**针对使用连麦sdk实现推流和播放功能的客户，可以采用我们新的推流SDK和播放器SDK来替代，这样做的好处有： **

- CPU、内存、发热、电耗方面都有较大提升；
- 稳定性也有非常大的提升；
- 接口使用更简洁方便；
- 美颜也有比较大的提升；
- 库更小，可支持外部美颜；

一、SDK升级

1.去掉连麦sdk库

2.添加推流sdk库

二、接口升级

一、主播端（AlivcVideoChatHost -> AlivcLivePusher）

基本功能

按照以下对应关系重构代码：

- init->init
- prepareToPublish->startPreview/startPreviewAsync
- startToPublish->startPush/startPushAsync
- stopPublishing->stopPush
- finishPublishing->stopPreview
- release->destroy
- pause->pause
- resume->resume

注意：如果接口在子线程调用，那么可以采用同步接口；如果接口在主线程执行因为建议使用异步startPushAsync

通知处理（OnInfoListener->AlivcLivePushInfoListener）

- onPreviewStarted(AlivcLivePusher pusher);
- onPreviewStoped(AlivcLivePusher pusher);
- onPushStarted(AlivcLivePusher pusher);

- onPushPauesed(AlivcLivePusher pusher);
- onPushResumed(AlivcLivePusher pusher);
- onPushStoped(AlivcLivePusher pusher);
- onPushRestarted(AlivcLivePusher pusher);
- onFirstFramePreviewed(AlivcLivePusher pusher);

网络通知（OnInfoListener/OnErrorListener->AlivcLivePushNetworkListener）

- onNetworkPoor(AlivcLivePusher pusher);
- onNetworkRecovery(AlivcLivePusher pusher);
- onReconnectStart(AlivcLivePusher pusher);
- onReconnectFail(AlivcLivePusher pusher);
    - 对应处理: reconnectPushAsync/设置网络/返回上一界面

- onReconnectSucceed(AlivcLivePusher pusher);
- onSendDataTimeout(AlivcLivePusher pusher);
    - 对应处理: reconnectPushAsync/设置网络/返回上一界面

- onConnectFail(AlivcLivePusher pusher);
    - 对应处理: reconnectPushAsync/设置网络/返回上一界面

错误通知（OnErrorListener->AlivcLivePushErrorListener）

- onSystemError(AlivcLivePusher livePusher, AlivcLivePushError error);
    - 对应处理: 退出应用

- onSDKError(AlivcLivePusher livePusher, AlivcLivePushError error);
    - 对应处理：restartPushAsync

码率控制

客户根据清晰度及流畅度要求，可以设置以下参数 * 目标码率（AlivcLiveConfig.setVideoTargetBitrate） * 最小码率（AlivcLiveConfig.setVideoMinBitrate）

```
180P 100~400
240P 100~600
360P 200~600
480P 200~800
540P 200~800
720P 200~1200
```

在网速正常的情况下，会按照目标码率进行推流，一旦遇到网速下降，码率控制模块会主动降低编码码率，以确保音视频流流畅。如果码率已经降低到最小码率，那么这个时候就会抛出网络差的错误警告。

二、观众端（AlivcVideoChatParter -> MediaPlayer）

- init->init
- startToPlay->prepareToPlay（异步接口）+play
- stopPlaying->stop
- release->destroy
- pause->pause
- resume->resume

通知处理（OnInfoListener->MediaPlayerInfoListener、MediaPlayerPrepareListener、MediaPlayerBufferingUpdateListener、MediaPlayerCompletedListener）

- MediaPlayerInfoListener ``` 当视频开始播放，用户需要知道视频的相关信息，可以注册该事件。

public interface MediaPlayerInfoListener{ void onInfo(int what, int extra); } 参数：

what：获取到的播放信息或警告的类型.播放的相关信息有：

• MEDIA_ INFO_ UNKNOW：未知信息

• MEDIA_ INFO_ BUFFERING_ START：当开始缓冲时，收到该信息

• MEDIA_ INFO_ BUFFERING_ END：缓冲结束时收到该信息

extra：对播放信息的额外表述。 `* MediaPlayerPrepareListener` 当调用prepareAsync后,视频准备完成后会发送准备完成事件，用户需要注册该事件，以便获取到该事件通知，在准备完成后调用start接口进行视频播放。

public interface MediaPlayerPrepareListener { void onPrepared();
}

```
* MediaPlayerBufferingUpdateListener
```

当网络下载速度较慢来不及播放时，会发送下载缓冲进度通知。

public interface MediaPlayerBufferingUpdateListener { void onBufferingUpdateListener(int percent); } 参数:

percent: 目前视频缓冲的进度，范围为0-100，100代表缓冲完成，0代表缓冲开始。
`* MediaPlayerCompletedListener` 当视频播放完成后，会发出该事件通知消息，用户需要注册该事件，在播放完成后完成相关清理工作。

public interface MediaPlayerCompletedListener { void onCompleted(); } ```

错误通知（OnErrorListener->MediaPlayerErrorListener）

当视频播放出现错误后,会发出该事件通知消息，用户需要注册该事件通知，以便在出现错误后给出相关错误提示。

```
public interface MediaPlayerErrorListener {
    void onError(int what, int extra);
}
```

参数：对应错误码描述

- ALIVC_ERR_INVALID_INPUTFILE(4003,"无效的输入，请检查输入地址或者网络链接")
- ALIVC_ERR_MEDIA_UNSUPPORTED(4018,"媒体源不支持或者无效")
- ALIVC_ERR_NO_SUPPORT_CODEC(4019,"视频编码格式不支持")
- ALIVC_ERROR_NO_INPUTFILE(4004,"没有设置视频源或视频地址不存在")
- ALIVC_ERR_READ_DATA_FAILED(4005,"读取视频源失败")
- ALIVC_ERROR_LOADING_TIMEOUT(4008,"视频加载超时，请检查网络状况")

对于ALIVC*ERR*INVALID*INPUTFILE*、*ALIVC*ERROR*LOADING*TIMEOUT 对应的处理 reset之后prepare

减少卡顿

```
setMaxBufferDuration

public void setMaxBufferDuration(int duration);

功能：设置直播最大缓冲时长。

参数：

duration：缓冲时长，单位毫秒。

备注：该函数仅对直播场景有效，主要用于缩短主播与观众之间的时间延迟。当缓冲区中的视频时长超过设置的 duration 时，播放器会自动丢弃部分音视频数据，以减少延迟。系统默认 rtmp、http flv 直播时 duration为8秒，HLS直播时duration为40秒。建议rtmp、http flv直播时，duration 的值超过 GOP 时长（即两个关键帧之间的时间长度）的 2 倍；HLS 直播时，duration 的值超过 m3u8 文件中所有 ts 分片的总时长。这样可以避免出现经常性的视频丢帧。
```

三、常见问题

- 主播推流
  - 需要自定义的初始配置（以下几项建议自定义，可以）
  - 重连：重连次数(5)/重连最小间隔时间(1000)

- 分辨率：360P/540P/720P
- 码率：初始码率/最小码率/目标码率
- 编码：硬编/软编
- 美颜：on/off
- 异步接口（如果在主线程中调用推流或播放sdk，那么建议采用异步接口）
- startPreviewAsync回调onPreviewStarted之后再调用startPushAsync
- startPushAsync回调onPushStarted之后再调用stopPush
- 错误回调
- System error：直接提示用户退出应用
- Sdk error：对于无麦克风权限和摄像头权限的错误，提示用户进行授权；对于其他错误，提示用户退出直播或restartPushAsync
- 网络回调
- 首次连接失败/重连失败/发送超时：这些情况可以提示用户设置网络、退出直播、reconnectPushAsync
- 对于网络差/网络恢复：在网络差时，在ui要有提示，让主播意识到当前网络推流比较卡；当网络恢复时，去掉提示；

- 观众播放
  - 设置播放参数
  - dropBufferDuration： 最大缓存时长，默认rtmp/http+flv 8秒、m3u840秒，可以根据不同用户的不同网络状况设置不同的参数
  - 为实现秒开，建议：
  - 1.提前获取播放的url；
  - 2.进入播放界面显示最少的ui；
  - 3.在首帧显示之后再显示其他的ui；
  - 在prepare的完成回调中调用play接口
  - 错误处理：
  - 针对流地址无效、流不存在的错误，提示用户退出直播间；
  - 针对下载超时的错误，进行重连；
  - 其他错误，提示用户退出直播间；
  - 重连的实现
  - stop的完成回调中执行prepare，在prepare的完成回调中执行play。 这个逻辑仅仅在需要重连的时候调用；
  - 在网络超时，及播放结束两个时间中需要进行重连；
  - 对于短时间断网或者网络切换，建议等待5秒钟之后进行重连；
  - 对于长时间断网，在等待5秒之后重连失败之后，等待服务端的推流成功通知再进行重连；如果这个时间超过20秒，那么结束直播。