形值)

返回值:业务对象的属性值，查询失败为0

参数:

dwObjectType	业务对象类型，见3.5.1.1 章节描述内容
dwObjectId	业务对象的Id
dwInfoName	业务对象属性名称，各类业务对象的属性定义见3.5.2 章节描述内容

备注:可以通过该方法查询业务对象的整型属性值，如队列人数，排队时间等。

String ObjectGetStringValue(int dwObjectType, int dwObjectId, int dwInfoName)

功能: 获取业务对象属性值(字符串)

返回值: 业务对象的属性值，查询失败为0

参数:

dwObjectType	业务对象类型，见3.5.1.1 章节描述内容
dwObjectId	业务对象的Id
dwInfoName	业务对象属性名称，各类业务对象的属性定义见3.5.2 章节描述内容

备注: 可以通过该方法查询业务对象的字符串属性值，像服务区域名称，队列名称

5.8.3 设置对象属性值

int ObjectSetIntValue(int dwObjectType, int dwObjectId, int dwInfoName, int dwValue)

功能: 设置业务对象属性值(整型)

返回值: 0表示成功，否则为错误码

参数:

dwObjectType 业务对象类型，见3.5.1.1 章节描述内容

dwObjectId 业务对象的Id

dwInfoName 业务对象属性名称，各类业务对象的属性定义见3.5.2 章节描述内容

dwValue 要设置的业务对象的属性名称

备注: 可以通过该方法设置业务对象的整型属性值，像业务员队列的Id，服务区域的Id

int ObjectSetStringValue(int dwObjectType, int dwObjectId, int dwInfoName, String lpStrValue)

功能: 设置业务对象属性值(字符型)

返回值:0表示成功，否则为错误码

参数:

dwObjectType 业务对象类型，见3.5.1.1 章节描述内容

dwObjectId 业务对象的Id

dwInfoName 业务对象属性名称，各类业务对象的属性定义见3.5.2 章节描述内容

lpStrValue 要设置的业务对象的属性名称

备注: 可以通过该方法设置业务对象的字符串属性值，像服务区域名称，队列名

称

5.8.4 业务对象参数控制

int ObjectControl(int dwObjectType, int dwObjectId, int dwCtrlCode, int dwParam1, int dwParam2, int dwParam3, int dwParam4, String lpStrValue)

功能: 排队业务流程的控制，进入服务区域，出服务区域，进入队列，出对列，数据同步等所有的业务都由它负责。

返回值: 0 表示成功，否则表示错误码

参数:

dwObjectType	业务对象类型，见 3.5.1.1 章节描述内容
dwObjectId	业务对象的 Id
dwCtrlCode	各类业务对象的方法定义，见 3.5.3 章节描述内容；
dwParam1	整型参数值一（值由业务对象方法参数值决定，默认为 0）
dwParam2	整型参数值二（值由业务对象方法参数值决定，默认为 0）
dwParam3	整型参数值三（值由业务对象方法参数值决定，默认为 0）
dwParam4	整型参数值四（值由业务对象方法参数值决定，默认为 0）
lpStrValue	字符串参数值（值由业务对象方法参数值决定，默认为空）

备注: 可以通过该方法来控制排队业务流程的事件，如数据同步，进入服务区域，进队列，出对列， 出服务区域。

5.9 媒体播放

5.9.1 流媒体播放初始化

int StreamPlayInit(String lpTaskGuid, String lpStreamPath,

int dwFlags,String lpStrParam)

功能： 初始化流媒体播放。

返回值： 0 表示成功， 否则为出错代码

参数:

lpTaskGuid 任务 Guid，可通过 BRAC_GetSDKOption 获取（见 3.1 章节描述内容）

lpStreamPath 流媒体的路径， 可为网络 URL 或本地路径

dwFlags 播放类型的标志（见 3.5.6.2 章节描述内容）

lpStrParam 预留参数，可传空字符串

5.9.2 流媒体播放控制

int StreamPlayControl(String lpTaskGuid, int dwCtrlCode, int dwParam, int dwFlags, String lpStrParam)

功能： 控制流媒体播放。

返回值： 0 表示成功， 否则为出错代码

参数：

lpTaskGuid 任务 Guid， 需要与初始化是传入的一致

dwCtrlCode 流媒体播放控制码（见 3.5.6.4 章节描述内容）

dwParam 预留参数， 可传 0

dwFlags 预留参数， 可传 0

lpStrParam 预留参数， 可传空字符串

5.9.3 设置流媒体播放视频显示位置

int StreamPlaySetVideoPos(String lpTaskGuid, Surface s, int dwLeft, int dwTop, int dwRight, int dwBottom)

功能： 设置流媒体播放视频显示位置。

返回值：无

参数：

lpTaskGuid 任务 Guid， 需要与初始化是传入的一致

s 视频显示表面

dwLeft、dwTop、dwRight、dwBottom 位置信息，默认为 0

5.9.4 流媒体播放获取参数信息

String StreamPlayGetInfo(String lpTaskGuid, int dwInfoName)

功能： 获取流媒体播放的参数信息。

返回值： 返回包含了对应信息的 json 字符串

参数：

lpTaskGuid 任务 Guid， 需要与初始化是传入的一致

dwInfoName 要获取的信息代码（见 3.5.6.3 章节描述内容）

返回 json 详情

```
{
    "audiobitrate":256,//音频码率
    "audiocodec":23,//编码器
    "audioduration":45540,//音频时长
    "bitspersample":16,//采样精度
    "channels":1,//单声道
    "errorcode":0,//错误码
    "filebitrate":256,//文件码率
    "fileduration":45540,//媒体总时间
    "filename":"2.mp3",//文件名
    "playspeed":1,//播放速度
    "playstatus":0,//播放状态
    "playtime":0, //播放到进度时间
    "samplespersec":16000,//音频采样率
    "taskguid":"E444CCD1-4D27-48FE-A9D5-BD0074A0B557"//任务 ID
}
```

5.9.5 流媒体播放释放资源

int StreamPlayDestroy(String lpTaskGuid, int dwFlags)

功能： 释放流媒体播放的资源。

返回值： 0 表示成功， 否则为出错代码

参数：

lpTaskGuid 任务 Guid， 需要与初始化是传入的一致

dwFlags 预留参数， 可传 0

5.10 SDK 控制

String SDKControl(int dwCtrlCode, String strInParam)

功能： 执行相应的控制。

返回值： 0 表示成功， 否则为出错代码

参数：

dwCtrlCode SDK 控制码（见 3.5.7 章节内容描述）

strInParam json 字符串

接口示例（删除文件）：

```
SDKControl(ANYCHAT_SDKCTRL_FILEDELETE, “{‘filename’: ‘d:\\me.mp4’}”);
```

6 版本变更记录

2020-01-01 V8.0

优化视频通信模块，支持 5G 网络下高清视频通信；
新增 AI SDK 模块，实现音视频与第三方 AI 原子能力的无缝对接；
新增 IPv6 接入模块，支持 IPv6/IPv4 双栈视频通信能力；
新增物联网平台 SDK，实现全平台视频接入能力覆盖；
新增电子白板模块，多方协同更加便捷；
兼容最新版本 Android 10 平台，适配更多新款移动终端设备；
优化智能排队能力，支持更灵活的排队策略、坐席服务策略配置；

2019-07-15 V7.4

适配最新移动端平台，全面兼容 Android Q；
支持多种风险揭示能力，包括语音、视频、PPT 等，集成双录业务更加灵活；
优化移动端屏幕录制能力，录制视频更加流畅；
优化智能排队，支持坐席跨营业厅服务及关联队列组模式；
连接服务器时，支持一次传入多个服务器地址，支持高可用；
增加业务数据传输通道，不限制传输数据容量；
新增用户 APP 信息传入接口，用于跟踪接入渠道、业务流水号等信息；
修正文件传输回调速度慢的问题；
增加禁止客户端日志信息写入本地文件的 API 接口；
客户端增加查询用户详细信息的接口，且支持批量查询；
支持服务器录像时，定期向客户端通知录像状态；
修正视频呼叫时附带参数超过 1000 字节长度会被截断的问题；

2019-01-18 V7.3

全面兼容 Android 9.0，同步优化语音通话质量
支持加载 SSL 安全证书，采用 TLS 通信机制，提供高安全保障
服务器录像加入多路流的支持
支持录像文件计算 MD5 值回调给上层业务
服务器合成录像支持叠加水印、时间戳
开放更多录像参数设置能力，上层业务使用更灵活
支持小程序、H5 通信过程中进行服务器拍照

2018-09-18 V7.2

优化流媒体传输能力，提升弱网环境下的通信质量；
优化全景录像，提供多录像任务能力；
新增移动端屏幕录制能力；
优化智能排队，提供呼损率等数据统计能力；

2018-05-30 V7.1

优化弱网环境下的服务器录像能力；
优化 HTML5 的音视频接入；
支持小程序接入；
新增在“服务器合成录制”时叠加水印；
新增服务器、客户端同步录制功能，可同时在服务器以及客户端生成相同的录像文件；
新增在实时视频、双录视频上使用服务器时间戳；
优化智能排队业务逻辑，支持坐席端显示队列用户详细信息；

2018-02-01 V7.0

新增支持 Android 8.x 平台；
优化底层音视频库，编码效率大幅提升；
双录过程中支持风险揭示 PPT、风险揭示视频的切换；
修正一个用户同时录制两路以上视频时，有时只能生成单个录像文件的问题；
开放同步数据传输接口，确认数据已送达；
优化智能排队业务逻辑，坐席可同时服务多个客户；
支持 PCM 音频格式录制，可用于声纹特征提取；
开放录像参数更新接口；
修正手机锁屏再恢复界面无视频画面的问题；
支持录像过程中取消录像；

2017-09-10 V6.5

优化视频编码与复杂网络自适应能力，用户体验更好；
新增坐席虚拟背景能力，支持动态调整背景；
提高本地拍照、服务器拍照的图片质量；
兼容一些 USB 摄像头，修正录制过程中导致音视频不同步的问题；
修正核心服务器与录像服务器网络抖动影响录像的问题；
支持 TCP 通道传输流媒体数据；
优化智能排队业务逻辑，坐席支持跨营业厅服务；
新增对多路流拍照的支持；

2017-05-21 V6.4

优化音视频编解码算法，提升音视频质量；
优化网络传输，提升在高丢包率下的通话效果；
优化网络连接，缩短连接服务器的时间，响应更快；
增加 SDKControl API 接口，满足更复杂的业务需求；
增加媒体文件及网络流媒体播放能力；
优化底层库，全面兼容 Android 7.0；
修正部分新款手机的兼容性问题；

2017-01-21 V6.3

新增会话保持功能，WiFi、4G 网络动态切换保持视频通话不中断；
优化断线重连机制，修正有时不能自动重连的 Bug；
增加客户端日志文件控制接口；
多路视频流合成录制时，支持选择指定的流进行录制；

2016-11-01 V6.2

支持 arm64-v8a 芯片架构，提供对应指令集的 SDK 组件；
优化文字消息发送接口，支持发送超大容量内容；
向服务器传送文件时，支持触发本地回调事件；
优化网络连接流程，适应高延迟卫星网络环境；
智能排队新增自动路由功能；
移动端 SDK 提供单例模式接口；

2016-06-01 V6.1

兼容 Android 6.x 内核；
支持最新互联网协议－IPv6-only 标准；
拍照、录像支持自定义文件名；
修正开启静音检查特性后服务器录制的文件音视频不同步的问题；

2016-02-02 V6.0

更新 AnyChat Platform Core SDK V6.0 内核，采用与 Core SDK 一致的版本命名；
支持 RSA 非对称加密签名用户接入验证体系；
支持字符串用户 ID，可与默认的整形用户 ID 混用；
支持 HTML5 音视频接入，与 WebRTC 音视频互通；
支持上层业务自定义录像文件名；
支持公有云、私有云的接入；
优化透明通道功能，突破传输缓冲区为 1KByte 的限制；

2015-07-15 V2.2

优化移动设备上服务器合成流录制，功能更稳定；
增加业务对象 API，支持智能排队等新功能；
支持动态调节视频参数，包括码率、视频质量、预设参数等；
增加中心服务器拍照功能；
优化网络数据传输，提高移动网络抖动时的音视频质量；
修正在 android 5.0 系统上应用闪退的问题；

2015-01-15 V2.2

优化音视频内核，编解码效率更高；
增加服务器合成录制、合成流录制功能；
视频录制过程中支持动态改变视频分辨率；
优化网络抖动，减少网络连接断开，提高连接速度；
支持通过 API 接口开启 AnyChat 内核调试模式；
增加对 Android arm64-v8a 平台的支持；
针对部分海思芯片手机视频初始化失败的问题进行适配；
Android 增加本地视频录制，拍照功能；

2014-09-01 V2.1

优化并更新内核 Codec 库，整体性能有较大幅度提升；

针对网络状况较差时网络连接的稳定性进行优化；
AnyChat 支持录像文件格式设置（MP4、WMV、FLV、MP3）；
中心录像服务器返回录像文件路径可配置；
中心服务器录像支持触发客户端回调事件；
实现服务器集中收集客户端日志信息功能；
Android 平台播放语音时支持听筒、喇叭之间切换；
修正 Android 平台特定场景下会导致应用闪退的问题；
修正低性能设备上视频通话时延迟累计增大的问题；
修正合成录像时本地视频有时会卡顿的问题；

2014-06-01 V2.0

增加数据加密、解密 API 接口，可实现上层应用对语音、视频等数据的加解密；
AnyChat Server SDK 支持 64bit Java 环境；
增加视频方向手工修正 API 接口，包括本地视频采集方向和远程视频显示方向；
修正文件传输时在某些网络环境下文件内容接收不完整的问题；
优化文件传输模块，提高文件传输效率，特别针对卫星网络环境下的文件传输优化；
修正传输大容量文件时内存占用过高的问题，新增文件传输的断点续传功能；
修正 Android 平台 Java 驱动模式下，摄像头状态不准的 Bug；

2014-01-01 V1.9

调整 Android 平台缓冲区（文件）传输 API 接口参数，支持 taskid 输出；
提供完整的视频呼叫解决方案；
提供完整的大厅好友解决方案；
修正 Android 平台收发大量文字消息时会导致错误的 Bug；
修正 Android 平台不能接收中文文件名的 Bug；
优化回声消除算法，提高通话过程中的用户体验；
支持最新的 Android 4.4 平台；
修正 Android 平台频繁分配内存导致 java 内存回收机制占用资源过多的 Bug；
修正 Android 平台竖屏模式（Portrait）本地预览图像旋转 90 度的 Bug；
将 Java 层的 AnyChatTransTaskOutParam 类由通用的 AnyChatOutParam 类代替；
修正 Android 平台视频显示有时只部分刷新的问题；

2013-07-08 V1.8

基于 AnyChat Platform Core SDK V4.8 版本编译

2013-03-20 V1.7

基于 AnyChat Platform Core SDK V4.7 版本编译

2012-11-22 V1.6

基于 AnyChat Platform Core SDK V4.6 版本编译

2012-09-10 V1.5

基于 AnyChat Platform Core SDK V4.5 版本编译

2012-05-11 V1.4

基于 AnyChat Platform Core SDK V4.4 版本编译

2012-02-20 V1.3

基于 AnyChat Platform Core SDK V4.3 版本编译；

优化语音通信效果；

支持 Android 4.x 平台；

2011-11-15 V1.2

基于 AnyChat Platform Core SDK V4.2 版本编译；

修正内核多线程操作界面元素导致程序异常的 Bug；

增加回音抑制模块，提高语音通信体验；

2011-10-08 V1.0

初始版本，基于 AnyChat Platform Core SDK V4.1 版本编译

7 附录一：错误代码参考

```
#define GV_ERR_SUCCESS              0       ///< 成功
#define GV_ERR_DB_ERROR             1       ///< 数据库错误
#define GV_ERR_NOTINIT              2       ///< 系统没有初始化
#define  GV_ERR_NOTINROOM           3       ///< 还未进入房间
#define GV_ERR_FUNCNOTALLOW         20      ///< 函数功能不允许（初始化时没有指定该功能）
//连接部分
#define GV_ERR_CONNECT_TIMEOUT      100     ///< 连接服务器超时
#define GV_ERR_CONNECT_ABORT        101     ///< 与服务器的连接中断
#define GV_ERR_CONNECT_AUTHFAIL     102     ///< 未能通过服务器的认证，属于非法连接
//登录部分
#define GV_ERR_CERTIFY_FAIL         200     ///< 认证失败，用户名或密码有误
#define GV_ERR_ALREADY_LOGIN        201     ///< 该用户已登录
#define GV_ERR_ACCOUNT_LOCK         202     ///< 帐户已被暂时锁定
#define GV_ERR_IPADDR_LOCK          203     ///< IP 地址已被暂时锁定
#define GV_ERR_VISITOR_DENY         204     ///< 游客登录被禁止（登录时没有输入密码）
#define GV_ERR_INVALID_USERID       205     ///< 无效的用户 ID（用户不存在）
//进入房间
#define GV_ERR_ROOM_LOCK            300     ///< 房间已被锁住，禁止进入
#define GV_ERR_ROOM_PASSERR         301     ///< 房间密码错误，禁止进入
#define GV_ERR_ROOM_FULLUSER        302     ///< 房间已满员，不能进入
#define GV_ERR_ROOM_INVALID         303     ///< 房间不存在
#define GV_ERR_ROOM_EXPIRE          304     ///< 房间服务时间已到期
#define GV_ERR_ROOM_REJECT          305     ///< 房主拒绝进入
#define GV_ERR_ROOM_OWNERBEOUT      306     ///< 房主不在，不能进入房间
#define GV_ERR_ROOM_ENTERFAIL       307     ///< 不能进入房间
#define GV_ERR_ROOM_ALREADIN        308     ///< 已经在房间里面了，本次进入房间请求忽略
//私聊
#define GV_ERR_ROOM_PRINULL         401     ///< 用户已经离开房间
#define GV_ERR_ROOM_REJECTPRI       402     ///< 用户拒绝了私聊邀请
#define GV_ERR_ROOM_PRIDENY         403     ///< 不允许与该用户私聊，或是用户禁止私聊
#define GV_ERR_ROOM_PRIREQIDERR     420     ///< 私聊请求 ID 号错误，或请求不存在
#define GV_ERR_ROOM_PRIALRCHAT      421     ///< 已经在私聊列表中
```

更多错误代码可参考：SDK\Client\C++\GVErrorCodeDefine.h 文件。