点对点 SDK(Java 版)说明(V1.0.0)

www.mediapro.cc

一、基本概念

音视频纯发送端：可以将自身音视频发送给对端，但不具备接收对端音视频流的能力。
音视频纯接收端：可以接收对端发送过来音视频，但不具备发送音视频给对端的能力。
音视频收发一体端：同时具备音视频发送和接收的能力。
在不同的业务场景里，通过设置不同的客户端类型来降低 SDK 资源的占用。比如投屏类应用，一端设置为纯发送端，另一端设置为纯接收端，这样发送端无需创建接收线程等资源，接收端也无需创建发送所需的资源。当需要全双工通讯时，设置双方为收发一体类型。
传输参数：本文传输参数是指视频通道的 FEC 上行 FEC 冗余度、上行 FEC Group 组大小、接收 Qos 丢包等待时间，音频通道在内部已经根据经验数据配置好了合适的值且不对外开放。对于音视频纯接收者，同样也可以设置 FEC 上行冗余度、上行 FEC Group 组大小，只是没有实际意义(因为不会发送数据，当然也不会进行 FEC 编码)。
主动 API 接口：由外层应用主动发起的调用，比如登录、下线、发送音视频等。
被动 API 接口：称为回调接口更为贴切，比如接收到来对端的音视频数据、底层帧率自适应建议的通知、底层 IDR 帧请求的通知。

二、主动接口

以下接口均为线程安全，可以在多线程中调用，定义位于 CSDInterface.java 中

1、系统初始化

系统初始化主要完成日志模块等资源的初始化，该 API 在整个系统中只需要调用一次即可。
int SDsysinit(String strLogFileDir, byte byLogFileLevel)
参数：
@strLogFileDir，日志文件输出的目录，若目录不存在，SDK 将自动创建，支持相对路径或绝对路径。
@byLogFileLevel，日志输出的级别，只有等于或者高于该级别的日志会输出到文件。级别定义位于 Constant.java：

```
final byte LOG_LEVEL_DEBUG = 1;
final byte LOG_LEVEL_INFO = 2;
final byte LOG_LEVEL_WARN = 3;
final byte LOG_LEVEL_ERROR = 4;
final byte LOG_LEVEL_ALARM = 5;
final byte LOG_LEVEL_FATAL = 6;
final byte LOG_LEVEL_NONE = 7;
```

当指定为 SD_LOG_LEVEL_NONE 时，将不会生成日志文件。
返回值：
返回 0 表示初始化成功，返回负数则为失败，负数值为其错误码。

2、系统反初始化

void SDsysexit()
与 SDsysinit 对应，系统退出前调用。
参数：无

返回值：无

3、创建本地资源

```
int SDOnlineUser(byte byUserType, String strLocalIp, int nLocalPort, String strRemoteIp, int nRemotePort);
```

参数：

@byUserType，表示客户端的类型，CSDInterface.java 中，包括

```
final byte USER_TYPE_OTHER = 0;
final byte USER_TYPE_AV_SEND_RECV = 1;
final byte USER_TYPE_AV_RECV_ONLY = 2;
final byte USER_TYPE_AV_SEND_ONLY = 3;
```

用户根据自身业务选择合适的客户端类型，以获得资源的最低占用。

@strLocalIp，绑定的本地 IP 地址，当设置为空字符串”“时（不是 NULL，而是空字符串），内部将使用 INADDR_ANY，交由操作系统选择一个网卡 IP。当存在多个网卡时，建议指定使用的哪一个网卡 IP，避免数据无法正确发出。

@nLocalPort，绑定的本地端口号，当客户端为 USER_TYPE_AV_RECV_ONLY 或者 USER_TYPE_AV_SEND_RECV 时，必须设置非 0 的本地端口，以便接收对方发出的流。当客户端为 USER_TYPE_AV_SEND_ONLY 时，允许设置本地端口为 0，由操作系统选择一个当前可用的端口发出数据。

@strRemoteIP，远端 IP 地址。当客户端为纯接收端 USER_TYPE_AV_RECV_ONLY 时，设置远端 IP 地址为空字符串即可（作为纯接收端，一般是不知道发送端的 IP 和端口的，内部将在收到远端数据后自动翻转 IP 和端口，从而获得可用于向远端发送数据的 IP 和端口）。

@nRemotePort，远端端口号。当客户端为纯接收端 USER_TYPE_AV_RECV_ONLY 时，设置远端端口号为 0 即可。

返回值：

返回 0 表示登录成功，返回负数则为失败，负数值为其错误码。

4、回收创建的资源

```
void SDOfflineUser();
```

参数：无

返回值：无

5、发送视频数据

```
void SDSendVideoStreamData(byte[] byBuf, int nlen);
```

发送已编码的一帧视频码流，内部自带拆分功能，一次传入**带 H264 起始码**的一帧码流。SDK 内部管理时间戳。

参数：

@byBuf，码流存放区。

@nlen，码流长度。

返回值：无

6、发送音频数据

```
void SDSendAudioStreamData(byte[] byBuf, int nlen);
```

发送已编码的一帧音频码流，一次传一帧 **ADTS** 码流。SDK 内部管理时间戳。

参数：

@byBuf，码流存放区。

@nlen，码流长度。

返回值：无

7、设置音视频传输参数

```
void SDSetTransParams(int nRedunRatio, int nGroupSize, int nEnableNack, int
nJitterBufTime);
```

参数：

@nRedunRatio，设置为非零值时表示：固定冗余度时对应的上行冗余比率。比如设置为 30，则表示使用 30%的固定冗余。当设置为 0 时表示使用 AUTO_REDUN 自动冗余度。

@nGroupSize， 为上行 FEC 分组大小，512Kbps 以下建议设置为 8，512Kbps~1Mbps 建议设置为 16，1Mbps~2Mbps 建议设置 24，2Mbps~4Mbp 建议设置 28，4Mbps 以上建议 36。当设置为 0 时表示关闭发送 FEC 编码功能，不做 FEC 保护。

@nEnableNack，是否启用 NACK 功能（1 表示开启，0 表示关闭），关于 NACK 请阅读相关文档，建议设置为 1 开启。

@nJitterBufTime，本客户端接收码流时的内部缓存时间（毫秒）,范围 0~600。设置为 0 时，将关闭内部接收 JitterBuff 功能。本参数仅影响接收，对于 USER_TYPE_AV_SEND_ONLY 不会生效。

返回值：无

注意：本函数需在 SDOnlineUser 之前调用，本 API 的使用若有疑问，请联系技术支持获得帮助。

8、获取当前统计数据

```
MediaTransStatis SDGetMediaTransStatis(MediaTransStatis transStatis)
```

参数：

@transStatis，底层 C++获取传输统计数据后将写入本对象。MediaTransStatis 的定义请参考 MediaTransStatis.java

返回值：获取到的当前统计数据

说明：使用举例

```
private MediaTransStatis mTransStatis = new MediaTransStatis();
mTransStatis = mInterface.SDGetMediaTransStatis(mTransStatis);
Log.d(TAG,
"DownBitrate:" + mTransStatis.nVideoDownRate +
"DownLostratio:"+ mTransStatis.nVideoDownLostRatio);
```

三、接收音视频数据接口

1、视频接收

用户需自行实现接收视频数据的 Interface，方法定义定义位于 CSDInterface.java 中，如下：

```
public interface OnVideoDataCallBack
{
    public void onDataCallBack(byte[] byBuf, int nlen, int nPts, int nPosition,
```

```
                                    int bPacketLost, int bKeyFrame);
}
```

底层 C++接收到远端的视频数据后将调用本接口的 onDataCallBack 方法。

参数：

@byBuf，指向接收的码流帧存放区域

@nlen，接收码流帧的长度，**注意请务必使用本参数指定的长度读取 byBuf**

@nPts，当前码流帧的时间戳

@nPosition，保留，暂未使用。

@bPacketLost 表示当前帧是否接收完整，若网络丢包且 FEC 未能恢复时，该标志将置位。

@bKeyFrame 表示当前帧是否为 IDR 关键帧。

如何使用后两个参数实现丢帧冻结可以联系技术支持获得帮助。需要说明的是，当没有丢包发生时，本函数的输出与对方调用 SDSendVideoStreamData 函数的输入完全一致。

返回值：无

说明：SDK 内部是在独立于网络接收线程之外的线程中调用本接口，所以外层可以将一定耗时的操作（比如解码）放置在此。

说明：用户实现 OnVideoDataCallBack 后需调用 CSDInterface 类的如下方法生效让其生效：

```
void setVideoRecvListener(OnVideoDataCallBack videoRecvListener)
```

2、音频接收

用户需自行实现接收视频数据的 Interface，方法定义定义位于 CSDInterface.java 中，如下：

```
public interface OnAudioDataCallBack
{
    public void onDataCallBack(byte[] byBuf,  int nlen,  int nPts,  int nPosition);
}
```

底层 C++接收到远端的视频数据后将调用本接口的 onDataCallBack 方法。

参数：

@byBuf，指向接收的码流帧存放区域

@nlen，接收码流帧的长度，**注意请务必使用本参数指定的长度读取 byBuf**

@nPts，当前码流帧的时间戳

@nPosition，保留，暂未使用。

返回值：无

说明：SDK 内部是在独立于网络接收线程之外的线程中调用本接口，所以外层可以将一定耗时的操作（比如解码）放置在此。

说明：用户实现 OnAudioDataCallBack 后需调用 CSDInterface 类的如下方法生效让其生效：

```
void setAudioRecvListener(OnAudioDataCallBack audioRecvListener)
```