直播 SDK

客户端 API

腾讯云

文档目录

客户端 API 升级指南

最近更新时间：2023-10-11 16:20:22

升级概述

为了更好地满足客户的接入需求，直播 SDK 在 API 以及支持的协议上做了重要的升级。API 2.0 相比与 API 1.0 在设计上更简洁也更实用，同时也增加了更多的能力，例如 RTC 的推流以及新的连麦方案。相对于原来的 RTMP 推流，RTC 推流具有更低地延迟，弱网下也有更好表现，全球覆盖使您出海业务更方便。另外 API 2.0 也增加了快直播播放，快直播（Live Event Broadcasting，LEB）是标准直播在超低延迟播放场景下的延伸，能够为观众提供毫秒级的极致直播观看体验。

下面从两个方面介绍升级步骤以及注意事项：

仅 API 升级

> ⚠ **注意**
>
> **适用范围：使用 `TXLivePusher` 和 `TXLivePlayer` API 的客户。**

直播 SDK 从 8.4 版本开始增加了一套新的 API `V2TXLivePusher` 和 `V2TXLivePlayer` 用来替代 `TXLivePusher` 和 `TXLivePlayer` ，主要优化点有以下几个方面：

- 去掉了 V1 一些冗余接口，例如 `TXLivePushConfig#setAutoAdjustBitrate` 等。
- 简化了 V1 一些复杂的使用方式，例如 `TXLivePushConfig` 类。
- 新增了一些实用的接口，例如 `V2TXLivePremierObserver#onLicenceLoaded` 用来告知用户 Licence 设置的结果以及失败原因。
- 新增了一些特性，例如支持 RTC 协议 URL 推流，支持 快直播播放 等。

为了方便开发者升级 API，请参考 V1 升级 V2 API 对比说明 。

> ⚠ **注意**
>
> `TXLivePusher` 和 `TXLivePlayer` 会在未来合适的时候废弃，请尽快升级。

推流协议升级

> ⚠ **适用范围**
>
> 对延迟以及弱网表现有更高追求的客户。

直播 SDK V1 API 只能使用 RTMP 协议推流，V2 API 新增了 RTC 协议的推流能力。RTC 协议基于 TRTC SDK ，在延迟以及弱网下有极佳的表现，详见 RTC 推流优势 。

针对不同的业务状态升级策略会有些不同。

如果您的业务不存在连麦场景

如果当前您的 App 只使用了 RTMP 协议推流，即没有连麦业务，升级到 RTC 协议推流是最简单的。

类型	注意事项

推流端	• 必要前提：您需要升级使用 V2TXLivePusher 推 RTC 流。
	• RTC 推流与 RTMP 推流仅在两个 API 使用上有参数差异，其他的 API 调用方式没有任何区别。
	  ○ 构造 V2TXLivePusher 对象时传入 RTC 协议参数：V2TXLiveDef.V2TXLiveMode.TXLiveMode_RTC，具体请参见 摄像头推流 # 初始化 V2TXLivePusher 组件 。
	  ○ 启动推流时传入 RTC 协议 URL，请参见 摄像头推流 – 启动和结束推流 。

播放端	在观众端您有三种方式可以播放，CDN 最便宜但延迟最大，RTC 最贵但延迟最小，快直播是延迟和费用之间的平衡。三种协议 API 除了播放的 URL 不同，调用方式是一致的。 • 播放 CDN 流：推荐您升级使用 V2TXLivePlayer 播放 。当前使用 TXLivePlayer 仍然是可以播放的。 • 播放 RTC 流：必须升级使用 V2TXLivePlayer 播放。RTC 拉流的 URL 请参见 自主拼装 RTC 连麦/PK URL 。 • 播放快直播流：必须升级使用 V2TXLivePlayer 播放，具体请参见 快直播拉流 。

如果您的业务即将接入连麦业务

如果当前您的 App 只使用了 RTMP 协议推流，即将接入连麦业务，此时您需要使用 RTC 实现连麦能力。
那么接入连麦后您的业务可能存在两种状态：

- 不推荐：连麦时使用 RTC 协议，非连麦使用 RTMP 协议。不推荐两种协议并存的方式，因为主播在连麦状态和非连麦状态之间切换时都需要重新推流，这会造成 CDN 观众在观看时会有中断的体验。
- 推荐：连麦和非连麦都使用 RTC 协议。推荐这种方式，主播在连麦和非连麦状态之间切换，CDN 观众观看时无感知。

接入时需要注意：

类型	注意事项
推流端	• 必要前提：您需要升级使用 V2TXLivePusher 推 RTC 流。 • RTC 推流与 RTMP 推流仅在两个 API 使用上有参数差异，其他的 API 调用方式没有任何区别。 • 构造 V2TXLivePusher 对象时传入 RTC 协议参数：V2TXLiveDef.V2TXLiveMode.TXLiveMode_RTC，具体请参见 摄像头推流 # 初始化 V2TXLivePusher 组件 。 • 启动推流时传入 RTC 协议 URL，请参见 摄像头推流 – 启动和结束推流 。

	• 实现连麦的具体步骤请参见 观众连麦 。
播放端	连麦的观众或者主播需要用 RTC 协议播放，非连麦的观众可以观看 CDN 流或者快直播流。三种协议 API 除了传入的 URL 不同，调用方式是一致的。 1. 必要前提：您需要升级使用 V2TXLivePlayer 来拉流播放。 2. 播放 RTC 流：连麦的双方必须播放 RTC 流，具体的 API 使用请参见 观众连麦 。 3. 播放 CDN 流：具体 API 使用请参见 标准直播拉流 。 4. 播放快直播流：具体 API 使用请参见 快直播拉流 。

如果您的业务已有连麦业务

如果当前您的 App 已经存在连麦业务，需要升级到 RTC 连麦，您可以通过以下步骤进行。

1. 先实现：
   - 实现新连麦：使用 RTC 实现连麦能力（具体方案请参见 即将接入连麦业务 ）。
   - 增加云控：增加云控来获取当前应该使用哪一种连麦方式。
2. 再并存：
   - 逐步升级含有新连麦方式的 App 版本。
   - 因为新老版本可能会有不通的地方，所以此时不能放开新连麦方式。
3. 最终上线：
   - 版本全覆盖时通过云控全部切换为新连麦方案。
   - 下线老连麦方式。

V1 升级 V2 API 对比说明

最近更新时间：2023-07-26 10:57:31

升级概述

为了更好地满足客户的接入需求，直播 SDK 在 API 以及支持的协议上做了重要的升级。API 2.0 相比与 API 1.0 在设计上更简洁也更实用。

TXLivePushConfig

V1 API	V2 API 对应接口	备注
setHomeOrientation	V2TXLivePusher.setVideoQuality	请参见 V2TXLiveVideoEncoderparam.videoResolutionMode
setVideoResolution		请参见 V2TXLiveVideoEncoderparam.videoResolution
setVideoFPS		请参见 V2TXLiveVideoEncoderparam.videoFps
setVideoBitrate		请参见 V2TXLiveVideoEncoderparam.videoBitrate
setMinVideoBitrate		请参见 V2TXLiveVideoEncoderparam.minVideoBitrate
setVideoEncodeGop	V2 API 去掉该接口	-
setAutoAdjustBitrate	V2 API 去掉该接口	-
setAutoAdjustStrategy	V2 API 去掉该接口	-
setMaxVideoBitrate	V2 API 去掉该接口	-
setTouchFocus	V2TXLivePusher.getDeviceManager.setCameraFocusPosition	-
setEnableZoom	V2 API 去掉该接口	-

setWatermark(Bitmap watermark, int x, int y)	V2TXLivePusher.setWatermark	–
setWatermark(Bitmap watermark, float x, float y, float width)		–
setPauseImg(Bitmap img)	V2TXLivePusher.startVirtualCamera	V2 中有关垫片推流的场景使用主要会用到以下接口： • startMicrophone 打开麦克风 • stopMicrophone 关闭麦克风 • startCamera 打开本地摄像头 • stopCamera 关闭本地摄像头 • startVirtualCamera 开启图片推流 • stopVirtualCamera 关闭图片推流 • pauseAudio 暂停推流器的音频流 • resumeAudio 恢复推流器的音频流 • pauseVideo 暂停推流器的视频流 • resumeVideo 恢复视频推流 **场景1** **主播上行音视频过程中，切后台推送默认背景图，回到前台继续推摄像头数据。** 1. **主播上行音视频：** startMicrophone + startCamera + startPush 2. **主播切后台，不上行摄像头，上行主播已离开的提示图片：** stopCamera + startVirtualCamera 3. **主播回到前台，重新上行摄像头：** stopVirtualCamera + startCamera
setPauseImg(int time, int fps)	V2 API 去掉该接口	
setPauseFlag	V2 API 细分了暂停音频视频推流的接口，因此去掉该接口。	

场景2

主播以纯音频开播，可选择推送一张封面图；也可中途恢复推送摄像头数据；或者停止上行视频，只在本地预览摄像头数据。

1. 主播纯音频推垫片：startMicrophone + startVirtualCamera + startPush
2. 主播打开摄像头上行摄像头数据：stopVirtualCamera + startCamera
3. camera 不关闭，停止上行视频：pauseVideo
4. camera 不关闭，恢复上行视频：resumeVideo

场景3

主播直播过程中需要调整视频画面如美颜等，短暂暂停上行视频，同时保持摄像头预览。

1. 主播上行音视频：startMicrophone + startCamera + startPush
2. camera 不关闭，停止上行视频：pauseVideo
3. camera 不关闭，恢复上行视频：resumeVideo

setAudioSampleRate setAudioChannels	V2TXLivePusher.setAudioQuality	V2TXLivePusher.setAudioQuality 接口可针对以下场景设置： 1. V2TXLiveAudioQualitySpeech 采样率：16k、单声道适合在线会议，语音通话 2. V2TXLiveAudioQualityDefault 采样率：48k、单声道如无特殊

		需求推荐 3. V2TXLiveAudioQualityMusic 采样率：48k、双声道 、全频带适合需要高保真传输音乐的场景，比如K 歌、音乐直播
enableAudioEarMonitoring	V2TXLivePusher.getAudioEffectManager.enableVoiceEarMonitor	-
enablePureAudioPush	V2 API 去掉该接口	-
enableScreenCaptureAutoRotate	V2 API 去掉该接口	-
enableHighResolutionCaptureMode	V2 API 去掉该接口	-
setVideoEncoderXMirror	V2 API 去掉该接口	-
setConnectRetryCount	V2 API 去掉该接口，重连逻辑由内部自动处理	-
setConnectRetryInterval	V2 API 去掉该接口，重连逻辑由内部自动处理	-
setCustomModeType	开启自定义音频采集：V2TXLivePusher.enableCustomAudioCapture 开始自定义视频采集：V2TXLivePusher.enableCustomVideoCapture	-
enableANS	V2 API 去掉该接口，由内部自动处理	-
enableAEC	V2 API 去掉该接口，由内部自动处理	-
enableAGC	V2 API 去掉该接口，由内部自动处理	-
setVolumeType	V2TXLivePusher.getDeviceManager.setSystemVolumeType	-

setHardwareAcceleration	V2 API 去掉该接口，由内部自动处理，优先使用硬编	-
enableVideoHardEncoderMainProfile	V2 API 去掉该接口，由内部自动处理	-
setMetaData	V2 API 去掉该接口	-
enableMute	V2TXLivePusher.pauseAudio	-
setFrontCamera	V2TXLivePusher.getDeviceManager.switchCamera	-
setBeautyFilter	• V2TXLivePusher.getBeautyManager.setBeautyLevel • V2TXLivePusher.getBeautyManager.setWhitenessLevel • V2TXLivePusher.getBeautyManager.setRuddyLevel	-
setEyeScaleLevel	V2TXLivePusher.getBeautyManager.setEyeScaleLevel	-
setFaceSlimLevel	V2TXLivePusher.getBeautyManager.setFaceSlimLevel	-
enableNearestIP	V2 API 去掉该接口，由内部自动处理	-
setRtmpChannelType	V2 API 去掉该接口，由内部自动处理	-
setCustomVideoPreProcessLibrary	V2 API 去掉该接口	-
setCustomAudioPreProcessLibrary	V2 API 去掉该接口	-

TXLivePusher

V1 API	V2 API 对应接口	备注

setPushListener	V2TXLivePusher.setObserver	-
setAudioVolumeEvaluationListener		V2TXLivePusherObserver.onMicrophoneVolumeUpdate
enableAudioVolumeEvaluation	V2TXLivePusher.enableVolumeEvaluation	
startCameraPreview	V2TXLivePusher.startCamera	-
stopCameraPreview	V2TXLivePusher.stopCamera	-
startPusher	V2TXLivePusher.startPush	-
stopPusher	V2TXLivePusher.stopPush	-
pausePusher	V2TXLivePusher.pauseAudio V2TXLivePusher.pauseVideo	-
resumePusher	V2TXLivePusher.resumeAudio V2TXLivePusher.resumeVideo	-
isPushing	V2TXLivePusher.isPushing	-
setVideoQuality	V2TXLivePusher.setVideoQuality	adjustBitrate 和 adjustResolution 不再支持，由内部自动处理
switchCamera	V2TXLivePusher.getDeviceManager.switchCamera	-
frontCamera	V2TXLivePusher.getDeviceManager.isFrontCamera	-
setMirror	V2TXLivePusher.setRenderMirror	-
setRenderRotation	V2TXLivePusher.setRenderRotation	-
turnOnFlashLight	V2TXLivePusher.getDeviceManager.enableCameraTorch	-
getMaxZoom	V2TXLivePusher.getDeviceManager.getCameraZoomMaxRa	-

	tio	
setZoom	V2TXLivePusher.getDeviceManager.setCameraZoomRatio	-
setFocusPosition	V2TXLivePusher.getDeviceManager.setCameraFocusPosition	-
setExposureCompensation	V2TXLivePusher.getDeviceManager.setExposureCompensation	-
getBeautyManager	V2TXLivePusher.getBeautyManager	-
setMute	V2TXLivePusher.pauseAudio	-
getAudioEffectManager	V2TXLivePusher.getAudioEffectManager	-
setVideoRecordListener	V2 API 去掉该接口	-
startRecord	V2 API 去掉该接口	-
stopRecord	V2 API 去掉该接口	-
snapshot	V2TXLivePusher.snapshot	V2TXLivePusherObserver.onSnapshotComplete
sendCustomVideoTexture	V2TXLivePusher.sendCustomVideoFrame	V2TXLiveVideoFrame.pixelFormat = V2TXLivePixelFormatTexture2D V2TXLiveVideoFrame.pixelFormat.bufferType = V2TXLiveBufferTypeTexture
sendCustomVideoData		V2TXLiveVideoFrame.pixelFormat.bufferType = V2TXLiveBufferTypeByteArray
sendCustomPCMData	V2TXLivePusher.sendCustomAudioFrame	-
setVideoProcessLis	V2TXLivePusher.enableCusto	V2TXLivePusherObserver.o

tener	mVideoProcess	nProcessVideoFrame
setAudioProcessListener	V2TXLivePusher.enableAudioProcessObserver	V2TXLivePusherObserver.onProcessAudioFrame
setSurface	V2TXLivePusher.setRenderView	-
setSurfaceSize	V2 API 去掉该接口	-
sendMessageEx	V2TXLivePusher.sendSeiMessage	-
onLogRecord	V2 API 去掉该接口	-
startScreenCapture	V2TXLivePusher.startScreenCapture	-
stopScreenCapture	V2TXLivePusher.stopScreenCapture	-
pauseScreenCapture	V2 API 去掉该接口	-
resumeScreenCapture	V2 API 去掉该接口	-
getBeautyManager	V2TXLivePusher.getBeautyManager	-
getAudioEffectManager	V2TXLivePusher.getAudioEffectManager	-
callExperimentalAPI	V2TXLivePusher.setProperty	-

ITXLivePushListener

V1 API	V2 API 对应接口	备注
onPushEvent	• V2TXLivePusherObserver.onError • V2TXLivePusherObserver.onWarning • V2TXLivePusherObserver.onPushStatusUpdate	V2 对 V1 onPushEvent 回调进行了重新设计，细分为 onError、onWarning、onPushStatusUpdate 三个模块： • `onError` 中会抛出阻塞直播推流的错误信息 • `onWarning` 中处理一些不会阻塞直播推流但是影响体验的警告信息 • `onPushStatusUpdate` 是直播推流状态的改变信息回调

onNetStatus	V2TXLivePusherObserver.onStatisticsUpdate	-

TXLivePlayConfig

V1 API	V2 API 对应接口	备注
setCacheTime	V2TXLivePlayer.setCacheParams	当正确设置 V2TXLivePlayer.setCacheParams 中 minTime 和 maxTime 时，播放器会按照预设的最小值最大值范围内自动调整
setAutoAdjustCacheTime		
setMaxAutoAdjustCacheTime		
setMinAutoAdjustCacheTime		
setVideoBlockThreshold	V2 API 去掉该接口	-
setConnectRetryCount	V2TXLivePlayer.setProperty	kV2MaxNumberOfReconnection
setConnectRetryInterval		kV2SecondsBetweenReconnection
setHeaders		kV2SetHeaders
enableAEC	V2 API 去掉该接口，由内部自动处理	-
setEnableMessage	V2TXLivePlayer.enableReceiveSeiMessage	V2 开启接收 SEI 消息后通过 V2TXLivePlayerObserver.onReceiveSeiMessage 回调接收到 SEI 消息
setEnableMetaData	V2 API 去掉该接口，默认开启	-
setFlvSessionKey	V2 API 去掉该接口	-
setEnableNearestIP	V2 API 去掉该接口，由内部自动处理	-
setRtmpChannelType	V2 API 去掉该接口，由内部自动处理	-

setCacheFolderPath	V2 API 去掉该接口	-
setMaxCacheItems	V2 API 去掉该接口	-

TXLivePlayer

V1 API	V2 API 对应接口	备注
setPlayListener	V2TXLivePlayer.setObserver	
setAudioRawDataListener		V2TXLivePlayerObserver.onPlayoutAudioFrame
setAudioVolumeEvaluationListener		V2TXLivePlayerObserver.onPlayoutVolumeUpdate
enableAudioVolumeEvaluation	V2TXLivePlayer.enableVolumeEvaluation	
recordDelegate	V2 API 去掉该接口	
isAutoPlay	V2 API 去掉该接口	
setPlayerView	V2TXLivePlayer.setRenderView	
setSurface		
setSurfaceSize	V2 API 去掉该接口	
startPlay	V2TXLivePlayer.startLivePlay	
stopPlay	V2TXLivePlayer.stopPlay	
isPlaying	V2TXLivePlayer.isPlaying	
pause	V2TXLivePlayer.pauseAudio V2TXLivePlayer.pauseVideo	
resume	V2TXLivePlayer.resumeAudio V2TXLivePlayer.resumeVideo	
setRenderRotation	V2TXLivePlayer.setRenderRotation	

setRenderMode	V2TXLivePlayer.setRenderFillMode	
enableHardwareDecode	V2 API 去掉该接口，由内部自动处理，优先硬解	
snapshot	V2TXLivePlayer.snapshot	V2TXLivePlayerObserver.onSnapshotComplete
getCurrentRenderPts	V2 API 去掉该接口	
setMute	V2TXLivePlayer.pauseAudio V2TXLivePlayer.resumeAudio	
setVolume	V2TXLivePlayer.setPlayoutVolume	
setAudioRoute	V2TXLivePlayer.getDeviceManager.setAudioRoute	
prepareLiveSeek	V2 API 去掉该接口	
resumeLive	V2 API 去掉该接口	
seek	V2 API 去掉该接口	
setAutoPlay	V2 API 去掉该接口	
startRecord	V2 API 去掉该接口	
stopRecord	V2 API 去掉该接口	
setRate	V2 API 去掉该接口	
switchStream	V2TXLivePlayer.switchStream	
callExperimentalAPI	V2 API 去掉该接口	

ITXLivePlayListener

V1 API	V2 API 对应接口	备注
onPlayEve	• V2TXLivePlayerObserver.onError	V2 对 V1 onPushEvent 回调进行了重新设计，细分为 onError、onWarning、onVideoPlaying、

nt	• V2TXLivePlayerObserver.onWarning • V2TXLivePlayerObserver.onVideoPlaying • V2TXLivePlayerObserver.onVideoLoading • V2TXLivePlayerObserver.onAudioPlaying • V2TXLivePlayerObserver.onAudioLoading	onVideoLoading、onAudioPlaying、onAudioLoading 六个模块： • onError 中会抛出阻塞拉流的错误信息 • onWarning 中处理一些不会阻塞拉流但是影响体验的警告信息 • onAudioPlaying 和 onAudioLoading 是拉流音频状态的改变信息回调 • onVideoPlaying 和 onVideoLoading 是拉流视频状态的改变信息回调
onNetStatus	V2TXLivePlayerObserver.onStatisticsUpdate	–

iOS&Mac API 概览

最近更新时间：2024-01-16 14:35:21

API OVERVIEW

V2 推流器相关接口

函数列表	描述
setObserver:	设置推流器回调
setRenderView:	设置本地摄像头预览 View
setRenderMirror:	设置本地摄像头预览镜像
setEncoderMirror:	设置视频编码镜像
setRenderRotation:	设置本地摄像头预览画面的旋转角度
setRenderFillMode:	设置本地摄像头预览画面的填充模式
startCamera:	打开本地摄像头
stopCamera	关闭本地摄像头
startMicrophone	打开麦克风
stopMicrophone	关闭麦克风
startVirtualCamera:	开启图片推流
stopVirtualCamera	关闭图片推流
startScreenCapture:	开始全系统的屏幕分享（该接口支持 iOS 11.0 及以上的 iPhone 和 iPad）
stopScreenCapture	关闭屏幕采集
pauseAudio	静音本地音频
resumeAudio	取消静音本地音频
pauseVideo	暂停推流器的视频流

resumeVideo	恢复推流器的视频流
startPush:	开始音视频数据推流
stopPush	停止推送音视频数据
isPushing	当前推流器是否正在推流中
setAudioQuality:	设置推流音频质量
setVideoQuality:	设置推流视频编码参数
getAudioEffectManager	获取音效管理对象
getBeautyManager	获取美颜管理对象
getDeviceManager	获取设备管理对象
snapshot	截取推流过程中的本地画面
setWatermark:x:y:scale:	设置推流器水印。默认情况下，水印不开启
enableVolumeEvaluation:	启用采集音量大小提示
enableCustomVideoProcess:pixelFormat:bufferType:	开启/关闭自定义视频处理
enableCustomVideoCapture:	开启/关闭自定义视频采集
enableCustomAudioCapture:	开启/关闭自定义音频采集
sendCustomVideoFrame:	在自定义视频采集模式下，将采集的视频数据发送到SDK
sendCustomAudioFrame:	在自定义音频采集模式下，将采集的音频数据发送到SDK
enableAudioProcessObserver:format:	开启/关闭对经过前处理后的本地音频帧的监听回调
sendSeiMessage:data:	发送 SEI 消息
showDebugView:	显示仪表盘
setProperty:value:	调用 V2TXLivePusher 的高级 API 接口。
setMixTranscodingConfig:	设置云端的混流转码参数
startLocalRecording:	开始录制音视频流

stopLocalRecording	停止录制音视频流

直播推流器事件回调

函数列表	描述
onError:message:extraInfo:	直播推流器错误通知，推流器出现错误时，会回调该通知
onWarning:message:extraInfo:	直播推流器警告通知
onCaptureFirstAudioFrame	首帧音频采集完成的回调通知
onCaptureFirstVideoFrame	首帧视频采集完成的回调通知
onMicrophoneVolumeUpdate:	麦克风采集音量值回调
onPushStatusUpdate:message:extraInfo:	推流器连接状态回调通知
onStatisticsUpdate:	直播推流器统计数据回调
onSnapshotComplete:	截图回调
onProcessAudioFrame:	本地采集并经过音频模块前处理、音效处理和混 BGM 后的音频数据回调
onProcessVideoFrame:dstFrame:	自定义视频处理回调
onGLContextDestroyed	SDK 内部的 OpenGL 环境的销毁通知
onSetMixTranscodingConfig:message:	设置云端的混流转码参数的回调，对应于 setMixTranscodingConfig 接口
onScreenCaptureStarted	当屏幕分享开始时，SDK 会通过此回调通知
onScreenCaptureStopped:	当屏幕分享停止时，SDK 会通过此回调通知
onLocalRecordBegin:storagePath:	录制任务开始的事件回调
onLocalRecording:storagePath:	录制任务正在进行中的进展事件回调
onLocalRecordComplete:storagePath:	录制任务已经结束的事件回调

V2 播放器相关接口

函数列表	描述
setObserver:	设置播放器回调
setRenderView:	设置播放器的视频渲染 View，该控件负责显示视频内容
setRenderRotation:	设置播放器画面的旋转角度
setRenderFillMode:	设置画面的填充模式
startLivePlay:	开始播放音视频流
stopPlay	停止播放音视频流
isPlaying	播放器是否正在播放中
pauseAudio	暂停播放器的音频流
resumeAudio	恢复播放器的音频流
pauseVideo	暂停播放器的视频流
resumeVideo	恢复播放器的视频流
setPlayoutVolume:	设置播放器音量
setCacheParams:maxTime:	设置播放器缓存自动调整的最小和最大时间（单位：秒）
switchStream:	直播流无缝切换，支持 FLV 和 LEB
getStreamList	获取码流信息
enableVolumeEvaluation:	启用播放音量大小提示
snapshot	截取播放过程中的视频画面
enableObserveVideoFrame:pixelFormat:bufferType:	开启/关闭对视频帧的监听回调
enableObserveAudioFrame:	开启/关闭对音频数据的监听回调
enableReceiveSeiMessage:payloadType:	开启接收 SEI 消息
enablePictureInPicture:	开启画中画功能，仅支持直播和快直播播放
showDebugView:	是否显示播放器状态信息的调试浮层

setProperty:value:	调用 V2TXLivePlayer 的高级 API 接口
startLocalRecording:	开始录制音视频流
stopLocalRecording	停止录制音视频流

直播播放器事件回调

函数列表	描述
onError:code:message:extraInfo:	直播播放器错误通知，播放器出现错误时，会回调该通知
onWarning:code:message:extraInfo:	直播播放器警告通知
onVideoResolutionChanged:width:height:	直播播放器分辨率变化通知
onConnected:extraInfo:	已经成功连接到服务器
onVideoPlaying:firstPlay:extraInfo:	视频播放事件
onAudioPlaying:firstPlay:extraInfo:	音频播放事件
onVideoLoading:extraInfo:	视频加载事件
onAudioLoading:extraInfo:	音频加载事件
onPlayoutVolumeUpdate:volume:	播放器音量大小回调
onStatisticsUpdate:statistics:	直播播放器统计数据回调
onSnapshotComplete:image:	截图回调
onRenderVideoFrame:frame:	自定义视频渲染回调
onPlayoutAudioFrame:frame:	音频数据回调
onReceiveSeiMessage:payloadType:data:	收到 SEI 消息的回调，发送端通过 V2TXLivePusher 中的 `sendSeiMessage` 来发送 SEI 消息
onStreamSwitched:url:code:	分辨率无缝切换回调
onPictureInPictureStateUpdate:state:message:extraInfo:	画中画状态变更回调
onLocalRecordBegin:errCode:storagePath:	录制任务开始的事件回调

onLocalRecording:durationMs:storagePath:	录制任务正在进行中的进展事件回调
onLocalRecordComplete:errCode:storagePath:	录制任务已经结束的事件回调

V2TXLive 高级接口

函数列表	描述
getSDKVersionStr	获取 SDK 版本号
setObserver:	设置 V2TXLivePremier 回调接口
setLogConfig:	设置 Log 的配置信息
setEnvironment:	设置 SDK 接入环境
setLicence:key:	设置 SDK 的授权 License
setSocks5Proxy:port:username:password:config:	设置 SDK socks5 代理配置
enableAudioCaptureObserver:format:	开启/关闭对音频采集数据的监听回调（可读写）
enableAudioPlayoutObserver:format:	开启/关闭对最终系统要播放出的音频数据的监听回调
enableVoiceEarMonitorObserver:	开启/关闭耳返音频数据的监听回调
setUserId:	设置 userId
callExperimentalAPI:	调用实验性 API 接口

V2TXLive 高级回调接口

函数列表	描述
onLog:log:	自定义 Log 输出回调接口
onLicenceLoaded:Reason:	setLicence 接口回调
onCaptureAudioFrame:	本地麦克风采集到的音频数据回调

onPlayoutAudioFrame:	将各路待播放音频混合之后并在最终提交系统播放之前的数据回调
onVoiceEarMonitorAudioFrame:	耳返的音频数据

人声相关的特效接口

函数列表	描述
enableVoiceEarMonitor:	开启耳返
setVoiceEarMonitorVolume:	设置耳返音量
setVoiceReverbType:	设置人声的混响效果
setVoiceChangerType:	设置人声的变声特效
setVoiceVolume:	设置语音音量
setVoicePitch:	设置语音音调

背景音乐的相关接口

函数列表	描述
startPlayMusic:onStart:onProgress:onComplete:	开始播放背景音乐
stopPlayMusic:	停止播放背景音乐
pausePlayMusic:	暂停播放背景音乐
resumePlayMusic:	恢复播放背景音乐
setAllMusicVolume:	设置所有背景音乐的本地音量和远端音量的大小
setMusicPublishVolume:volume:	设置某一首背景音乐的远端音量的大小
setMusicPlayoutVolume:volume:	设置某一首背景音乐的本地音量的大小
setMusicPitch:pitch:	调整背景音乐的音调高低
setMusicSpeedRate:speedRate:	调整背景音乐的变速效果

getMusicCurrentPosInMS:	获取背景音乐的播放进度（单位：毫秒）
getMusicDurationInMS:	获取背景音乐的总时长（单位：毫秒）
seekMusicToPosInMS:pts:	设置背景音乐的播放进度（单位：毫秒）
setMusicScratchSpeedRate:speedRate:	调整搓碟的变速效果
preloadMusic:onProgress:onError:	预加载背景音乐
getMusicTrackCount:	获取背景音乐的音轨数量
setMusicTrack:track:	指定背景音乐的播放音轨

美颜相关接口

函数列表	描述
setBeautyStyle:	设置美颜（磨皮）算法
setBeautyLevel:	设置美颜级别
setWhitenessLevel:	设置美白级别
enableSharpnessEnhancement:	开启清晰度增强
setRuddyLevel:	设置红润级别
setFilter:	设置色彩滤镜效果
setFilterStrength:	设置色彩滤镜的强度
setGreenScreenFile:	设置绿幕背景视频
setEyeScaleLevel:	设置大眼级别
setFaceSlimLevel:	设置瘦脸级别
setFaceVLevel:	设置 V 脸级别
setChinLevel:	设置下巴拉伸或收缩
setFaceShortLevel:	设置短脸级别
setFaceNarrowLevel:	设置窄脸级别
setNoseSlimLevel:	设置瘦鼻级别

setEyeLightenLevel:	设置亮眼级别
setToothWhitenLevel:	设置牙齿美白级别
setWrinkleRemoveLevel:	设置祛皱级别
setPounchRemoveLevel:	设置祛眼袋级别
setSmileLinesRemoveLevel:	设置法令纹去除级别
setForeheadLevel:	设置发际线调整级别
setEyeDistanceLevel:	设置眼距
setEyeAngleLevel:	设置眼角调整级别
setMouthShapeLevel:	设置嘴型调整级别
setNoseWingLevel:	设置鼻翼调整级别
setNosePositionLevel:	设置鼻子位置
setLipsThicknessLevel:	设置嘴唇厚度
setFaceBeautyLevel:	设置脸型
setMotionTmpl:inDir:	选择 AI 动效挂件
setMotionMute:	是否在动效素材播放时静音

音视频设备相关的类型定义

函数列表	描述
onDeviceChanged:type:state:	本地设备的通断状态发生变化（仅适用于桌面系统）

设备操作接口

函数列表	描述
isFrontCamera	判断当前是否为前置摄像头（仅适用于移动端）

switchCamera:	切换前置或后置摄像头（仅适用于移动端）
isCameraZoomSupported	查询当前摄像头是否支持缩放（仅适用于移动端）
getCameraZoomMaxRatio	获取摄像头的最大缩放倍数（仅适用于移动端）
setCameraZoomRatio:	设置摄像头的缩放倍数（仅适用于移动端）
isAutoFocusEnabled	查询是否支持自动识别人脸位置（仅适用于移动端）
enableCameraAutoFocus:	开启自动对焦功能（仅适用于移动端）
setCameraFocusPosition:	设置摄像头的对焦位置（仅适用于移动端）
isCameraTorchSupported	查询是否支持开启闪光灯（仅适用于移动端）
enableCameraTorch:	开启/关闭闪光灯，也就是手电筒模式（仅适用于移动端）
setAudioRoute:	设置音频路由（仅适用于移动端）
setExposureCompensation:	设置摄像头的曝光参数，取值范围从-1到1
getDevicesList:	获取设备列表（仅适用于桌面端）
setCurrentDevice:deviceId:	设置当前要使用的设备（仅适用于桌面端）
getCurrentDevice:	获取当前正在使用的设备（仅适用于桌面端）
setCurrentDeviceVolume:deviceType:	设置当前设备的音量（仅适用于桌面端）
getCurrentDeviceVolume:	获取当前设备的音量（仅适用于桌面端）
setCurrentDeviceMute:deviceType:	设置当前设备的静音状态（仅适用于桌面端）
getCurrentDeviceMute:	获取当前设备的静音状态（仅适用于桌面端）
enableFollowingDefaultAudioDevice:enable:	设置 SDK 使用的音频设备根据跟随系统默认设备（仅适用于桌面端）
startCameraDeviceTest:	开始摄像头测试（仅适用于桌面端）
stopCameraDeviceTest	结束摄像头测试（仅适用于桌面端）
startMicDeviceTest:	开始麦克风测试（仅适用于桌面端）
startMicDeviceTest:playback:	开始麦克风测试（仅适用于桌面端）

stopMicDeviceTest	结束麦克风测试（仅适用于桌面端）
startSpeakerDeviceTest:	开始扬声器测试（仅适用于桌面端）
stopSpeakerDeviceTest	结束扬声器测试（仅适用于桌面端）
setObserver:	设备热插拔回调（仅适用于 Mac 系统）

弃用接口

函数列表	描述
setSystemVolumeType:	设置系统音量类型（仅适用于移动端）

V2TXLivePusher

最近更新时间：2024-03-07 15:14:11

Module: V2TXLivePusher @ TXLiteAVSDK

Function: 腾讯云直播推流器

功能

腾讯云直播推流器

介绍

主要负责将本地的音频和视频画面进行编码，并推送到指定的推流地址，支持任意的推流服务端。
推流器包含如下能力：

- 自定义的视频采集，让您可以根据项目需要定制自己的音视频数据源。
- 美颜、滤镜、贴纸，包含多套美颜磨皮算法（自然&光滑）和多款色彩空间滤镜（支持自定义滤镜）。
- Qos 流量控制技术，具备上行网络自适应能力，可以根据主播端网络的具体情况实时调节音视频数据量。
- 脸型调整、动效挂件，支持基于优图 AI 人脸识别技术的大眼、瘦脸、隆鼻等脸型微调以及动效挂件效果，只需要购买 优图 License 就可以轻松实现丰富的直播效果。

V2TXLivePusher

V2TXLivePusher

函数列表	描述
setObserver:	设置推流器回调
setRenderView:	设置本地摄像头预览 View
setRenderMirror:	设置本地摄像头预览镜像
setEncoderMirror:	设置视频编码镜像
setRenderRotation:	设置本地摄像头预览画面的旋转角度

setRenderFillMode:	设置本地摄像头预览画面的填充模式
startCamera:	打开本地摄像头
stopCamera	关闭本地摄像头
startMicrophone	打开麦克风
stopMicrophone	关闭麦克风
startVirtualCamera:	开启图片推流
stopVirtualCamera	关闭图片推流
startScreenCapture:	开始全系统的屏幕分享（该接口支持 iOS 11.0 及以上的 iPhone 和 iPad）
stopScreenCapture	关闭屏幕采集
pauseAudio	静音本地音频
resumeAudio	取消静音本地音频
pauseVideo	暂停推流器的视频流
resumeVideo	恢复推流器的视频流
startPush:	开始音视频数据推流
stopPush	停止推送音视频数据
isPushing	当前推流器是否正在推流中
setAudioQuality:	设置推流音频质量
setVideoQuality:	设置推流视频编码参数
getAudioEffectManager	获取音效管理对象
getBeautyManager	获取美颜管理对象
getDeviceManager	获取设备管理对象
snapshot	截取推流过程中的本地画面
setWatermark:x:y:scale:	设置推流器水印。默认情况下，水印不开启
enableVolumeEvaluation:	启用采集音量大小提示
enableCustomVideoProcess:pixelForm	开启/关闭自定义视频处理

at:bufferType:	
enableCustomVideoCapture:	开启/关闭自定义视频采集
enableCustomAudioCapture:	开启/关闭自定义音频采集
sendCustomVideoFrame:	在自定义视频采集模式下，将采集的视频数据发送到SDK
sendCustomAudioFrame:	在自定义音频采集模式下，将采集的音频数据发送到SDK
enableAudioProcessObserver:format:	开启/关闭对经过前处理后的本地音频帧的监听回调
sendSeiMessage:data:	发送 SEI 消息
showDebugView:	显示仪表盘
setProperty:value:	调用 V2TXLivePusher 的高级 API 接口。
setMixTranscodingConfig:	设置云端的混流转码参数
startLocalRecording:	开始录制音视频流
stopLocalRecording	停止录制音视频流

setObserver:

setObserver:

- (void)setObserver: (id<V2TXLivePusherObserver>)observer

设置推流器回调

通过设置回调，可以监听 V2TXLivePusher 推流器的一些回调事件，
包括推流器状态、音量回调、统计数据、警告和错误信息等。

参数	描述
observer	推流器的回调目标对象，更多信息请查看 V2TXLivePusherObserver 。

setRenderView:

setRenderView:

```
- (V2TXLiveCode)setRenderView:(TXView *)view
```

设置本地摄像头预览 View

本地摄像头采集到的画面，经过美颜、脸型调整、滤镜等多种效果叠加之后，最终会显示到传入的 View 上。

参数	描述
view	本地摄像头预览 View。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK：成功。

setRenderMirror:

setRenderMirror:

```
- (V2TXLiveCode)setRenderMirror:(V2TXLiveMirrorType)mirrorType
```

设置本地摄像头预览镜像

本地摄像头分为前置摄像头和后置摄像头，系统默认情况下，是前置摄像头镜像，后置摄像头不镜像，这里可以修改前置后置摄像头的默认镜像类型。

参数	描述
mirrorType	摄像头镜像类型 V2TXLiveMirrorType 。 • V2TXLiveMirrorTypeAuto 【默认值】：默认镜像类型. 在这种情况下，前置摄像头的画面是镜像的，后置摄像头的画面不是镜像的。 • V2TXLiveMirrorTypeEnable: 前置摄像头 和 后置摄像头，都切换为镜像模式。 • V2TXLiveMirrorTypeDisable: 前置摄像头 和 后置摄像头，都切换为非镜像模式。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setEncoderMirror:

setEncoderMirror:

```
- (V2TXLiveCode)setEncoderMirror:(BOOL)mirror
```

设置视频编码镜像

参数	描述
mirror	是否镜像。 • NO【默认值】：播放端看到的是非镜像画面。 • YES: 播放端看到的是镜像画面。

> **注意**
> 编码镜像只影响观众端看到的视频效果。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setRenderRotation:

setRenderRotation:

```
- (V2TXLiveCode)setRenderRotation:(V2TXLiveRotation)rotation
```

设置本地摄像头预览画面的旋转角度

参数	描述
rotation	预览画面的旋转角度 V2TXLiveRotation 。 • V2TXLiveRotation0【默认值】：0度，不旋转。

	• V2TXLiveRotation90: 顺时针旋转90度。 • V2TXLiveRotation180: 顺时针旋转180度。 • V2TXLiveRotation270: 顺时针旋转270度。

注意
只旋转本地预览画面，不影响推流出去的画面。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setRenderFillMode:

setRenderFillMode:

```
- (V2TXLiveCode)setRenderFillMode:(V2TXLiveFillMode)mode
```

设置本地摄像头预览画面的填充模式

参数	描述
mode	画面填充模式 V2TXLiveFillMode 。 • V2TXLiveFillModeFill【默认值】: 图像铺满屏幕，不留黑边，如果图像宽高比不同于屏幕宽高比，部分画面内容会被裁剪掉。 • V2TXLiveFillModeFit: 图像适应屏幕，保持画面完整，但如果图像宽高比不同于屏幕宽高比，会有黑边的存在。 • V2TXLiveFillModeScaleFill: 图像拉伸铺满，因此长度和宽度可能不会按比例变化。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startCamera:

startCamera:

```
- (V2TXLiveCode)startCamera:(BOOL)frontCamera
```

打开本地摄像头

参数	描述
frontCamera	是否为前置摄像头。 • YES【默认值】：切换到前置摄像头。 • NO: 切换到后置摄像头。

注意
startVirtualCamera，startCamera，startScreenCapture，同一 Pusher 实例下，仅有一个采集源可以上行，不同采集源之间切换，请先关闭前一采集源，再开启后一采集源，保证同一采集源的开启和关闭是成对调用的。比如：采集源从Camera切换到VirtualCamera，调用顺序是 startCamera -> stopCamera -> startVirtualCamera。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

stopCamera

stopCamera

关闭本地摄像头

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startMicrophone

startMicrophone

打开麦克风

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

stopMicrophone

stopMicrophone

关闭麦克风

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startVirtualCamera:

startVirtualCamera:

```
- (V2TXLiveCode)startVirtualCamera:      (TXImage *)image
```

开启图片推流

参数	描述
image	图片。

> **注意**
> startVirtualCamera，startCamera，startScreenCapture，同一 Pusher 实例下，仅有一个采集源可以上行，不同采集源之间切换，请先关闭前一采集源，再开启后一采集源，保证同一采集源的开启和关闭是成对调用的。比如：采集源从Camera切换到VirtualCamera，调用顺序是 startCamera -> stopCamera -> startVirtualCamera。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

stopVirtualCamera

stopVirtualCamera

关闭图片推流

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

startScreenCapture:

startScreenCapture:

```
- (V2TXLiveCode)startScreenCapture:      (NSString *)appGroup
```

开始全系统的屏幕分享（该接口支持 iOS 11.0 及以上的 iPhone 和 iPad）

参数	描述
appGroup	主 App 与 Broadcast 共享的 Application Group Identifier，可以指定为 nil，但按照文档设置会使功能更加可靠。

> **注意**
> startVirtualCamera，startCamera，startScreenCapture，同一 Pusher 实例下，仅有一个采集源可以上行，不同采集源之间切换，请先关闭前一采集源，再开启后一采集源，保证同一采集源的开启和关闭是成对调用的。比如：采集源从Camera切换到ScreenCapture，调用顺序是 startCamera -> stopCamera -> startScreenCapture。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_ERROR_NOT_SUPPORTED: 功能不支持。

stopScreenCapture

stopScreenCapture

关闭屏幕采集

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

pauseAudio

pauseAudio

静音本地音频

静音本地音频后，SDK 不会继续采集麦克风的声音。
与 stopMicrophone 不同之处在于 pauseAudio 并不会停止发送音频数据，而是继续发送码率极低的静音包。
由于 MP4 等视频文件格式，对于音频的连续性是要求很高的，使用 stopMicrophone 会导致录制出的 MP4 不易播放，
因此在对录制质量要求很高的场景中，建议选择 pauseAudio，从而录制出兼容性更好的 MP4 文件。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

resumeAudio

resumeAudio

取消静音本地音频

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

pauseVideo

pauseVideo

暂停推流器的视频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

resumeVideo

resumeVideo

恢复推流器的视频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startPush:

startPush:

```
- (V2TXLiveCode)startPush:      (NSString *)url
```

开始音视频数据推流

参数	描述
url	推流的目标地址，支持任意推流服务端。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 操作成功，开始连接推流目标地址。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，url 不合法。
- V2TXLIVE_ERROR_INVALID_LICENSE: 操作失败，license 不合法，鉴权失败。
- V2TXLIVE_ERROR_REFUSED: 操作失败，RTC 不支持同一设备上同时推拉同一个 StreamId。

stopPush

stopPush

停止推送音视频数据

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

isPushing

isPushing

当前推流器是否正在推流中

返回值说明：

是否正在推流。
- 1: 正在推流中。
- 0: 已经停止推流。

setAudioQuality:

setAudioQuality:

```
- (V2TXLiveCode)setAudioQuality:(V2TXLiveAudioQuality)quality
```

设置推流音频质量

参数	描述
quality	音频质量 V2TXLiveAudioQuality 。 • V2TXLiveAudioQualityDefault【默认值】: 通用。 • V2TXLiveAudioQualitySpeech: 语音。 • V2TXLiveAudioQualityMusic: 音乐。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 推流过程中，不允许调整音质。

setVideoQuality:

setVideoQuality:

- (V2TXLiveCode)setVideoQuality: (V2TXLiveVideoEncoderParam *)param

设置推流视频编码参数

参数	描述
param	视频编码参数 V2TXLiveVideoEncoderParam 。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

getAudioEffectManager

getAudioEffectManager

获取音效管理对象

通过音效管理，您可以使用以下功能：
- 调整麦克风收集的人声音量。
- 设置混响和变声效果。
- 开启耳返，设置耳返音量。
- 添加背景音乐，调整背景音乐的播放效果。

参考 TXAudioEffectManager

getBeautyManager

getBeautyManager

获取美颜管理对象

通过美颜管理，您可以使用以下功能：

- 设置”美颜风格”、”美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小鼻”、“亮眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。
- 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”。
- 设置人脸挂件（素材）等动态效果。
- 添加美妆。
- 进行手势识别。

参考 TXBeautyManager

getDeviceManager

getDeviceManager

获取设备管理对象

通过设备管理，您可以使用以下功能：

- 切换前后摄像头。
- 设置自动聚焦。
- 设置摄像头缩放倍数。
- 打开或关闭闪光灯。
- 切换耳机或者扬声器。
- 修改音量类型(媒体音量或者通话音量)。

参考 TXDeviceManager

snapshot

snapshot

截取推流过程中的本地画面

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 已经停止推流，不允许调用截图操作。

setWatermark:x:y:scale:

setWatermark:x:y:scale:

```
- (V2TXLiveCode)setWatermark:    (TXImage *)image
                           x:    (float)x
                           y:    (float)y
                       scale:    (float)scale
```

设置推流器水印。默认情况下，水印不开启

参数	描述
image	水印图片。如果该值为 nil，则等效于禁用水印。
scale	水印图片的缩放比例，取值范围为0 - 1的浮点数。
x	水印的横坐标，取值范围为0 - 1的浮点数。
y	水印的纵坐标，取值范围为0 - 1的浮点数。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

enableVolumeEvaluation:

enableVolumeEvaluation:

```
- (V2TXLiveCode)enableVolumeEvaluation:(NSUInteger)intervalMs
```

启用采集音量大小提示

开启后可以在 onMicrophoneVolumeUpdate 回调中获取到 SDK 对音量大小值的评估。

参数	描述
intervalMs	决定了音量大小回调的触发间隔，单位为 ms，最小间隔为 100ms，如果小于等于0则会关闭回调，建议设置为 300ms。【默认值】：0，不开启。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

enableCustomVideoProcess:pixelFormat:bufferType:

enableCustomVideoProcess:pixelFormat:bufferType:

```
- (V2TXLiveCode)enableCustomVideoProcess:(BOOL)enable
                             pixelFormat:(V2TXLivePixelFormat)pixelFormat
                              bufferType:(V2TXLiveBufferType)bufferType
```

开启/关闭自定义视频处理

参数	描述
bufferType	指定回调的数据格式。
enable	YES: 开启; NO: 关闭。【默认值】：NO。
pixelFormat	指定回调的像素格式。

注意

支持的格式组合：
V2TXLivePixelFormatTexture2D+V2TXLiveBufferTypeTexture
V2TXLivePixelFormatNV12+V2TXLiveBufferTypePixelBuffer
V2TXLivePixelFormatBGRA32+V2TXLiveBufferTypePixelBuffer

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_NOT_SUPPORTED: 不支持的格式。

enableCustomVideoCapture:

enableCustomVideoCapture:

```
- (V2TXLiveCode)enableCustomVideoCapture:      (BOOL)enable
```

开启/关闭自定义视频采集

在自定义视频采集模式下，SDK 不再从摄像头采集图像，只保留编码和发送能力。

参数	描述
enable	YES：开启自定义采集；NO：关闭自定义采集。【默认值】：NO。

注意
需要在 startPush 之前调用，才会生效。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

enableCustomAudioCapture:

enableCustomAudioCapture:

```
- (V2TXLiveCode)enableCustomAudioCapture:      (BOOL)enable
```

开启/关闭自定义音频采集

@brief 开启/关闭自定义音频采集。
在自定义音频采集模式下，SDK 不再从麦克风采集声音，只保留编码和发送能力。

参数	描述
enable	YES: 开启自定义采集; NO: 关闭自定义采集。【默认值】: NO。

> **注意**
> 需要在 startPush 前调用才会生效。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

sendCustomVideoFrame:

sendCustomVideoFrame:

```
- (V2TXLiveCode)sendCustomVideoFrame:(V2TXLiveVideoFrame *)videoFrame
```

在自定义视频采集模式下，将采集的视频数据发送到SDK

在自定义视频采集模式下，SDK不再采集摄像头数据，仅保留编码和发送功能。
您可以把采集到的 SampleBuffer 打包到 V2TXLiveVideoFrame 中，然后通过该API定期的发送。

参数	描述
videoFrame	向 SDK 发送的 视频帧数据 V2TXLiveVideoFrame 。

> **注意**
> 需要在 startPush 之前调用 enableCustomVideoCapture 开启自定义采集。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 发送失败，视频帧数据不合法。

sendCustomAudioFrame:

sendCustomAudioFrame:

```
- (V2TXLiveCode)sendCustomAudioFrame:    (V2TXLiveAudioFrame *)audioFrame
```

在自定义音频采集模式下，将采集的音频数据发送到SDK

@info 在自定义音频采集模式下，将采集的音频数据发送到SDK，SDK不再采集麦克风数据，仅保留编码和发送功能。

@param audioFrame 向 SDK 发送的 音频帧数据 V2TXLiveAudioFrame 。
@return 返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 发送失败，音频帧数据不合法。

@note 需要在 startPush 之前调用 enableCustomAudioCapture 开启自定义采集。

enableAudioProcessObserver:format:

enableAudioProcessObserver:format:

```
- (V2TXLiveCode)enableAudioProcessObserver:    (BOOL)enable
                                    format:    (V2TXLiveAudioFrameObserverFo
```

开启/关闭对经过前处理后的本地音频帧的监听回调

参数	描述
enable	是否开启。【默认值】：false。
form	设置回调出的 AudioFrame 的格式。

at	

注意
需要在 startPush 之前调用，才会生效。

sendSeiMessage:data:

sendSeiMessage:data:

```
- (V2TXLiveCode)sendSeiMessage:(int)payloadType
                          data:(NSData *)data
```

发送 SEI 消息

播放端 V2TXLivePlayer 通过 V2TXLivePlayerObserver 中的 `onReceiveSeiMessage` 回调来接收该消息。

参数	描述
data	待发送的数据。
payloadType	数据类型，支持 5、242。推荐填：242。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

showDebugView:

showDebugView:

```
- (void)showDebugView:(BOOL)isShow
```

显示仪表盘

参数	描述
isShow	是否显示。【默认值】：NO。

setProperty:value:

setProperty:value:

```
- (V2TXLiveCode)setProperty:    (NSString *)key
                      value:    (NSObject *)value
```

调用 V2TXLivePusher 的高级 API 接口。

参数	描述
key	高级 API 对应的 key，详情请参考 V2TXLiveProperty 定义。
value	调用 key 所对应的高级 API 时，需要的参数。

> **注意**
> 该接口用于调用一些高级功能。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，key 不允许为空。

setMixTranscodingConfig:

setMixTranscodingConfig:

```
- (V2TXLiveCode)setMixTranscodingConfig:    (V2TXLiveTranscodingConfig *)config
```

设置云端的混流转码参数

如果您在实时音视频 控制台 中的功能配置页开启了“启用旁路推流”功能，
房间里的每一路画面都会有一个默认的直播 CDN 地址 。

一个直播间中可能有不止一位主播，而且每个主播都有自己的画面和声音，但对于 CDN 观众来说，他们只需要一路直播流，
所以您需要将多路音视频流混成一路标准的直播流，这就需要混流转码

当您调用 setMixTranscodingConfig 接口时，SDK 会向腾讯云的转码服务器发送一条指令，目的是将房间里的多路音视频流混合为一路，
您可以通过 mixUsers 参数来调整每一路画面的位置，以及是否只混合声音，也可以通过 videoWidth、videoHeight、videoBitrate 等参数控制混合音视频流的编码参数。

【画面1】=> 解码 ====> \
\
【画面2】=> 解码 => 画面混合 => 编码 => 【混合后的画面】
/
【画面3】=> 解码 ====> /

【声音1】=> 解码 ====> \
\
【声音2】=> 解码 => 声音混合 => 编码 => 【混合后的声音】
/
【声音3】=> 解码 ====> /

参考文档： 云端混流转码 。

参数	描述
config	请参考 V2TXLiveDef.h 中关于 V2TXLiveTranscodingConfig 的介绍。如果传入 nil 则取消云端混流转码。

注意
关于云端混流的注意事项：

- 云端转码会引入一定的 CDN 观看延时，大概会增加1 – 2秒。
- 调用该函数的用户，会将连麦中的多路画面混合到自己当前这路画面或者 config 中指定的 streamId 上。
- 请注意，若您还在房间中且不再需要混流，请务必传入 nil 进行取消，因为当您发起混流后，云端混流模块就会开始工作，不及时取消混流可能会引起不必要的计费损失。
- 请放心，您退房时会自动取消混流状态。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 未开启推流时，不允许设置混流转码参数。

startLocalRecording:

startLocalRecording:

```
- (V2TXLiveCode)startLocalRecording:(V2TXLiveLocalRecordingParams *)para
```

开始录制音视频流

> **注意**
> 推流开启后才能开始录制，非推流状态下开启录制无效。
> - 录制过程中不要动态切换分辨率和软/硬剪辑，生成的视频极有可能出现异常。

返回值说明：

返回值 V2TXLiveCode 。

- `V2TXLIVE_OK` ：成功。
- `V2TXLIVE_ERROR_INVALID_PARAMETER` ：参数不合法，比如filePath 为空。
- `V2TXLIVE_ERROR_REFUSED` ：API被拒绝，推流尚未开始。

stopLocalRecording

stopLocalRecording

停止录制音视频流

注意

当停止推流后，如果视频还在录制中，SDK 内部会自动结束录制。

V2TXLivePusherObserver

最近更新时间：2024-01-16 14:35:21

Module: V2TXLivePusherObserver @ TXLiteAVSDK

Function: 腾讯云直播推流的回调通知

功能

腾讯云直播的推流回调通知。

介绍

可以接收 V2TXLivePusher 推流器的一些推流通知，包括推流器连接状态、音视频首帧回调、统计数据、警告和错误信息等。

V2TXLivePusherObserver

V2TXLivePusherObserver

函数列表	描述
onError:message:extraInfo:	直播推流器错误通知，推流器出现错误时，会回调该通知
onWarning:message:extraInfo:	直播推流器警告通知
onCaptureFirstAudioFrame	首帧音频采集完成的回调通知
onCaptureFirstVideoFrame	首帧视频采集完成的回调通知
onMicrophoneVolumeUpdate:	麦克风采集音量值回调
onPushStatusUpdate:message:extraInfo:	推流器连接状态回调通知
onStatisticsUpdate:	直播推流器统计数据回调
onSnapshotComplete:	截图回调

onProcessAudioFrame:	本地采集并经过音频模块前处理、音效处理和混 BGM 后的音频数据回调
onProcessVideoFrame:dstFrame:	自定义视频处理回调
onGLContextDestroyed	SDK 内部的 OpenGL 环境的销毁通知
onSetMixTranscodingConfig:message:	设置云端的混流转码参数的回调，对应于 setMixTranscodingConfig 接口
onScreenCaptureStarted	当屏幕分享开始时，SDK 会通过此回调通知
onScreenCaptureStopped:	当屏幕分享停止时，SDK 会通过此回调通知
onLocalRecordBegin:storagePath:	录制任务开始的事件回调
onLocalRecording:storagePath:	录制任务正在进行中的进展事件回调
onLocalRecordComplete:storagePath:	录制任务已经结束的事件回调

onError:message:extraInfo:

onError:message:extraInfo:

```
- (void)onError:    (V2TXLiveCode)code
        message:    (NSString *)msg
      extraInfo:    (NSDictionary *)extraInfo
```

直播推流器错误通知，推流器出现错误时，会回调该通知

参数	描述
code	错误码 V2TXLiveCode 。
extraInfo	扩展信息。
msg	错误信息。

onWarning:message:extraInfo:

onWarning:message:extraInfo:

```
- (void)onWarning:    (V2TXLiveCode)code
          message:    (NSString *)msg
        extraInfo:    (NSDictionary *)extraInfo
```

直播推流器警告通知

参数	描述
code	警告码 V2TXLiveCode 。
extraInfo	扩展信息。
msg	警告信息。

onCaptureFirstAudioFrame

onCaptureFirstAudioFrame

首帧音频采集完成的回调通知

onCaptureFirstVideoFrame

onCaptureFirstVideoFrame

首帧视频采集完成的回调通知

onMicrophoneVolumeUpdate:

onMicrophoneVolumeUpdate:

```
- (void)onMicrophoneVolumeUpdate:    (NSInteger)volume
```

麦克风采集音量值回调

参数	描述
volume	音量大小。

注意
调用 enableVolumeEvaluation 开启采集音量大小提示之后，会收到这个回调通知。

onPushStatusUpdate:message:extraInfo:

onPushStatusUpdate:message:extraInfo:

```
- (void)onPushStatusUpdate:    (V2TXLivePushStatus)status
                   message:    (NSString *)msg
                 extraInfo:    (NSDictionary *)extraInfo
```

推流器连接状态回调通知

参数	描述
extraInfo	扩展信息。
msg	连接状态信息。
status	推流器连接状态 V2TXLivePushStatus 。

onStatisticsUpdate:

onStatisticsUpdate:

```
- (void)onStatisticsUpdate:    (V2TXLivePusherStatistics *)statistics
```

直播推流器统计数据回调

参数	描述
statistics	推流器统计数据 V2TXLivePusherStatistics

onSnapshotComplete:

onSnapshotComplete:

```
- (void)onSnapshotComplete:(nullable TXImage *)image
```

截图回调

参数	描述
image	已截取的视频画面。

> **注意**
> 调用 snapshot 截图之后，会收到这个回调通知。

onProcessAudioFrame:

onProcessAudioFrame:

```
- (void) onProcessAudioFrame:(V2TXLiveAudioFrame *)frame
```

本地采集并经过音频模块前处理、音效处理和混 BGM 后的音频数据回调

当您设置完音频数据自定义回调之后，SDK 内部会把刚采集到并经过前处理、音效处理和混 BGM 之后的数据，在最终进行网络编码之前，以 PCM 格式的形式通过本接口回调给您。

- 此接口回调出的音频时间帧长固定为 0.02s，格式为 PCM 格式。
- 由时间帧长转化为字节帧长的公式为 `采样率 × 时间帧长 × 声道数 × 采样点位宽` 。
- 以 SDK 默认的音频录制格式 48000 采样率、单声道、16采样点位宽为例，字节帧长为 `48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节` 。

参数	描述

frame	PCM 格式的音频数据帧。

注意

1. 请不要在此回调函数中做任何耗时操作，由于 SDK 每隔 20ms 就要处理一帧音频数据，如果您的处理时间超过 20ms，就会导致声音异常。
2. 此接口回调出的音频数据是可读写的，也就是说您可以在回调函数中同步修改音频数据，但请保证处理耗时。

onProcessVideoFrame:dstFrame:

onProcessVideoFrame:dstFrame:

```
- (void)onProcessVideoFrame:(V2TXLiveVideoFrame * _Nonnull)srcFrame
                   dstFrame:(V2TXLiveVideoFrame * _Nonnull)dstFrame
```

自定义视频处理回调

参数	描述
dstFrame	用于承载处理过的视频画面。
srcFrame	用于承载未处理的视频画面。

注意

需要调用 enableCustomVideoProcess 开启自定义视频处理，才会收到这个回调通知。

【情况一】美颜组件会产生新的纹理

如果您使用的美颜组件会在处理图像的过程中产生一帧全新的纹理（用于承载处理后的图像），那请您在回调函数中将 dstFrame.textureId 设置为新纹理的 ID。

```ObjectiveC
- (void) onProcessVideoFrame:(V2TXLiveVideoFrame * _Nonnull)srcFrame
dstFrame:(V2TXLiveVideoFrame * _Nonnull)dstFrame
{
GLuint dstTextureId = renderItemWithTexture(srcFrame.textureId,
srcFrame.width, srcFrame.height);
dstFrame.textureId = dstTextureId;
```

```
return 0;
}
```
【情况二】美颜组件并不自身产生新纹理
如果您使用的第三方美颜模块并不生成新的纹理，而是需要您设置给该模块一个输入纹理和一个输出纹理，则可以考虑如下方案：
```ObjectiveC
- (void) onProcessVideoFrame:(V2TXLiveVideoFrame * _Nonnull)srcFrame dstFrame:(V2TXLiveVideoFrame * _Nonnull)dstFrame
{
thirdparty_process(srcFrame.textureId, srcFrame.width, srcFrame.height, dstFrame.textureId);
return 0;
}
```

onGLContextDestroyed

onGLContextDestroyed

SDK 内部的 OpenGL 环境的销毁通知

onSetMixTranscodingConfig:message:

onSetMixTranscodingConfig:message:

```
- (void)onSetMixTranscodingConfig:  (V2TXLiveCode)code
                           message:  (NSString *)msg
```

设置云端的混流转码参数的回调，对应于 {@link setMixTranscodingConfig} 接口

参数	描述
code	0表示成功，其余值表示失败。

msg	具体错误原因。

onScreenCaptureStarted

onScreenCaptureStarted

当屏幕分享开始时，SDK 会通过此回调通知

onScreenCaptureStopped:

onScreenCaptureStopped:

```
- (void)onScreenCaptureStopped:    (int)reason
```

当屏幕分享停止时，SDK 会通过此回调通知

参数	描述
reason	停止原因 • 0：表示用户主动停止。 • 1：iOS 表示录屏被系统中断；Mac、Windows 表示屏幕分享窗口被关闭。 • 2：Windows 表示屏幕分享的显示屏状态变更（如接口被拔出、投影模式变更等）；其他平台不抛出。

onLocalRecordBegin:storagePath:

onLocalRecordBegin:storagePath:

```
- (void)onLocalRecordBegin:    (NSInteger)errCode
                storagePath:    (NSString *)storagePath
```

录制任务开始的事件回调

参数	描述

code	状态码。 • 0：录制任务启动成功。 • -1：内部错误导致录制任务启动失败。 • -2：文件后缀名有误（比如不支持的录制格式）。 • -6：录制已经启动，需要先停止录制。 • -7：录制文件已存在，需要先删除文件。 • -8：录制目录无写入权限，请检查目录权限问题。
storagePath	录制的文件地址。

onLocalRecording:storagePath:

onLocalRecording:storagePath:

```
- (void)onLocalRecording:    (NSInteger)durationMs
            storagePath:     (NSString *)storagePath
```

录制任务正在进行中的进展事件回调

参数	描述
durationMs	录制时长。
storagePath	录制的文件地址。

onLocalRecordComplete:storagePath:

onLocalRecordComplete:storagePath:

```
- (void)onLocalRecordComplete:    (NSInteger)errCode
                   storagePath:   (NSString *)storagePath
```

录制任务已经结束的事件回调

参数	描述
code	状态码。 • 0：结束录制任务成功。 • -1：录制失败。 • -2：切换分辨率或横竖屏导致录制结束。 • -3：录制时间太短，或未采集到任何视频或音频数据，请检查录制时长，或是否已开启音、视频采集。
storagePath	录制的文件地址。

V2TXLivePlayer

最近更新时间：2024-05-08 17:33:43

Module: V2TXLivePlayer @ TXLiteAVSDK

Function: 腾讯云直播播放器

功能

腾讯云直播播放器。
主要负责从指定的直播流地址拉取音视频数据，并进行解码和本地渲染播放。

介绍

播放器包含如下能力：

- 支持 RTMP、HTTP-FLV、HLS、TRTC、WebRTC 协议。
- 屏幕截图，可以截取当前直播流的视频画面。
- 延时调节，可以设置播放器缓存自动调整的最小和最大时间。
- 自定义的视频数据处理，您可以根据项目需要处理直播流中的视频数据后，再进行渲染以及播放。

V2TXLivePlayer

V2TXLivePlayer

函数列表	描述
setObserver:	设置播放器回调
setRenderView:	设置播放器的视频渲染 View，该控件负责显示视频内容
setRenderRotation:	设置播放器画面的旋转角度
setRenderFillMode:	设置画面的填充模式
startLivePlay:	开始播放音视频流

stopPlay	停止播放音视频流
isPlaying	播放器是否正在播放中
pauseAudio	暂停播放器的音频流
resumeAudio	恢复播放器的音频流
pauseVideo	暂停播放器的视频流
resumeVideo	恢复播放器的视频流
setPlayoutVolume:	设置播放器音量
setCacheParams:maxTime:	设置播放器缓存自动调整的最小和最大时间（单位：秒）
switchStream:	直播流无缝切换，支持 FLV 和 LEB
getStreamList	获取码流信息
enableVolumeEvaluation:	启用播放音量大小提示
snapshot	截取播放过程中的视频画面
enableObserveVideoFrame:pixelFormat:bufferType:	开启/关闭对视频帧的监听回调
enableObserveAudioFrame:	开启/关闭对音频数据的监听回调
enableReceiveSeiMessage:payloadType:	开启接收 SEI 消息
enablePictureInPicture:	开启画中画功能，仅支持直播和快直播播放
showDebugView:	是否显示播放器状态信息的调试浮层
setProperty:value:	调用 V2TXLivePlayer 的高级 API 接口
startLocalRecording:	开始录制音视频流
stopLocalRecording	停止录制音视频流

setObserver:

setObserver:

```
- (void)setObserver:(id<V2TXLivePlayerObserver>)observer
```

设置播放器回调

通过设置回调，可以监听 V2TXLivePlayer 播放器的一些回调事件，
包括播放器状态、播放音量回调、音视频首帧回调、统计数据、警告和错误信息等。

参数	描述
observer	播放器的回调目标对象，更多信息请查看 V2TXLivePlayerObserver

setRenderView:

setRenderView:

```
- (V2TXLiveCode)setRenderView:(TXView *)view
```

设置播放器的视频渲染 View，该控件负责显示视频内容

参数	描述
view	播放器渲染 View。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK：成功。

setRenderRotation:

setRenderRotation:

```
- (V2TXLiveCode)setRenderRotation:(V2TXLiveRotation)rotation
```

设置播放器画面的旋转角度

参数	描述

rotation	旋转角度 V2TXLiveRotation 。 • V2TXLiveRotation0【默认值】: 0度, 不旋转。 • V2TXLiveRotation90: 顺时针旋转90度。 • V2TXLiveRotation180: 顺时针旋转180度。 • V2TXLiveRotation270: 顺时针旋转270度。

返回值说明:

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setRenderFillMode:

setRenderFillMode:

```
- (V2TXLiveCode)setRenderFillMode:(V2TXLiveFillMode)mode
```

设置画面的填充模式

参数	描述
mode	画面填充模式 V2TXLiveFillMode 。 • V2TXLiveFillModeFill【默认值】: 图像铺满屏幕，不留黑边，如果图像宽高比不同于屏幕宽高比，部分画面内容会被裁剪掉。 • V2TXLiveFillModeFit: 图像适应屏幕，保持画面完整，但如果图像宽高比不同于屏幕宽高比，会有黑边的存在。 • V2TXLiveFillModeScaleFill: 图像拉伸铺满，因此长度和宽度可能不会按比例变化。

返回值说明:

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startLivePlay:

startLivePlay:

```
- (V2TXLiveCode)startLivePlay:(NSString *)url
```

开始播放音视频流

参数	描述
url	音视频流的播放地址，支持 RTMP，HTTP-FLV，TRTC。

> **注意**
> 10.7 版本开始，需要通过 setLicence 或者 setLicence 设置 Licence 后方可成功播放，否则将播放失败（黑屏），全局仅设置一次即可。直播 Licence、短视频 Licence 和视频播放 Licence 均可使用，若您暂未获取上述 Licence，可 快速免费申请测试版 Licence 以正常播放，正式版 Licence 需 购买 。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 操作成功，开始连接并播放。
- V2TXLIVE_ERROR_INVALID_PARAMETER：操作失败，url 不合法。
- V2TXLIVE_ERROR_REFUSED：RTC 不支持同一设备上同时推拉同一个 StreamId。
- V2TXLIVE_ERROR_INVALID_LICENSE：licence 不合法，播放失败。

stopPlay

stopPlay

停止播放音视频流

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

isPlaying

isPlaying

播放器是否正在播放中

返回值说明：

是否正在播放。

- 1: 正在播放中。
- 0: 已经停止播放。

pauseAudio

pauseAudio

暂停播放器的音频流

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

resumeAudio

resumeAudio

恢复播放器的音频流

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

pauseVideo

pauseVideo

暂停播放器的视频流

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

resumeVideo

resumeVideo

恢复播放器的视频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setPlayoutVolume:

setPlayoutVolume:

```
- (V2TXLiveCode)setPlayoutVolume:(NSUInteger)volume
```

设置播放器音量

参数	描述
volume	音量大小，取值范围0 - 100。【默认值】：100。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setCacheParams:maxTime:

setCacheParams:maxTime:

```
- (V2TXLiveCode)setCacheParams:(CGFloat)minTime
                       maxTime:(CGFloat)maxTime
```

设置播放器缓存自动调整的最小和最大时间（单位：秒）

参数	描述
maxTime	缓存自动调整的最大时间，取值需要大于0。【默认值】：5。
minTime	缓存自动调整的最小时间，取值需要大于0。【默认值】：1。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，minTime 和 maxTime 需要大于0。
- V2TXLIVE_ERROR_REFUSED: 播放器处于播放状态，不支持修改缓存策略。

switchStream:

switchStream:

```
- (V2TXLiveCode)switchStream:      (NSString*)newUrl
```

直播流无缝切换，支持 FLV 和 LEB

参数	描述
newUrl	新的拉流地址。

getStreamList

getStreamList

获取码流信息

enableVolumeEvaluation:

enableVolumeEvaluation:

```
- (V2TXLiveCode)enableVolumeEvaluation:      (NSUInteger)intervalMs
```

启用播放音量大小提示

开启后可以在 onPlayoutVolumeUpdate 回调中获取到 SDK 对音量大小值的评估。

参数	描述
intervalMs	决定了 onPlayoutVolumeUpdate 回调的触发间隔，单位为ms，最小间隔为100ms，如果小于等于0则会关闭回调，建议设置为300ms；【默认值】：0，不开启。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

snapshot

snapshot

截取播放过程中的视频画面

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 播放器处于停止状态，不允许调用截图操作。

enableObserveVideoFrame:pixelFormat:bufferType:

enableObserveVideoFrame:pixelFormat:bufferType:

```
- (V2TXLiveCode)enableObserveVideoFrame:      (BOOL)enable
                            pixelFormat:      (V2TXLivePixelFormat)pixelFormat
                             bufferType:      (V2TXLiveBufferType)bufferType
```

开启/关闭对视频帧的监听回调

SDK 在您开启此开关后将不再渲染视频画面，您可以通过 V2TXLivePlayerObserver 获得视频帧，并执行自定义的渲染逻辑。

参数	描述
bufferType	自定义渲染回调的视频数据格式 V2TXLiveBufferType 。
enable	是否开启自定义渲染。【默认值】：NO。
pixelFormat	自定义渲染回调的视频像素格式 V2TXLivePixelFormat 。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_NOT_SUPPORTED: 像素格式或者数据格式不支持。

enableObserveAudioFrame:

enableObserveAudioFrame:

```
- (V2TXLiveCode)enableObserveAudioFrame:      (BOOL)enable
```

开启/关闭对音频数据的监听回调

如果您开启此开关，您可以通过 V2TXLivePlayerObserver 获得音频数据，并执行自定义的逻辑。

参数	描述
enable	是否开启音频数据回调。【默认值】：NO。

返回值说明：

返回值 V2TXLiveCode

- V2TXLIVE_OK: 成功

enableReceiveSeiMessage:payloadType:

enableReceiveSeiMessage:payloadType:

```
- (V2TXLiveCode)enableReceiveSeiMessage:      (BOOL)enable
                                payloadType:      (int)payloadType
```

开启接收 SEI 消息

参数	描述
enable	YES: 开启接收 SEI 消息; NO: 关闭接收 SEI 消息。【默认值】: NO。
payloadType	指定接收 SEI 消息的 payloadType，支持 5、242，请与发送端的 payloadType 保持一致。

返回值说明:

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

enablePictureInPicture:

enablePictureInPicture:

```
- (V2TXLiveCode)enablePictureInPicture:      (BOOL)enable
```

开启画中画功能，仅支持直播和快直播播放

参数	描述
enable	YES: 开启画中画功能; NO: 关闭画中画功能。【默认值】: NO。

返回值说明:

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

showDebugView:

showDebugView:

```
- (void)showDebugView:      (BOOL)isShow
```

是否显示播放器状态信息的调试浮层

参数	描述
isShow	是否显示。【默认值】：NO。

setProperty:value:

setProperty:value:

```
- (V2TXLiveCode)setProperty:    (NSString *)key
                      value:    (NSObject *)value
```

调用 V2TXLivePlayer 的高级 API 接口

参数	描述
key	高级 API 对应的 key, 详情请参考 V2TXLiveProperty 定义。
value	调用 key 所对应的高级 API 时，需要的参数。

> **注意**
> 该接口用于调用一些高级功能。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，key 不允许为 nil。

startLocalRecording:

startLocalRecording:

```
- (V2TXLiveCode)startLocalRecording:    (V2TXLiveLocalRecordingParams *)para
```

开始录制音视频流

> **注意**
> 拉流开启后才能开始录制，非拉流状态下开启录制无效。
> - 录制过程中不要动态切换软/硬解，生成的视频极有可能出现异常。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK ：成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER ：参数不合法，例如 filePath 为空。
- V2TXLIVE_ERROR_REFUSED ：API被拒绝，拉流尚未开始。

stopLocalRecording

stopLocalRecording

停止录制音视频流

> **注意**
> 当停止拉流后，如果视频还在录制中，SDK 内部会自动结束录制。

V2TXLivePlayerObserver

最近更新时间：2024-01-16 14:35:21

Module: V2TXLivePlayerObserver @ TXLiteAVSDK

Function: 腾讯云直播的播放器回调通知

功能

腾讯云直播的播放器回调通知。

介绍

可以接收 V2TXLivePlayer 播放器的一些回调通知，包括播放器状态、播放音量回调、音视频首帧回调、统计数据、警告和错误信息等。

V2TXLivePlayerObserver

V2TXLivePlayerObserver

函数列表	描述
onError:code:message:extraInfo:	直播播放器错误通知，播放器出现错误时，会回调该通知
onWarning:code:message:extraInfo:	直播播放器警告通知
onVideoResolutionChanged:width:height:	直播播放器分辨率变化通知
onConnected:extraInfo:	已经成功连接到服务器
onVideoPlaying:firstPlay:extraInfo:	视频播放事件
onAudioPlaying:firstPlay:extraInfo:	音频播放事件
onVideoLoading:extraInfo:	视频加载事件
onAudioLoading:extraInfo:	音频加载事件

onPlayoutVolumeUpdate:volume:	播放器音量大小回调
onStatisticsUpdate:statistics:	直播播放器统计数据回调
onSnapshotComplete:image:	截图回调
onRenderVideoFrame:frame:	自定义视频渲染回调
onPlayoutAudioFrame:frame:	音频数据回调
onReceiveSeiMessage:payloadType:data:	收到 SEI 消息的回调，发送端通过 V2TXLivePusher 中的 sendSeiMessage 来发送 SEI 消息
onStreamSwitched:url:code:	分辨率无缝切换回调
onPictureInPictureStateUpdate:state:message:extraInfo:	画中画状态变更回调
onLocalRecordBegin:errCode:storagePath:	录制任务开始的事件回调
onLocalRecording:durationMs:storagePath:	录制任务正在进行中的进展事件回调
onLocalRecordComplete:errCode:storagePath:	录制任务已经结束的事件回调

onError:code:message:extraInfo:

onError:code:message:extraInfo:

```
- (void)onError:    (id<V2TXLivePlayer>)player
          code:     (V2TXLiveCode)code
       message:     (NSString *)msg
     extraInfo:     (NSDictionary *)extraInfo
```

直播播放器错误通知，播放器出现错误时，会回调该通知

参数	描述

code	错误码 V2TXLiveCode 。
extraInfo	扩展信息。
msg	错误信息。
player	回调该通知的播放器对象。

onWarning:code:message:extraInfo:

onWarning:code:message:extraInfo:

```
- (void)onWarning:    (id<V2TXLivePlayer>)player
             code:    (V2TXLiveCode)code
          message:    (NSString *)msg
        extraInfo:    (NSDictionary *)extraInfo
```

直播播放器警告通知

参数	描述
code	警告码 V2TXLiveCode 。
extraInfo	扩展信息。
msg	警告信息。
player	回调该通知的播放器对象。

onVideoResolutionChanged:width:height:

onVideoResolutionChanged:width:height:

```
- (void)onVideoResolutionChanged:    (id<V2TXLivePlayer>)player
                           width:    (NSInteger)width
                          height:    (NSInteger)height
```

直播播放器分辨率变化通知

参数	描述
height	视频高。
player	回调该通知的播放器对象。
width	视频宽。

onConnected:extraInfo:

onConnected:extraInfo:

```
- (void)onConnected:    (id<V2TXLivePlayer>)player
         extraInfo:    (NSDictionary *)extraInfo
```

已经成功连接到服务器

参数	描述
extraInfo	扩展信息。
player	回调该通知的播放器对象。

onVideoPlaying:firstPlay:extraInfo:

onVideoPlaying:firstPlay:extraInfo:

```
- (void)onVideoPlaying:    (id<V2TXLivePlayer>)player
          firstPlay:    (BOOL)firstPlay
         extraInfo:    (NSDictionary *)extraInfo
```

视频播放事件

参数	描述
extraInfo	扩展信息。
firstPlay	第一次播放标志。
player	回调该通知的播放器对象。

onAudioPlaying:firstPlay:extraInfo:

onAudioPlaying:firstPlay:extraInfo:

```
- (void)onAudioPlaying:    (id<V2TXLivePlayer>)player
             firstPlay:    (BOOL)firstPlay
             extraInfo:    (NSDictionary *)extraInfo
```

音频播放事件

参数	描述
extraInfo	扩展信息。
firstPlay	第一次播放标志。
player	回调该通知的播放器对象。

onVideoLoading:extraInfo:

onVideoLoading:extraInfo:

```
- (void)onVideoLoading:    (id<V2TXLivePlayer>)player
             extraInfo:    (NSDictionary *)extraInfo
```

视频加载事件

参数	描述

extraInfo	扩展信息。
player	回调该通知的播放器对象。

onAudioLoading:extraInfo:

onAudioLoading:extraInfo:

```
- (void)onAudioLoading:    (id<V2TXLivePlayer>)player
             extraInfo:    (NSDictionary *)extraInfo
```

音频加载事件

参数	描述
extraInfo	扩展信息。
player	回调该通知的播放器对象。

onPlayoutVolumeUpdate:volume:

onPlayoutVolumeUpdate:volume:

```
- (void)onPlayoutVolumeUpdate:    (id<V2TXLivePlayer>)player
                       volume:    (NSInteger)volume
```

播放器音量大小回调

参数	描述
player	回调该通知的播放器对象。
volume	音量大小。

注意

调用 enableVolumeEvaluation 开启播放音量大小提示之后，会收到这个回调通知。

onStatisticsUpdate:statistics:

onStatisticsUpdate:statistics:

```
- (void)onStatisticsUpdate:    (id<V2TXLivePlayer>)player
                statistics:    (V2TXLivePlayerStatistics *)statistics
```

直播播放器统计数据回调

参数	描述
player	回调该通知的播放器对象。
statistics	播放器统计数据 V2TXLivePlayerStatistics 。

onSnapshotComplete:image:

onSnapshotComplete:image:

```
- (void)onSnapshotComplete:    (id<V2TXLivePlayer>)player
                     image:    (nullable TXImage *)image
```

截图回调

参数	描述
image	已截取的视频画面。
player	回调该通知的播放器对象。

注意

调用 snapshot 截图之后，会收到这个回调通知。

onRenderVideoFrame:frame:

onRenderVideoFrame:frame:

```
- (void)onRenderVideoFrame:    (id<V2TXLivePlayer>)player
                     frame:    (V2TXLiveVideoFrame *)videoFrame
```

自定义视频渲染回调

参数	描述
player	回调该通知的播放器对象。
videoFrame	视频帧数据 V2TXLiveVideoFrame 。

注意
需要您调用 enableObserveVideoFrame 开启回调开关。

onPlayoutAudioFrame:frame:

onPlayoutAudioFrame:frame:

```
- (void)onPlayoutAudioFrame:    (id<V2TXLivePlayer>)player
                      frame:    (V2TXLiveAudioFrame *)audioFrame
```

音频数据回调

参数	描述
audioFrame	音频帧数据 V2TXLiveAudioFrame 。
player	回调该通知的播放器对象。

注意

需要您调用 enableObserveAudioFrame 开启回调开关。请在当前回调中使用 audioFrame 的 data。

onReceiveSeiMessage:payloadType:data:

onReceiveSeiMessage:payloadType:data:

```
- (void)onReceiveSeiMessage:    (id<V2TXLivePlayer>)player
                payloadType:    (int)payloadType
                       data:    (NSData *)data
```

收到 SEI 消息的回调，发送端通过 {@link V2TXLivePusher} 中的 `sendSeiMessage` 来发送 SEI 消息

参数	描述
data	数据。
payloadType	回调数据的SEI payloadType。
player	回调该通知的播放器对象。

注意

调用 V2TXLivePlayer 中的 `enableReceiveSeiMessage` 开启接收 SEI 消息之后，会收到这个回调通知。

onStreamSwitched:url:code:

onStreamSwitched:url:code:

```
- (void)onStreamSwitched:    (id<V2TXLivePlayer>)player
                      url:    (NSString *)url
                     code:    (NSInteger)code
```

分辨率无缝切换回调

参数	描述
code	状态码，0：成功，-1：切换超时，-2：切换失败，服务端错误，-3：切换失败，客户端错误。
player	回调该通知的播放器对象。
url	切换的播放地址。

> **注意**
> 调用 V2TXLivePlayer 中的 `switchStream` 切换分辨率，会收到这个回调通知。

onPictureInPictureStateUpdate:state:message:extraInfo:

onPictureInPictureStateUpdate:state:message:extraInfo:

```
- (void)onPictureInPictureStateUpdate:(id<V2TXLivePlayer>)player
                                state:(V2TXLivePictureInPictureState)state
                              message:(NSString *)msg
                            extraInfo:(NSDictionary *)extraInfo
```

画中画状态变更回调

参数	描述
extraInfo	扩展信息。
player	回调该通知的播放器对象。
state	画中画的状态。

> **注意**
> 调用 V2TXLivePlayer 中的 `enablePictureInPicture` 开启画中画之后，会收到这个回调通知。

onLocalRecordBegin:errCode:storagePath:

onLocalRecordBegin:errCode:storagePath:

```
- (void)onLocalRecordBegin:    (id<V2TXLivePlayer>)player
                   errCode:    (NSInteger)errCode
                storagePath:   (NSString *)storagePath
```

录制任务开始的事件回调

参数	描述
code	状态码。 • 0：录制任务启动成功。 • -1：内部错误导致录制任务启动失败。 • -2：文件后缀名有误（比如不支持的录制格式）。 • -6：录制已经启动，需要先停止录制。 • -7：录制文件已存在，需要先删除文件。 • -8：录制目录无写入权限，请检查目录权限问题。
player	回调该通知的播放器对象。
storagePath	录制的文件地址。

onLocalRecording:durationMs:storagePath:

onLocalRecording:durationMs:storagePath:

```
- (void)onLocalRecording:    (id<V2TXLivePlayer>)player
                 durationMs: (NSInteger)durationMs
                storagePath: (NSString *)storagePath
```

录制任务正在进行中的进展事件回调

参数	描述

durationMs	录制时长。
player	回调该通知的播放器对象。
storagePath	录制的文件地址。

onLocalRecordComplete:errCode:storagePath:

onLocalRecordComplete:errCode:storagePath:

```
- (void)onLocalRecordComplete:    (id<V2TXLivePlayer>)player
                       errCode:   (NSInteger)errCode
                   storagePath:   (NSString *)storagePath
```

录制任务已经结束的事件回调

参数	描述
code	状态码。 • 0：结束录制任务成功。 • -1：录制失败。 • -2：切换分辨率或横竖屏导致录制结束。 • -3：录制时间太短，或未采集到任何视频或音频数据，请检查录制时长，或是否已开启音、视频采集。
player	回调该通知的播放器对象。
storagePath	录制的文件地址。

V2TXLivePremier

最近更新时间：2024-03-07 15:14:11

Module: V2TXLivePremier @ TXLiteAVSDK

Function: V2TXLive 高级接口

V2TXLivePremier

V2TXLivePremier

函数列表	描述
getSDKVersionStr	获取 SDK 版本号
setObserver:	设置 V2TXLivePremier 回调接口
setLogConfig:	设置 Log 的配置信息
setEnvironment:	设置 SDK 接入环境
setLicence:key:	设置 SDK 的授权 License
setSocks5Proxy:port:username:password:config:	设置 SDK socks5 代理配置
enableAudioCaptureObserver:format:	开启/关闭对音频采集数据的监听回调（可读写）
enableAudioPlayoutObserver:format:	开启/关闭对最终系统要播放出的音频数据的监听回调
enableVoiceEarMonitorObserver:	开启/关闭耳返音频数据的监听回调
setUserId:	设置 userId
callExperimentalAPI:	调用实验性 API 接口

V2TXLivePremierObserver

函数列表	描述
onLog:log:	自定义 Log 输出回调接口
onLicenceLoaded:Reason:	setLicence 接口回调
onCaptureAudioFrame:	本地麦克风采集到的音频数据回调
onPlayoutAudioFrame:	将各路待播放音频混合之后并在最终提交系统播放之前的数据回调
onVoiceEarMonitorAudioFrame:	耳返的音频数据

getSDKVersionStr

getSDKVersionStr

获取 SDK 版本号

setObserver:

setObserver:

```
+ (void)setObserver:(id<V2TXLivePremierObserver>)observer
```

设置 V2TXLivePremier 回调接口

setLogConfig:

setLogConfig:

```
+ (V2TXLiveCode)setLogConfig:(V2TXLiveLogConfig *)config
```

设置 Log 的配置信息

setEnvironment:

setEnvironment:

```
+ (V2TXLiveCode)setEnvironment:      (const char *)env
```

设置 SDK 接入环境

参数	描述
env	目前支持 “default” 和 “GDPR” 两个参数。 • default：默认环境，SDK 会在全球寻找最佳接入点进行接入。 • GDPR：所有音视频数据和质量统计数据都不会经过中国大陆地区的服务器。

注意
如您的应用无特殊需求，请不要调用此接口进行设置。

setLicence:key:

setLicence:key:

```
+ (void)setLicence:      (NSString *)url
               key:      (NSString *)key
```

设置 SDK 的授权 License

文档地址：https://cloud.tencent.com/document/product/454/34750。

参数	描述
key	licence的秘钥。
url	licence的地址。

setSocks5Proxy:port:username:password:config:

setSocks5Proxy:port:username:password:config:

```
+ (V2TXLiveCode)setSocks5Proxy:    (NSString *)host
                          port:    (NSInteger)port
                      username:    (NSString *)username
                      password:    (NSString *)password
                        config:    (V2TXLiveSocks5ProxyConfig *)config
```

设置 SDK socks5 代理配置

参数	描述
config	配置使用 socks5 代理服务器的协议。
host	socks5 代理服务器的地址。
password	socks5 代理服务器的验证的密码。
port	socks5 代理服务器的端口。
username	socks5 代理服务器的验证的用户名。

enableAudioCaptureObserver:format:

enableAudioCaptureObserver:format:

```
+ (V2TXLiveCode)enableAudioCaptureObserver:    (BOOL)enable
                                    format:    (V2TXLiveAudioFrameObserverFo
```

开启/关闭对音频采集数据的监听回调（可读写）

参数	描述
enable	是否开启。【默认值】：false。

format	设置回调出的 AudioFrame 的格式。

注意
需要在 startPush 之前调用，才会生效。

enableAudioPlayoutObserver:format:

enableAudioPlayoutObserver:format:

```
+ (V2TXLiveCode)enableAudioPlayoutObserver:    (BOOL)enable
                                    format:    (V2TXLiveAudioFrameObserverFo
```

开启/关闭对最终系统要播放出的音频数据的监听回调

参数	描述
enable	是否开启。【默认值】：false。
format	设置回调出的 AudioFrame 的格式。

enableVoiceEarMonitorObserver:

enableVoiceEarMonitorObserver:

```
+ (V2TXLiveCode)enableVoiceEarMonitorObserver:    (BOOL)enable
```

开启/关闭耳返音频数据的监听回调

参数	描述
enable	是否开启。【默认值】：false。

setUserId:

setUserId:

```
+ (void)setUserId:      (NSString *)userId
```

设置 userId

参数	描述
userId	业务侧自身维护的用户/设备id。

callExperimentalAPI:

callExperimentalAPI:

```
+ (V2TXLiveCode)callExperimentalAPI:      (NSString *)jsonStr
```

调用实验性 API 接口

参数	描述
jsonStr	接口及参数描述的 JSON 字符串。

注意
该接口用于调用一些实验性功能。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，参数非法。

onLog:log:

onLog:log:

- (void)onLog:	(V2TXLiveLogLevel)level
log:	(NSString *)log

自定义 Log 输出回调接口

onLicenceLoaded:Reason:

onLicenceLoaded:Reason:

- (void)onLicenceLoaded:	(int)result
Reason:	(NSString *)reason

setLicence 接口回调

参数	描述
reason	设置 licence 失败原因。
result	设置 licence 结果 0 成功，负数失败。

onCaptureAudioFrame:

onCaptureAudioFrame:

- (void) onCaptureAudioFrame:	(V2TXLiveAudioFrame *)frame

本地麦克风采集到的音频数据回调

参数	描述
frame	音频数据。

注意

- 请不要在此回调函数中做任何耗时操作，建议直接拷贝到另一线程进行处理，否则会导致各种声音问题。
- 此接口回调出的音频数据支持修改。
- 此接口回调出的音频时间帧长固定为0.02s。

由时间帧长转化为字节帧长的公式为【采样率 × 时间帧长 × 声道数 × 采样点位宽】。
以SDK默认的音频录制格式48000采样率、单声道、16采样点位宽为例，字节帧长为【48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节】。

- 此接口回调出的音频数据**不包含**背景音、音效、混响等前处理效果，延迟极低。
- 需要您调用 enableAudioCaptureObserver 开启回调开关。

onPlayoutAudioFrame:

onPlayoutAudioFrame:

```
- (void) onPlayoutAudioFrame:      (V2TXLiveAudioFrame *)frame
```

将各路待播放音频混合之后并在最终提交系统播放之前的数据回调

当您设置完音频数据自定义回调之后，SDK 内部会把各路待播放的音频混合之后的音频数据，在提交系统播放之前，以 PCM 格式的形式通过本接口回调给您。

- 此接口回调出的音频时间帧长固定为 0.02s，格式为 PCM 格式。
- 由时间帧长转化为字节帧长的公式为 `采样率 × 时间帧长 × 声道数 × 采样点位宽` 。
- 以 SDK 默认的音频录制格式 48000 采样率、单声道、16 采样点位宽为例，字节帧长为 `48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节` 。

参数	描述
frame	PCM 格式的音频数据帧。

注意

1. 请不要在此回调函数中做任何耗时操作，由于 SDK 每隔 20ms 就要处理一帧音频数据，如果您的处理时间超过 20ms，就会导致声音异常。
2. 此接口回调出的音频数据是可读写的，也就是说您可以在回调函数中同步修改音频数据，但请保证处理耗时。
3. 此接口回调出的是对各路待播放音频数据的混合，但其中并不包含耳返的音频数据。

onVoiceEarMonitorAudioFrame:

onVoiceEarMonitorAudioFrame:

- (void) onVoiceEarMonitorAudioFrame: (V2TXLiveAudioFrame *)frame

耳返的音频数据

当您设置完音频数据自定义回调之后，SDK 内部会把耳返的音频数据在播放之前以 PCM 格式的形式通过本接口回调给您。

- 此接口回调出的音频时间帧长不固定，格式为 PCM 格式。
- 由时间帧长转化为字节帧长的公式为 `采样率 × 时间帧长 × 声道数 × 采样点位宽` 。
- 以 TRTC 默认的音频录制格式 48000 采样率、单声道、16采样点位宽为例，0.02s 的音频数据字节帧长为 `48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节` 。

参数	描述
frame	PCM 格式的音频数据帧。

注意

1. 请不要在此回调函数中做任何耗时操作，否则会导致声音异常。
2. 此接口回调出的音频数据是可读写的，也就是说您可以在回调函数中同步修改音频数据，但请保证处理耗时。

TXAudioEffectManager

最近更新时间：2024-03-07 15:14:11

Module: 背景音乐、短音效和人声特效的管理类

Function: 用于对背景音乐、短音效和人声特效进行设置的管理类

TXAudioEffectManager

TXAudioEffectManager

函数列表	描述
enableVoiceEarMonitor:	开启耳返
setVoiceEarMonitorVolume:	设置耳返音量
setVoiceReverbType:	设置人声的混响效果
setVoiceChangerType:	设置人声的变声特效
setVoiceVolume:	设置语音音量
setVoicePitch:	设置语音音调
startPlayMusic:onStart:onProgress:onComplete:	开始播放背景音乐
stopPlayMusic:	停止播放背景音乐
pausePlayMusic:	暂停播放背景音乐
resumePlayMusic:	恢复播放背景音乐
setAllMusicVolume:	设置所有背景音乐的本地音量和远端音量的大小
setMusicPublishVolume:volume:	设置某一首背景音乐的远端音量的大小
setMusicPlayoutVolume:volume:	设置某一首背景音乐的本地音量的大小
setMusicPitch:pitch:	调整背景音乐的音调高低

setMusicSpeedRate:speedRate:	调整背景音乐的变速效果
getMusicCurrentPosInMS:	获取背景音乐的播放进度（单位：毫秒）
getMusicDurationInMS:	获取背景音乐的总时长（单位：毫秒）
seekMusicToPosInMS:pts:	设置背景音乐的播放进度（单位：毫秒）
setMusicScratchSpeedRate:speedRate:	调整搓碟的变速效果
preloadMusic:onProgress:onError:	预加载背景音乐
getMusicTrackCount:	获取背景音乐的音轨数量
setMusicTrack:track:	指定背景音乐的播放音轨

结构体类型

函数列表	描述
TXAudioMusicParam	背景音乐的播放控制信息

枚举类型

枚举类型	描述
TXVoiceReverbType	混响特效
TXVoiceChangeType	变声特效

enableVoiceEarMonitor:

enableVoiceEarMonitor:

```
- (void)enableVoiceEarMonitor:    (BOOL)enable
```

开启耳返

主播开启耳返后，可以在耳机里听到麦克风采集到的自己发出的声音，该特效适用于主播唱歌的应用场景中。

需要您注意的是，由于蓝牙耳机的硬件延迟非常高，所以在主播佩戴蓝牙耳机时无法开启此特效，请尽量在用户界面上提示主播佩戴有线耳机。
同时也需要注意，并非所有的手机开启此特效后都能达到优秀的耳返效果，我们已经对部分耳返效果不佳的手机屏蔽了该特效。

参数	描述
enable	YES：开启；NO：关闭。

注意
仅在主播佩戴耳机时才能开启此特效，同时请您提示主播佩戴有线耳机。

setVoiceEarMonitorVolume:

setVoiceEarMonitorVolume:

```
- (void)setVoiceEarMonitorVolume:(NSInteger)volume
```

设置耳返音量

通过该接口您可以设置耳返特效中声音的音量大小。

参数	描述
volume	音量大小，取值范围为 0 – 100，默认值：100。

注意
如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setVoiceReverbType:

setVoiceReverbType:

```
- (void)setVoiceReverbType:(TXVoiceReverbType)reverbType
```

设置人声的混响效果

通过该接口您可以设置人声的混响效果，具体特效请参见枚举定义 TXVoiceReverbType 。

> **注意**
> 设置的效果在退出房间后会自动失效，如果下次进房还需要对应特效，需要调用此接口再次进行设置。

setVoiceChangerType:

setVoiceChangerType:

```
- (void)setVoiceChangerType:(TXVoiceChangeType)changerType
```

设置人声的变声特效

通过该接口您可以设置人声的变声特效，具体特效请参见枚举定义 TXVoiceChangeType 。

> **注意**
> 设置的效果在退出房间后会自动失效，如果下次进房还需要对应特效，需要调用此接口再次进行设置。

setVoiceVolume:

setVoiceVolume:

```
- (void)setVoiceVolume:(NSInteger)volume
```

设置语音音量

该接口可以设置语音音量的大小，一般配合音乐音量的设置接口 setAllMusicVolume 协同使用，用于调谐语音和音乐在混音前各自的音量占比。

参数	描述
volume	音量大小，取值范围为0 - 100，默认值：100。

> **注意**
> 如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setVoicePitch:

setVoicePitch:

```
-(void)setVoicePitch:    (double)pitch
```

设置语音音调

该接口可以设置语音音调，用于实现变调不变速的目的。

参数	描述
pitch	音调，取值范围为-1.0f~1.0f，默认值：0.0f。

startPlayMusic:onStart:onProgress:onComplete:

startPlayMusic:onStart:onProgress:onComplete:

```
- (void)startPlayMusic:    (TXAudioMusicParam *)musicParam
onStart:    (TXAudioMusicStartBlock _Nullable)startB
onProgress:    (TXAudioMusicProgressBlock _Nullable)p
onComplete:    (TXAudioMusicCompleteBlock _Nullable)c
```

开始播放背景音乐

每个音乐都需要您指定具体的 ID，您可以通过该 ID 对音乐的开始、停止、音量等进行设置。

参数	描述
completeBloc	播放结束回调。

k	
musicParam	音乐参数。
progressBlock	播放进度回调。
startBlock	播放开始回调。

注意

1. 如果要多次播放同一首背景音乐，请不要每次播放都分配一个新的 ID，我们推荐使用相同的 ID。
2. 若您希望同时播放多首不同的音乐，请为不同的音乐分配不同的 ID 进行播放。
3. 如果使用同一个 ID 播放不同音乐，SDK 会先停止播放旧的音乐，再播放新的音乐。

stopPlayMusic:

stopPlayMusic:

```
- (void)stopPlayMusic:    (int32_t)id
```

停止播放背景音乐

参数	描述
id	音乐 ID。

pausePlayMusic:

pausePlayMusic:

```
- (void)pausePlayMusic:    (int32_t)id
```

暂停播放背景音乐

参数	描述
id	音乐 ID。

resumePlayMusic:

resumePlayMusic:

- (void)resumePlayMusic: (int32_t)id

恢复播放背景音乐

参数	描述
id	音乐 ID。

setAllMusicVolume:

setAllMusicVolume:

- (void)setAllMusicVolume: (NSInteger)volume

设置所有背景音乐的本地音量和远端音量的大小

该接口可以设置所有背景音乐的本地音量和远端音量。

- 本地音量：即主播本地可以听到的背景音乐的音量大小。
- 远端音量：即观众端可以听到的背景音乐的音量大小。

参数	描述
volume	音量大小，取值范围为0 - 100，默认值：60。

注意
如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setMusicPublishVolume:volume:

setMusicPublishVolume:volume:

```
- (void)setMusicPublishVolume:    (int32_t)id
                           volume:    (NSInteger)volume
```

设置某一首背景音乐的远端音量的大小

该接口可以细粒度地控制每一首背景音乐的远端音量，也就是观众端可听到的背景音乐的音量大小。

参数	描述
id	音乐 ID。
volume	音量大小，取值范围为0 - 100；默认值：60。

注意

如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setMusicPlayoutVolume:volume:

setMusicPlayoutVolume:volume:

```
- (void)setMusicPlayoutVolume:    (int32_t)id
                           volume:    (NSInteger)volume
```

设置某一首背景音乐的本地音量的大小

该接口可以细粒度地控制每一首背景音乐的本地音量，也就是主播本地可以听到的背景音乐的音量大小。

参数	描述
id	音乐 ID。
volume	音量大小，取值范围为0 - 100，默认值：60。

注意

> 如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setMusicPitch:pitch:

setMusicPitch:pitch:

```
- (void)setMusicPitch:    (int32_t)id
                pitch:    (double)pitch
```

调整背景音乐的音调高低

参数	描述
id	音乐 ID。
pitch	音调，默认值是0.0f，范围是：[-1 ~ 1] 之间的浮点数。

setMusicSpeedRate:speedRate:

setMusicSpeedRate:speedRate:

```
- (void)setMusicSpeedRate:    (int32_t)id
                speedRate:    (double)speedRate
```

调整背景音乐的变速效果

参数	描述
id	音乐 ID。
speedRate	速度，默认值是1.0f，范围是：[0.5 ~ 2] 之间的浮点数。

getMusicCurrentPosInMS:

getMusicCurrentPosInMS:

```
- (NSInteger)getMusicCurrentPosInMS:    (int32_t)id
```

获取背景音乐的播放进度（单位：毫秒）

参数	描述
id	音乐 ID。

返回值说明：

成功返回当前播放时间，单位：毫秒，失败返回 -1。

getMusicDurationInMS:

getMusicDurationInMS:

```
- (NSInteger)getMusicDurationInMS:    (NSString *)path
```

获取背景音乐的总时长（单位：毫秒）

参数	描述
path	音乐文件路径。

返回值说明：

成功返回时长，失败返回 -1。

seekMusicToPosInMS:pts:

seekMusicToPosInMS:pts:

```
- (void)seekMusicToPosInMS:    (int32_t)id
                       pts:    (NSInteger)pts
```

设置背景音乐的播放进度（单位：毫秒）

参数	描述
id	音乐 ID。
pts	单位: 毫秒。

注意

请尽量避免过度频繁地调用该接口，因为该接口可能会再次读写音乐文件，耗时稍高。
因此，当用户拖拽音乐的播放进度条时，请在用户完成拖拽操作后再调用本接口。
因为 UI 上的进度条控件往往会以很高的频率反馈用户的拖拽进度，如不做频率限制，会导致较差的用户体验。

setMusicScratchSpeedRate:speedRate:

setMusicScratchSpeedRate:speedRate:

```
- (void)setMusicScratchSpeedRate:    (int32_t)id
                       speedRate:    (double)scratchSpeedRate
```

调整搓碟的变速效果

参数	描述
id	音乐 ID。
scratchSpeedRate	搓碟速度，默认值是1.0f，范围是：[-12.0 ~ 12.0] 之间的浮点数, 速度值正/负表示方向正/反，绝对值大小表示速度快慢。

注意

前置条件 preloadMusic 成功。

preloadMusic:onProgress:onError:

preloadMusic:onProgress:onError:

```
- (void)preloadMusic:    (TXAudioMusicParam *)preloadParam
```

```
                                onProgress:    (TXMusicPreloadProgressBlock _Nullable
                                onError:       (TXMusicPreloadErrorBlock _Nullable)err
```

预加载背景音乐

每个音乐都需要您指定具体的 ID，您可以通过该 ID 对音乐的开始、停止、音量等进行设置。

参数	描述
preloadParam	预加载音乐参数。

注意

1. 预先加载最多同时支持2个不同 ID 的预加载，且预加载时长不超过10分钟，使用完需 stopPlayMusic，否则内存不释放。
2. 若该ID对应的音乐正在播放中，预加载会失败，需先调用 stopPlayMusic。
3. 当 musicParam 和传入 startPlayMusic 的 musicParam 完全相同时，预加载有效。

getMusicTrackCount:

getMusicTrackCount:

```
- (NSInteger)getMusicTrackCount:    (int32_t)id
```

获取背景音乐的音轨数量

参数	描述
id	音乐 ID。

setMusicTrack:track:

setMusicTrack:track:

```
- (void)setMusicTrack:    (int32_t)id
                track:    (NSInteger)track
```

指定背景音乐的播放音轨

参数	描述
id	音乐 ID。
index	默认播放第一个音轨。取值范围[0, 音轨总数)。

> **注意**
> 音轨总数量可通过 getMusicTrackCount 接口获取。

TXVoiceReverbType

TXVoiceReverbType

混响特效

混响特效可以作用于人声之上，通过声学算法对声音进行叠加处理，模拟出各种不同环境下的临场感受，目前支持如下几种混响效果：
0：关闭；1：KTV；2：小房间；3：大会堂；4：低沉；5：洪亮；6：金属声；7：磁性；8：空灵；9：录音棚；10：悠扬 11：录音棚2。

枚举	取值	描述
TXVoiceReverbType_0	0	关闭特效
TXVoiceReverbType_1	1	KTV
TXVoiceReverbType_2	2	小房间
TXVoiceReverbType_3	3	大会堂
TXVoiceReverbType_4	4	低沉
TXVoiceReverbTy	5	洪亮

pe_5		
TXVoiceReverbType_6	6	金属声
TXVoiceReverbType_7	7	磁性
TXVoiceReverbType_8	8	空灵
TXVoiceReverbType_9	9	录音棚
TXVoiceReverbType_10	10	悠扬
TXVoiceReverbType_11	11	录音棚2

TXVoiceChangeType

TXVoiceChangeType

变声特效

变声特效可以作用于人声之上，通过声学算法对人声进行二次处理，以获得与原始声音所不同的音色，目前支持如下几种变声特效：
0：关闭；1：熊孩子；2：萝莉；3：大叔；4：重金属；5：感冒；6：外语腔；7：困兽；8：肥宅；9：强电流；10：重机械；11：空灵。

枚举	取值	描述
TXVoiceChangeType_0	0	关闭
TXVoiceChangeType_1	1	熊孩子
TXVoiceChangeType_2	2	萝莉
TXVoiceChangeType_3	3	大叔

TXVoiceChangeType_4	4	重金属
TXVoiceChangeType_5	5	感冒
TXVoiceChangeType_6	6	外语腔
TXVoiceChangeType_7	7	困兽
TXVoiceChangeType_8	8	肥宅
TXVoiceChangeType_9	9	强电流
TXVoiceChangeType_10	10	重机械
TXVoiceChangeType_11	11	空灵

TXAudioMusicParam

TXAudioMusicParam

背景音乐的播放控制信息

该信息用于在接口 startPlayMusic 中指定背景音乐的相关信息，包括播放 ID、文件路径和循环次数等：
1. 如果要多次播放同一首背景音乐，请不要每次播放都分配一个新的 ID，我们推荐使用相同的 ID。
2. 若您希望同时播放多首不同的音乐，请为不同的音乐分配不同的 ID 进行播放。
3. 如果使用同一个 ID 播放不同音乐，SDK 会先停止播放旧的音乐，再播放新的音乐。

枚举类型	描述
ID	【字段含义】音乐 ID。 【特殊说明】SDK 允许播放多路音乐，因此需要使用 ID 进行标记，用于控制音乐的开始、停止、音量等。
endTimeMS	【字段含义】音乐结束播放时间点，单位毫秒，0表示播放至文件结尾。

isShortFile	【字段含义】播放的是否为短音乐文件。 【推荐取值】YES：需要重复播放的短音乐文件；NO：正常的音乐文件。默认值：NO。
loopCount	【字段含义】音乐循环播放的次数。 【推荐取值】取值范围为0 - 任意正整数，默认值：0。0 表示播放音乐一次；1 表示播放音乐两次；以此类推。
path	【字段含义】音效文件的完整路径或 URL 地址。支持的音频格式包括 MP3、AAC、M4A、WAV。
publish	【字段含义】是否将音乐传到远端。 【推荐取值】YES：音乐在本地播放的同时，远端用户也能听到该音乐；NO：主播只能在本地听到该音乐，远端观众听不到。默认值：NO。
startTimeMS	【字段含义】音乐开始播放时间点，单位：毫秒。

TXBeautyManager

最近更新时间：2024-03-07 15:14:11

Module: 美颜与图像处理参数设置类

Function: 修改美颜、滤镜、绿幕等参数

TXBeautyManager

TXBeautyManager

函数列表	描述
setBeautyStyle:	设置美颜（磨皮）算法
setBeautyLevel:	设置美颜级别
setWhitenessLevel:	设置美白级别
enableSharpnessEnhancement:	开启清晰度增强
setRuddyLevel:	设置红润级别
setFilter:	设置色彩滤镜效果
setFilterStrength:	设置色彩滤镜的强度
setGreenScreenFile:	设置绿幕背景视频
setEyeScaleLevel:	设置大眼级别
setFaceSlimLevel:	设置瘦脸级别
setFaceVLevel:	设置 V 脸级别
setChinLevel:	设置下巴拉伸或收缩
setFaceShortLevel:	设置短脸级别
setFaceNarrowLevel:	设置窄脸级别

setNoseSlimLevel:	设置瘦鼻级别
setEyeLightenLevel:	设置亮眼级别
setToothWhitenLevel:	设置牙齿美白级别
setWrinkleRemoveLevel:	设置祛皱级别
setPounchRemoveLevel:	设置祛眼袋级别
setSmileLinesRemoveLevel:	设置法令纹去除级别
setForeheadLevel:	设置发际线调整级别
setEyeDistanceLevel:	设置眼距
setEyeAngleLevel:	设置眼角调整级别
setMouthShapeLevel:	设置嘴型调整级别
setNoseWingLevel:	设置鼻翼调整级别
setNosePositionLevel:	设置鼻子位置
setLipsThicknessLevel:	设置嘴唇厚度
setFaceBeautyLevel:	设置脸型
setMotionTmpl:inDir:	选择 AI 动效挂件
setMotionMute:	是否在动效素材播放时静音

枚举类型

枚举类型	描述
TXBeautyStyle	美颜（磨皮）算法

setBeautyStyle:

setBeautyStyle:

- (void)setBeautyStyle: (TXBeautyStyle)beautyStyle

设置美颜（磨皮）算法

TRTC 内置多种不同的磨皮算法，您可以选择最适合您产品定位的方案：

参数	描述
beautyStyle	美颜风格，TXBeautyStyleSmooth：光滑；TXBeautyStyleNature：自然；TXBeautyStylePitu：优图。

setBeautyLevel:

setBeautyLevel:

- (void)setBeautyLevel: (float)beautyLevel

设置美颜级别

参数	描述
beautyLevel	美颜级别，取值范围 0 - 9； 0 表示关闭，9 表示效果最明显。

setWhitenessLevel:

setWhitenessLevel:

- (void)setWhitenessLevel: (float)whitenessLevel

设置美白级别

参数	描述
whitenessLevel	美白级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

enableSharpnessEnhancement:

enableSharpnessEnhancement:

```
- (void)enableSharpnessEnhancement:(BOOL)enable
```

开启清晰度增强

setRuddyLevel:

setRuddyLevel:

```
- (void)setRuddyLevel:(float)ruddyLevel
```

设置红润级别

参数	描述
ruddyLevel	红润级别，取值范围0 - 9；0 表示关闭，9 表示效果最明显。

setFilter:

setFilter:

```
- (void)setFilter:(nullable TXImage *)image
```

设置色彩滤镜效果

色彩滤镜，是一副包含色彩映射关系的颜色查找表图片，您可以在我们提供的官方 Demo 中找到预先准备好的几张滤镜图片。
SDK 会根据该查找表中的映射关系，对摄像头采集出的原始视频画面进行二次处理，以达到预期的滤镜效果。

参数	描述
ima	包含色彩映射关系的颜色查找表图片，必须是 png 格式。

ge	

setFilterStrength:

setFilterStrength:

- (void)setFilterStrength: (float)strength

设置色彩滤镜的强度

该数值越高，色彩滤镜的作用强度越明显，经过滤镜处理后的视频画面跟原画面的颜色差异越大。
我默认的滤镜浓度是 0.5，如果您觉得默认的滤镜效果不明显，可以设置为 0.5 以上的数字，最大值为 1。

参数	描述
strength	从 0 到 1，数值越大滤镜效果越明显，默认值为 0.5。

setGreenScreenFile:

setGreenScreenFile:

- (int)setGreenScreenFile: (nullable NSString *)path

设置绿幕背景视频

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见 腾讯美颜特效SDK ）中生效。此接口所开启的绿幕功能不具备智能去除背景的能力，需要被拍摄者的背后有一块绿色的幕布来辅助产生特效。

参数	描述
path	MP4格式的视频文件路径; 设置空值表示关闭特效。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeScaleLevel:

setEyeScaleLevel:

- (int)setEyeScaleLevel: (float)eyeScaleLevel

设置大眼级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeScaleLevel	大眼级别，取值范围 0 – 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceSlimLevel:

setFaceSlimLevel:

- (int)setFaceSlimLevel: (float)faceSlimLevel

设置瘦脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
faceSlimLevel	瘦脸级别，取值范围0 – 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceVLevel:

setFaceVLevel:

- (int)setFaceVLevel: (float)faceVLevel

设置 V 脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
faceVLevel	V 脸级别，取值范围 0 – 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setChinLevel:

setChinLevel:

- (int)setChinLevel: (float)chinLevel

设置下巴拉伸或收缩

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
chinLevel	下巴拉伸或收缩级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示收缩，大于 0 表示拉伸。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceShortLevel:

setFaceShortLevel:

- (int)setFaceShortLevel: (float)faceShortLevel

设置短脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
faceShortLevel	短脸级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceNarrowLevel:

setFaceNarrowLevel:

- (int)setFaceNarrowLevel: (float)faceNarrowLevel

设置窄脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
level	窄脸级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setNoseSlimLevel:

setNoseSlimLevel:

- (int)setNoseSlimLevel: (float)noseSlimLevel

设置瘦鼻级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
noseSlimLevel	瘦鼻级别，取值范围0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeLightenLevel:

setEyeLightenLevel:

```
- (int)setEyeLightenLevel:      (float)eyeLightenLevel
```

设置亮眼级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeLightenLevel	亮眼级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setToothWhitenLevel:

setToothWhitenLevel:

```
- (int)setToothWhitenLevel:      (float)toothWhitenLevel
```

设置牙齿美白级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
toothWhitenLevel	白牙级别，取值范围 0 - 9；0表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setWrinkleRemoveLevel:

setWrinkleRemoveLevel:

```
- (int)setWrinkleRemoveLevel:      (float)wrinkleRemoveLevel
```

设置祛皱级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
wrinkleRemoveLevel	祛皱级别，取值范围0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setPounchRemoveLevel:

setPounchRemoveLevel:

```
- (int)setPounchRemoveLevel:      (float)pounchRemoveLevel
```

设置祛眼袋级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
pounchRemoveLevel	祛眼袋级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setSmileLinesRemoveLevel:

setSmileLinesRemoveLevel:

```
- (int)setSmileLinesRemoveLevel:      (float)smileLinesRemoveLevel
```

设置法令纹去除级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
smileLinesRemoveLevel	法令纹级别，取值范围 0 - 9；0表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setForeheadLevel:

setForeheadLevel:

```
- (int)setForeheadLevel:      (float)foreheadLevel
```

设置发际线调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
foreheadLevel	发际线级别，取值范围-9 - 9；0表示关闭，9表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeDistanceLevel:

setEyeDistanceLevel:

```
- (int)setEyeDistanceLevel:      (float)eyeDistanceLevel
```

设置眼距

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeDistanceLevel	眼距级别，取值范围 -9 - 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeAngleLevel:

setEyeAngleLevel:

```
- (int)setEyeAngleLevel:      (float)eyeAngleLevel
```

设置眼角调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeAngleLevel	眼角调整级别，取值范围-9 - 9；0表示关闭，9表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setMouthShapeLevel:

setMouthShapeLevel:

- (int)setMouthShapeLevel: (float)mouthShapeLevel

设置嘴型调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
mouthShapeLevel	嘴型级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setNoseWingLevel:

setNoseWingLevel:

- (int)setNoseWingLevel: (float)noseWingLevel

设置鼻翼调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
noseWingLevel	鼻翼调整级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setNosePositionLevel:

setNosePositionLevel:

- (int)setNosePositionLevel: (float)nosePositionLevel

设置鼻子位置

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
nosePositionLevel	鼻子位置级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示抬高，大于 0 表示降低。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setLipsThicknessLevel:

setLipsThicknessLevel:

- (int)setLipsThicknessLevel: (float)lipsThicknessLevel

设置嘴唇厚度

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
lipsThicknessLevel	嘴唇厚度级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceBeautyLevel:

setFaceBeautyLevel:

```
- (int)setFaceBeautyLevel:    (float)faceBeautyLevel
```

设置脸型

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
faceBeautyLevel	美型级别，取值范围 0 - 9；0 表示关闭，1 - 9 值越大，效果越明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setMotionTmpl:inDir:

setMotionTmpl:inDir:

```
- (void)setMotionTmpl:    (nullable NSString *)tmplName
                inDir:    (nullable NSString *)tmplDir
```

选择 AI 动效挂件

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
tmplDir	动效素材文件所在目录。
tmplName	动效挂件名称。

setMotionMute:

setMotionMute:

- (void)setMotionMute: (BOOL)motionMute

是否在动效素材播放时静音

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。
有些挂件本身会有声音特效，通过此 API 可以关闭这些特效播放时所带的声音效果。

参数	描述
motionMute	YES：静音；NO：不静音。

TXBeautyStyle

TXBeautyStyle

美颜（磨皮）算法

TRTC 内置多种不同的磨皮算法，您可以选择最适合您产品定位的方案。

枚举	取值	描述
TXBeautyStyleSmooth	0	光滑，算法比较激进，磨皮效果比较明显，适用于秀场直播。
TXBeautyStyleNature	1	自然，算法更多地保留了面部细节，磨皮效果更加自然，适用于绝大多数直播场景。
TXBeautyStylePitu	2	优图，由优图实验室提供，磨皮效果介于光滑和自然之间，比光滑保留更多皮肤细节，比自然磨皮程度更高。

TXDeviceManager

最近更新时间：2024-03-07 15:14:11

Module: 音视频设备管理模块

Function: 用于管理摄像头、麦克风和扬声器等音视频相关的硬件设备

TXDeviceManager

TXDeviceObserver

函数列表	描述
onDeviceChanged:type:state:	本地设备的通断状态发生变化（仅适用于桌面系统）

TXDeviceManager

函数列表	描述
isFrontCamera	判断当前是否为前置摄像头（仅适用于移动端）
switchCamera:	切换前置或后置摄像头（仅适用于移动端）
isCameraZoomSupported	查询当前摄像头是否支持缩放（仅适用于移动端）
getCameraZoomMaxRatio	获取摄像头的最大缩放倍数（仅适用于移动端）
setCameraZoomRatio:	设置摄像头的缩放倍数（仅适用于移动端）
isAutoFocusEnabled	查询是否支持自动识别人脸位置（仅适用于移动端）
enableCameraAutoFocus:	开启自动对焦功能（仅适用于移动端）
setCameraFocusPosition:	设置摄像头的对焦位置（仅适用于移动端）
isCameraTorchSupported	查询是否支持开启闪光灯（仅适用于移动端）
enableCameraTorch:	开启/关闭闪光灯，也就是手电筒模式（仅适用于移动端）

setAudioRoute:	设置音频路由（仅适用于移动端）
setExposureCompensation:	设置摄像头的曝光参数，取值范围从-1到1
getDevicesList:	获取设备列表（仅适用于桌面端）
setCurrentDevice:deviceId:	设置当前要使用的设备（仅适用于桌面端）
getCurrentDevice:	获取当前正在使用的设备（仅适用于桌面端）
setCurrentDeviceVolume:deviceType:	设置当前设备的音量（仅适用于桌面端）
getCurrentDeviceVolume:	获取当前设备的音量（仅适用于桌面端）
setCurrentDeviceMute:deviceType:	设置当前设备的静音状态（仅适用于桌面端）
getCurrentDeviceMute:	获取当前设备的静音状态（仅适用于桌面端）
enableFollowingDefaultAudioDevice:enable:	设置 SDK 使用的音频设备根据跟随系统默认设备（仅适用于桌面端）
startCameraDeviceTest:	开始摄像头测试（仅适用于桌面端）
stopCameraDeviceTest	结束摄像头测试（仅适用于桌面端）
startMicDeviceTest:	开始麦克风测试（仅适用于桌面端）
startMicDeviceTest:playback:	开始麦克风测试（仅适用于桌面端）
stopMicDeviceTest	结束麦克风测试（仅适用于桌面端）
startSpeakerDeviceTest:	开始扬声器测试（仅适用于桌面端）
stopSpeakerDeviceTest	结束扬声器测试（仅适用于桌面端）
setObserver:	设备热插拔回调（仅适用于 Mac 系统）
setSystemVolumeType:	设置系统音量类型（仅适用于移动端）

结构体类型

函数列表	描述
TXMediaDeviceInfo	音视频设备的相关信息（仅适用于桌面平台）

枚举类型

枚举类型	描述
TXSystemVolumeType	系统音量类型（仅适用于移动设备）
TXAudioRoute	音频路由（即声音的播放模式）
TXMediaDeviceType	设备类型（仅适用于桌面平台）
TXMediaDeviceState	设备操作

onDeviceChanged:type:state:

onDeviceChanged:type:state:

```
- (void)onDeviceChanged:    (NSString*)deviceId
                   type:    (TXMediaDeviceType)mediaType
                  state:    (TXMediaDeviceState)mediaState
```

本地设备的通断状态发生变化（仅适用于桌面系统）

当本地设备（包括摄像头、麦克风以及扬声器）被插入或者拔出时，SDK 便会抛出此事件回调。

参数	描述
deviceId	设备 ID
state	通断状态，0：设备已添加；1：设备已被移除；2：设备已启用。
type	设备类型

isFrontCamera

isFrontCamera

判断当前是否为前置摄像头（仅适用于移动端）

switchCamera:

switchCamera:

```
- (NSInteger)switchCamera:    (BOOL)frontCamera
```

切换前置或后置摄像头（仅适用于移动端）

isCameraZoomSupported

isCameraZoomSupported

查询当前摄像头是否支持缩放（仅适用于移动端）

getCameraZoomMaxRatio

getCameraZoomMaxRatio

获取摄像头的最大缩放倍数（仅适用于移动端）

setCameraZoomRatio:

setCameraZoomRatio:

```
- (NSInteger)setCameraZoomRatio:    (CGFloat)zoomRatio
```

设置摄像头的缩放倍数（仅适用于移动端）

参数	描述
zoomRat	取值范围1 - 5，取值为1表示最远视角（正常镜头），取值为5表示最近视角（放大镜

io	头）。最大值推荐为5，若超过5，视频数据会变得模糊不清。

isAutoFocusEnabled

isAutoFocusEnabled

查询是否支持自动识别人脸位置（仅适用于移动端）

enableCameraAutoFocus:

enableCameraAutoFocus:

```
- (NSInteger)enableCameraAutoFocus:    (BOOL)enabled
```

开启自动对焦功能（仅适用于移动端）

开启后，SDK 会自动检测画面中的人脸位置，并将摄像头的焦点始终对焦在人脸位置上。

setCameraFocusPosition:

setCameraFocusPosition:

```
- (NSInteger)setCameraFocusPosition:    (CGPoint)position
```

设置摄像头的对焦位置（仅适用于移动端）

您可以通过该接口实现如下交互：
1. 在本地摄像头的预览画面上，允许用户单击操作。
2. 在用户的单击位置显示一个矩形方框，以示摄像头会在此处对焦。
3. 随后将用户点击位置的坐标通过本接口传递给 SDK，之后 SDK 会操控摄像头按照用户期望的位置进行对焦。

参数	描述
position	对焦位置，请传入期望对焦点的坐标值

注意

使用该接口的前提是先通过 enableCameraAutoFocus 关闭自动对焦功能。

返回值说明：

0：操作成功；负数：操作失败。

isCameraTorchSupported

isCameraTorchSupported

查询是否支持开启闪光灯（仅适用于移动端）

enableCameraTorch:

enableCameraTorch:

```
- (NSInteger)enableCameraTorch:(BOOL)enabled
```

开启/关闭闪光灯，也就是手电筒模式（仅适用于移动端）

setAudioRoute:

setAudioRoute:

```
- (NSInteger)setAudioRoute:(TXAudioRoute)route
```

设置音频路由（仅适用于移动端）

手机有两个音频播放设备：一个是位于手机顶部的听筒，一个是位于手机底部的立体声扬声器。
设置音频路由为听筒时，声音比较小，只有将耳朵凑近才能听清楚，隐私性较好，适合用于接听电话。
设置音频路由为扬声器时，声音比较大，不用将手机贴脸也能听清，因此可以实现“免提”的功能。

setExposureCompensation:

setExposureCompensation:

```
- (NSInteger)setExposureCompensation:(CGFloat)value
```

设置摄像头的曝光参数，取值范围从-1到1

getDevicesList:

getDevicesList:

```
- (NSArray<TXMediaDeviceInfo *> * _Nullable)getDevicesList:(TXMediaDeviceType
```

获取设备列表（仅适用于桌面端）

参数	描述
type	设备类型，指定需要获取哪种设备的列表。详见 TXMediaDeviceType 定义。

注意

- 使用完毕后请调用 release 方法释放资源，这样可以让 SDK 维护 ITXDeviceCollection 对象的生命周期。
- 不要使用 delete 释放返回的 Collection 对象，delete ITXDeviceCollection* 指针会导致异常崩溃。
- type 只支持 TXMediaDeviceTypeMic、TXMediaDeviceTypeSpeaker、TXMediaDeviceTypeCamera。
- 此接口只支持 Mac 和 Windows 平台。

setCurrentDevice:deviceId:

setCurrentDevice:deviceId:

```
- (NSInteger)setCurrentDevice:(TXMediaDeviceType)type
                     deviceId:(NSString *)deviceId
```

设置当前要使用的设备（仅适用于桌面端）

参数	描述
deviceId	设备ID，您可以通过接口 getDevicesList 获得设备 ID。
type	设备类型，详见 TXMediaDeviceType 定义。

返回值说明：

0：操作成功；负数：操作失败。

getCurrentDevice:

getCurrentDevice:

```
- (TXMediaDeviceInfo * _Nullable)getCurrentDevice:(TXMediaDeviceType)type
```

获取当前正在使用的设备（仅适用于桌面端）

setCurrentDeviceVolume:deviceType:

setCurrentDeviceVolume:deviceType:

```
- (NSInteger)setCurrentDeviceVolume:(NSInteger)volume
                         deviceType:(TXMediaDeviceType)type
```

设置当前设备的音量（仅适用于桌面端）

这里的音量指的是麦克风的采集音量或者扬声器的播放音量，摄像头是不支持设置音量的。

参数	描述
volume	音量大小，取值范围为0 - 100，默认值：100。

getCurrentDeviceVolume:

getCurrentDeviceVolume:

```
- (NSInteger)getCurrentDeviceVolume:(TXMediaDeviceType)type
```

获取当前设备的音量（仅适用于桌面端）

这里的音量指的是麦克风的采集音量或者扬声器的播放音量，摄像头是不支持获取音量的。

setCurrentDeviceMute:deviceType:

setCurrentDeviceMute:deviceType:

```
- (NSInteger)setCurrentDeviceMute:(BOOL)mute
                       deviceType:(TXMediaDeviceType)type
```

设置当前设备的静音状态（仅适用于桌面端）

这里的音量指的是麦克风和扬声器，摄像头是不支持静音操作的。

getCurrentDeviceMute:

getCurrentDeviceMute:

```
- (BOOL)getCurrentDeviceMute:(TXMediaDeviceType)type
```

获取当前设备的静音状态（仅适用于桌面端）

这里的音量指的是麦克风和扬声器，摄像头是不支持静音操作的。

enableFollowingDefaultAudioDevice:enable:

enableFollowingDefaultAudioDevice:enable:

```
- (NSInteger)enableFollowingDefaultAudioDevice:(TXMediaDeviceType)type
                                        enable:(BOOL)enable
```

设置 SDK 使用的音频设备根据跟随系统默认设备（仅适用于桌面端）

仅支持设置麦克风和扬声器类型，摄像头暂不支持跟随系统默认设备

参数	描述
enable	是否跟随系统默认的音频设备。 • true：跟随。当系统默认音频设备发生改变或者有新音频设备插入时，SDK 立即切换音频设备。 • false：不跟随。当系统默认音频设备发生改变或者有新音频设备插入时，SDK 不会切换音频设备。
type	设备类型，详见 TXMediaDeviceType 定义。

startCameraDeviceTest:

startCameraDeviceTest:

```
- (NSInteger)startCameraDeviceTest:      (NSView *)view
```

开始摄像头测试（仅适用于桌面端）

> **注意**
> 在测试过程中可以使用 setCurrentDevice 接口切换摄像头。

stopCameraDeviceTest

stopCameraDeviceTest

结束摄像头测试（仅适用于桌面端）

startMicDeviceTest:

startMicDeviceTest:

```
- (NSInteger)startMicDeviceTest:      (NSInteger)interval
```

开始麦克风测试（仅适用于桌面端）

该接口可以测试麦克风是否能正常工作，测试到的麦克风采集音量的大小，会以回调的形式通知给您，其中 volume 的取值范围为0 - 100。

参数	描述
interval	麦克风音量的回调间隔。

注意
该接口调用后默认会回播麦克风录制到的声音到扬声器中。

startMicDeviceTest:playback:

startMicDeviceTest:playback:

```
- (NSInteger)startMicDeviceTest:    (NSInteger)interval
                       playback:    (BOOL)playback
```

开始麦克风测试（仅适用于桌面端）

该接口可以测试麦克风是否能正常工作，测试到的麦克风采集音量的大小，会以回调的形式通知给您，其中 volume 的取值范围为0 - 100。

参数	描述
interval	麦克风音量的回调间隔。
playback	是否开启回播麦克风声音，开启后用户测试麦克风时会听到自己的声音。

stopMicDeviceTest

stopMicDeviceTest

结束麦克风测试（仅适用于桌面端）

startSpeakerDeviceTest:

startSpeakerDeviceTest:

```
- (NSInteger)startSpeakerDeviceTest:(NSString *)audioFilePath
```

开始扬声器测试（仅适用于桌面端）

该接口通过播放指定的音频文件，用于测试播放设备是否能正常工作。如果用户在测试时能听到声音，说明播放设备能正常工作。

参数	描述
filePath	声音文件的路径

stopSpeakerDeviceTest

stopSpeakerDeviceTest

结束扬声器测试（仅适用于桌面端）

setObserver:

setObserver:

```
- (void)setObserver:(nullable id<TXDeviceObserver>)observe
```

设备热插拔回调（仅适用于 Mac 系统）

setSystemVolumeType:

setSystemVolumeType:

```
- (NSInteger)setSystemVolumeType:(TXSystemVolumeType)type
```

设置系统音量类型（仅适用于移动端）

@deprecated v9.5 版本开始不推荐使用，建议使用 TRTCCloud 中的 startLocalAudio(quality) 接口替代之，通过 quality 参数来决策音质。

TXSystemVolumeType(Deprecated)

TXSystemVolumeType(Deprecated)

系统音量类型（仅适用于移动设备）

@deprecated v9.5 版本开始不推荐使用。

现代智能手机中一般都具备两套系统音量类型，即“通话音量”和“媒体音量”。

- 通话音量：手机专门为接打电话所设计的音量类型，自带回声抵消（AEC）功能，并且支持通过蓝牙耳机上的麦克风进行拾音，缺点是音质比较一般。

当您通过手机侧面的音量按键下调手机音量时，如果无法将其调至零（也就是无法彻底静音），说明您的手机当前处于通话音量。

- 媒体音量：手机专门为音乐场景所设计的音量类型，无法使用系统的 AEC 功能，并且不支持通过蓝牙耳机的麦克风进行拾音，但具备更好的音乐播放效果。

当您通过手机侧面的音量按键下调手机音量时，如果能够将手机音量调至彻底静音，说明您的手机当前处于媒体音量。

SDK 目前提供了三种系统音量类型的控制模式：自动切换模式、全程通话音量模式、全程媒体音量模式。

枚举	取值	描述
TXSystemVolumeTypeAuto	0	自动切换模式
TXSystemVolumeTypeMedia	1	全程媒体音量
TXSystemVolumeTypeVOIP	2	全程通话音量

TXAudioRoute

TXAudioRoute

音频路由（即声音的播放模式）

音频路由，即声音是从手机的扬声器还是从听筒中播放出来，因此该接口仅适用于手机等移动端设备。
手机有两个扬声器：一个是位于手机顶部的听筒，一个是位于手机底部的立体声扬声器。

- 设置音频路由为听筒时，声音比较小，只有将耳朵凑近才能听清楚，隐私性较好，适合用于接听电话。
- 设置音频路由为扬声器时，声音比较大，不用将手机贴脸也能听清，因此可以实现“免提”的功能。

枚举	取值	描述
TXAudioRouteSpeakerphone	0	Speakerphone：使用扬声器播放（即“免提”），扬声器位于手机底部，声音偏大，适合外放音乐。
TXAudioRouteEarpiece	1	Earpiece：使用听筒播放，听筒位于手机顶部，声音偏小，适合需要保护隐私的通话场景。

TXMediaDeviceType

TXMediaDeviceType

设备类型（仅适用于桌面平台）

该枚举值用于定义三种类型的音视频设备，即摄像头、麦克风和扬声器，以便让一套设备管理接口可以操控三种不同类型的设备。

枚举	取值	描述
TXMediaDeviceTypeUnknown	-1	未定义的设备类型
TXMediaDeviceTypeAudioInput	0	麦克风类型设备
TXMediaDeviceTypeAudioOutput	1	扬声器类型设备
TXMediaDeviceTypeVideoCamera	2	摄像头类型设备

TXMediaDeviceState

TXMediaDeviceState

设备操作

该枚举值用于本地设备的状态变化通知 onDeviceChanged 。

枚举	取值	描述
TXMediaDeviceStateAdd	0	设备已被插入
TXMediaDeviceStateRemove	1	设备已被移除
TXMediaDeviceStateActive	2	设备已启用
TXMediaDefaultDeviceChanged	3	系统默认设备变更

TXMediaDeviceInfo

TXMediaDeviceInfo

音视频设备的相关信息（仅适用于桌面平台）

该结构体用于描述一个音视频设备的关键信息，比如设备 ID、设备名称等等，以便用户能够在用户界面上选择自己期望使用的音视频设备。

枚举类型	描述
deviceId	设备 ID（UTF-8）
deviceName	设备名称（UTF-8）
deviceProperties	设备属性
type	设备类型

错误码表

最近更新时间：2024-03-07 15:14:11

Module: V2TXLiveCode @ TXLiteAVSDK

Function: 腾讯云直播服务(LVB)错误码和警告码的定义。

错误码表

枚举类型

枚举类型	描述
V2TXLiveCode	V2 错误码和警告码

V2TXLiveCode

V2TXLiveCode

V2 错误码和警告码

枚举	取值	描述
V2TXLIVE_OK	0	没有错误。
V2TXLIVE_ERROR_FAILED	-1	暂未归类的通用错误。
V2TXLIVE_ERROR_INVALID_PARAMETER	-2	调用 API 时，传入的参数不合法。
V2TXLIVE_ERROR_REFUSED	-3	API 调用被拒绝。
V2TXLIVE_ERROR_NOT_SUPPORTED	-4	当前 API 不支持调用。
V2TXLIVE_ERROR_INVALID_LICENSE	-5	license 不合法，调用失败。

V2TXLIVE_ERROR_REQUEST_TIMEOUT	-6	请求服务器超时。
V2TXLIVE_ERROR_SERVER_PROCESS_FAILED	-7	服务器无法处理您的请求。
V2TXLIVE_ERROR_DISCONNECTED	-8	连接断开。
V2TXLIVE_ERROR_NO_AVAILABLE_HEVC_DECODERS	-2304	找不到可用的 HEVC 解码器。
V2TXLIVE_WARNING_NETWORK_BUSY	1101	网络状况不佳：上行带宽太小，上传数据受阻。
V2TXLIVE_WARNING_VIDEO_BLOCK	2105	当前视频播放出现卡顿。
V2TXLIVE_WARNING_CAMERA_START_FAILED	-1301	摄像头打开失败。
V2TXLIVE_WARNING_CAMERA_OCCUPIED	-1316	摄像头正在被占用中，可尝试打开其他摄像头。
V2TXLIVE_WARNING_CAMERA_NO_PERMISSION	-1314	摄像头设备未授权，通常在移动设备出现，可能是权限被用户拒绝了。
V2TXLIVE_WARNING_MICROPHONE_START_FAILED	-1302	麦克风打开失败。
V2TXLIVE_WARNING_MICROPHONE_OCCUPIED	-1319	麦克风正在被占用中，例如移动设备正在通话时，打开麦克风会失败。
V2TXLIVE_WARNING_MICROPHONE_NO_PERMISSION	-1317	麦克风设备未授权，通常在移动设备出现，可能是权限被用户拒绝了。
V2TXLIVE_WARNING_SCREEN_CAPTURE_NOT_SUPPORTED	-1309	当前系统不支持屏幕分享。
V2TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	-1308	开始录屏失败，如果在移动设备出现，可能是权限被用户拒绝了。
V2TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	-7001	录屏被系统中断。
V2TXLIVE_WARNING_CURRENT_ENCODE_TYPE_CHANGED	1104	表示编码器发生改变，可以通过 onWarning 函数的扩展信息中的 codec_type 字段来获取当前的编码格式。

		其中 1 代表 265 编码，0 代表 264 编码。注意 Windows 端不支持此错误码的扩展信息。
V2TXLIVE_WARNING_CURRENT_DECODE_TYPE_CHANGED	2008	表示解码器发生改变，可以通过 onWarning 函数的扩展信息中的 codec_type 字段来获取当前的解码格式。 其中 1 代表 265 解码，0 代表 264 解码。注意 Windows 端不支持此错误码的扩展信息。

类型定义

最近更新时间：2024-01-16 14:35:21

Module: V2TXLiveDef @ TXLiteAVSDK

Function: 腾讯云直播服务(LVB)关键类型定义

类型定义

结构体类型

函数列表	描述
V2TXLiveVideoEncoderParam	视频编码参数
V2TXLiveVideoFrame	视频帧信息
V2TXLiveAudioFrame	音频帧数据
V2TXLiveAudioFrameObserverFormat	音频帧回调格式
V2TXLivePusherStatistics	推流器的统计数据
V2TXLivePlayerStatistics	播放器的统计数据
V2TXLiveMixStream	云端混流中每一路子画面的位置信息
V2TXLiveTranscodingConfig	云端混流（转码）配置
V2TXLiveLocalRecordingParams	本地录制音视频配置
V2TXLiveSocks5ProxyConfig	socks5 代理的协议配置
V2TXLiveLogConfig	Log配置
V2TXLiveStreamInfo	支持自适应切换的码流信息

枚举类型

枚举类型	描述
V2TXLiveMode	支持协议
V2TXLiveVideoResolution	视频分辨率
V2TXLiveVideoResolutionMode	视频宽高比模式
V2TXLiveMirrorType	本地摄像头镜像类型
V2TXLiveFillMode	视频画面填充模式
V2TXLiveRotation	视频画面顺时针旋转角度
V2TXLivePixelFormat	视频帧的像素格式
V2TXLiveBufferType	视频数据包装格式
V2TXLivePictureInPictureState	画中画的状态
V2TXLiveAudioQuality	声音音质
V2TXLiveAudioFrameOperationMode	音频回调数据读写模式
V2TXLivePushStatus	直播流的连接状态
V2TXAudioRoute	声音播放模式（音频路由）
V2TXLiveMixInputType	混流输入类型配置
V2TXLiveRecordMode	本地音视频录制模式
V2TXLiveLogLevel	日志级别枚举值

V2TXLiveMode

V2TXLiveMode

支持协议

枚举	取值	描述
V2TXLiveMode_RTMP	Not Defined	支持协议: RTMP。
V2TXLiveMode_RTC	Not Defined	支持协议: TRTC。

V2TXLiveVideoResolution

V2TXLiveVideoResolution

视频分辨率

枚举	取值	描述
V2TXLiveVideoResolution160x160	Not Defined	分辨率 160*160，码率范围：100Kbps ~ 150Kbps，帧率：15fps。
V2TXLiveVideoResolution270x270	Not Defined	分辨率 270*270，码率范围：200Kbps ~ 300Kbps，帧率：15fps。
V2TXLiveVideoResolution480x480	Not Defined	分辨率 480*480，码率范围：350Kbps ~ 525Kbps，帧率：15fps。
V2TXLiveVideoResolution320x240	Not Defined	分辨率 320*240，码率范围：250Kbps ~ 375Kbps，帧率：15fps。
V2TXLiveVideoResolution480x360	Not Defined	分辨率 480*360，码率范围：400Kbps ~ 600Kbps，帧率：15fps。
V2TXLiveVideoResolution640x480	Not Defi	分辨率 640*480，码率范围：600Kbps ~ 900Kbps，帧率：15fps。

	ned	
V2TXLiveVideoResolution320x180	Not Defined	分辨率 320*180，码率范围：250Kbps ~ 400Kbps，帧率：15fps。
V2TXLiveVideoResolution480x270	Not Defined	分辨率 480*270，码率范围：350Kbps ~ 550Kbps，帧率：15fps。
V2TXLiveVideoResolution640x360	Not Defined	分辨率 640*360，码率范围：500Kbps ~ 900Kbps，帧率：15fps。
V2TXLiveVideoResolution960x540	Not Defined	分辨率 960*540，码率范围：800Kbps ~ 1500Kbps，帧率：15fps。
V2TXLiveVideoResolution1280x720	Not Defined	分辨率 1280*720，码率范围：1000Kbps ~ 1800Kbps，帧率：15fps。
V2TXLiveVideoResolution1920x1080	Not Defined	分辨率 1920*1080，码率范围：2500Kbps ~ 3000Kbps，帧率：15fps。

V2TXLiveVideoResolutionMode

V2TXLiveVideoResolutionMode

视频宽高比模式

> **注意**
> - 横屏模式下的分辨率: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModeLandscape = 640 × 360。
> - 竖屏模式下的分辨率: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModePortrait = 360 × 640。

枚举	取值	描述
V2TXLiveVideoResolutionModeLandscape	0	横屏模式。

V2TXLiveVideoResolutionModePortrait	1	竖屏模式。

V2TXLiveMirrorType

V2TXLiveMirrorType

本地摄像头镜像类型

枚举	取值	描述
V2TXLiveMirrorTypeAuto	NotDefined	系统默认镜像类型，前置摄像头镜像，后置摄像头不镜像。
V2TXLiveMirrorTypeEnable	NotDefined	前置摄像头和后置摄像头，都切换为镜像模式。
V2TXLiveMirrorTypeDisable	NotDefined	前置摄像头和后置摄像头，都切换为非镜像模式。

V2TXLiveFillMode

V2TXLiveFillMode

视频画面填充模式

枚举	取值	描述
V2TXLiveFillModeFill	NotDefined	图像铺满屏幕，超出显示视窗的视频部分将被裁剪，画面显示可能不完整。
V2TXLiveFillModeFit	NotDefined	图像长边填满屏幕，短边区域会被填充黑色，画面的内容完整。

V2TXLiveFillModeScaleFill	Not Defined	图像拉伸铺满，因此长度和宽度可能不会按比例变化。

V2TXLiveRotation

V2TXLiveRotation

视频画面顺时针旋转角度

枚举	取值	描述
V2TXLiveRotation0	Not Defined	不旋转。
V2TXLiveRotation90	Not Defined	顺时针旋转90度。
V2TXLiveRotation180	Not Defined	顺时针旋转180度。
V2TXLiveRotation270	Not Defined	顺时针旋转270度。

V2TXLivePixelFormat

V2TXLivePixelFormat

视频帧的像素格式

枚举	取值	描述
V2TXLivePixelFormatUnknown	Not Defined	未知。

V2TXLivePixelFormatI420	Not Defined	YUV420P I420。
V2TXLivePixelFormatNV12	Not Defined	YUV420SP NV12。
V2TXLivePixelFormatBGRA32	Not Defined	BGRA8888。
V2TXLivePixelFormatTexture2D	Not Defined	OpenGL 2D 纹理。

V2TXLiveBufferType

V2TXLiveBufferType

视频数据包装格式

> **注意**
> 在自定义采集和自定义渲染功能，您需要用到下列枚举值来指定您希望以什么样的格式来包装视频数据。
> - PixelBuffer：直接使用效率最高，iOS 系统提供了众多 API 获取或处理 PixelBuffer。
> - NSData： 当使用自定义渲染时，PixelBuffer拷贝一次到NSData。当使用自定义采集时，NSData拷贝一次到PixelBuffer。因此，性能会受到一定程度的影响。

枚举	取值	描述
V2TXLiveBufferTypeUnknown	Not Defined	未知。
V2TXLiveBufferTypePixelBuffer	Not Defined	直接使用效率最高，iOS 系统提供了众多 API 获取或处理 PixelBuffer。
V2TXLiveBufferTypeNSData	Not Defi	会有一定的性能消耗，SDK 内部是直接处理 PixelBuffer 的，所以会存在 NSData 和

	ned	PixelBuffer 之间类型转换所产生的内存拷贝开销。
V2TXLiveBufferTypeTexture	Not Defined	直接操作纹理 ID，性能最好。

V2TXLivePictureInPictureState

V2TXLivePictureInPictureState

画中画的状态

枚举	取值	描述
V2TXLivePictureInPictureStateUndefined	Not Defined	未定义。
V2TXLivePictureInPictureStateOccurError	Not Defined	画中画发生错误。
V2TXLivePictureInPictureStateWillStart	Not Defined	画中画将要开始。
V2TXLivePictureInPictureStateDidStart	Not Defined	画中画已经开始。
V2TXLivePictureInPictureStateWillStop	Not Defined	画中画将要停止。
V2TXLivePictureInPictureStateDidStop	Not Defined	画中画已经停止。

V2TXLiveAudioQuality

V2TXLiveAudioQuality

声音音质

枚举	取值	描述
V2TXLiveAudioQualitySpeech	Not Defined	语音音质：采样率：16k；单声道；音频码率：16kbps；适合语音通话为主的场景，比如在线会议，语音通话。
V2TXLiveAudioQualityDefault	Not Defined	默认音质：采样率：48k；单声道；音频码率：50kbps；SDK 默认的音频质量，如无特殊需求推荐选择之。
V2TXLiveAudioQualityMusic	Not Defined	音乐音质：采样率：48k；双声道 + 全频带；音频码率：128kbps；适合需要高保真传输音乐的场景，比如K歌、音乐直播等。

V2TXLiveAudioFrameOperationMode

V2TXLiveAudioFrameOperationMode

音频回调数据读写模式

SDK 提供了两种音频回调数据的操作模式。

- 读写模式（ReadWrite）：可以获取并修改回调的音频数据，默认模式。
- 只读模式（ReadOnly）：仅从回调中获取音频数据。

枚举	取值	描述
V2TXLiveAudioFrameOperationModeReadWrite	0	读写模式：可以获取并修改回调的音频数据。
V2TXLiveAudioFrameOperationModeReadOnly	1	只读模式：仅从回调中获取音频数据。

V2TXLivePushStatus

V2TXLivePushStatus

直播流的连接状态

枚举	取值	描述
V2TXLivePushStatusDisconnected	Not Defined	与服务器断开连接。
V2TXLivePushStatusConnecting	Not Defined	正在连接服务器。
V2TXLivePushStatusConnectSuccess	Not Defined	连接服务器成功。
V2TXLivePushStatusReconnecting	Not Defined	重连服务器中。

V2TXAudioRoute

V2TXAudioRoute

声音播放模式（音频路由）

枚举	取值	描述
V2TXAudioModeSpeakerphone	Not Defined	扬声器。
V2TXAudioModeEarpiece	Not Defined	听筒。

V2TXLiveMixInputType

V2TXLiveMixInputType

混流输入类型配置

枚举	取值	描述
V2TXLiveMixInputTypeAudioVideo	Not Defined	混入音视频。
V2TXLiveMixInputTypePureVideo	Not Defined	只混入视频。
V2TXLiveMixInputTypePureAudio	Not Defined	只混入音频。

V2TXLiveRecordMode

V2TXLiveRecordMode

本地音视频录制模式

枚举	取值	描述
V2TXLiveRecordModeBoth	Not Defined	Both mode: 录制音频和视频

V2TXLiveLogLevel

V2TXLiveLogLevel

日志级别枚举值

枚举	取值	描述
V2TXLiveLogLevelAll	0	输出所有级别的 log。
V2TXLiveLogLevelDe	1	输出 DEBUG，INFO，WARNING，ERROR 和 FATAL

bug		级别的 log。
V2TXLiveLogLevelInfo	2	输出 INFO，WARNING，ERROR 和 FATAL 级别的 log。
V2TXLiveLogLevelWarning	3	只输出 WARNING，ERROR 和 FATAL 级别的 log。
V2TXLiveLogLevelError	4	只输出 ERROR 和 FATAL 级别的 log。
V2TXLiveLogLevelFatal	5	只输出 FATAL 级别的 log。
V2TXLiveLogLevelNULL	6	不输出任何 sdk log。

V2TXLiveVideoEncoderParam

V2TXLiveVideoEncoderParam

视频编码参数

该设置决定远端用户看到的画面质量。

枚举类型	描述
minVideoBitrate	【字段含义】最低视频码率，SDK 会在网络不佳的情况下主动降低视频码率以保持流畅度，最低会降至 minVideoBitrate 所设定的数值。 【推荐取值】您可以通过同时设置 videoBitrate 和 minVideoBitrate 两个参数，用于约束 SDK 对视频码率的调整范围： • 如果您将 videoBitrate 和 minVideoBitrate 设置为同一个值，等价于关闭 SDK 对视频码率的自适应调节能力。
videoBitrate	【字段含义】目标视频码率，SDK 会按照目标码率进行编码，只有在弱网络环境下才会主动降低视频码率。 【推荐取值】请参考 V2TXLiveVideoResolution 在各档位注释的最佳码率，也可以在此基础上适当调高。 比如：V2TXLiveVideoResolution1280x720 对应 1200kbps 的目标码率，您也可以设置为 1500kbps 用来获得更好的观感清晰度。 【特别说明】您可以通过同时设置 videoBitrate 和 minVideoBitrate 两个参数，用于约束 SDK 对视频码率的调整范围： • 如果您将 videoBitrate 和 minVideoBitrate 设置为同一个值，等价于关闭 SDK 对视频码率的自适应调节能力。

videoFps	【字段含义】视频采集帧率。 【推荐取值】15fps 或 20fps。5fps 以下，卡顿感明显。10fps 以下，会有轻微卡顿感。20fps 以上，会浪费带宽（电影的帧率为 24fps）。
videoResolution	【字段含义】视频分辨率。 【特别说明】如需使用竖屏分辨率，请指定 videoResolutionMode 为 Portrait，例如： 640 × 360 + Portrait = 360 × 640。 【推荐取值】 • 桌面平台（Win + Mac）：建议选择 640 × 360 及以上分辨率，videoResolutionMode 选择 Landscape，即横屏分辨率。
videoResolutionMode	【字段含义】分辨率模式（横屏分辨率 or 竖屏分辨率）。 【推荐取值】桌面平台（Windows、Mac）建议选择 Landscape。 【特别说明】如需使用竖屏分辨率，请指定 resMode 为 Portrait，例如：640 × 360 + Portrait = 360 × 640。

V2TXLiveVideoFrame

V2TXLiveVideoFrame

视频帧信息

> **注意**
> 自定义采集和自定义渲染时使用。自定义采集时，需要使用 V2TXLiveVideoFrame 来包装待发送的视频帧；自定义渲染时，会返回经过 V2TXLiveVideoFrame 包装的视频帧。

枚举类型	描述
bufferType	【字段含义】视频数据包装格式。 【推荐取值】V2TXLiveBufferTypePixelBuffer。
data	【字段含义】bufferType 为 V2TXLiveBufferTypeNSData 时的视频数据。
height	【字段含义】视频高度。
pixelBuffer	【字段含义】bufferType 为 V2TXLiveBufferTypePixelBuffer 时的视频数据。
pixelFormat	【字段含义】视频帧像素格式。 【推荐取值】V2TXLivePixelFormatNV12。

rotation	【字段含义】视频帧的顺时针旋转角度。
textureId	【字段含义】视频纹理ID。
width	【字段含义】视频宽度。

V2TXLiveAudioFrame

V2TXLiveAudioFrame

音频帧数据

枚举类型	描述
channel	【字段含义】声道数。
data	【字段含义】音频数据。
sampleRate	【字段含义】采样率。
timestamp	【字段含义】时间戳，单位ms。

V2TXLiveAudioFrameObserverFormat

V2TXLiveAudioFrameObserverFormat

音频帧回调格式

枚举类型	描述
channel	【字段含义】声道数。 【推荐取值】默认值：1，代表单声道。可设定的数值只有两个数字：1-单声道，2-双声道。
mode	【字段含义】回调数据读写模式。 【推荐取值】V2TXLiveAudioFrameOperationModeReadOnly：仅从回调中获取音频数据。可设定的模式有V2TXLiveAudioFrameOperationModeReadOnly，V2TXLiveAudioFrameOperationModeReadWrite。

sampleRate	【字段含义】采样率。 【推荐取值】默认值：48000Hz。支持 16000, 32000, 44100, 48000。
samplesPerCall	【字段含义】采样点数。 【推荐取值】取值必须是 sampleRate/100 的整数倍。

V2TXLivePusherStatistics

V2TXLivePusherStatistics

推流器的统计数据

枚举类型	描述
appCpu	【字段含义】当前 App 的 CPU 使用率（%）。
audioBitrate	【字段含义】音频码率（Kbps）。
fps	【字段含义】帧率（fps）。
height	【字段含义】视频高度。
netSpeed	【字段含义】上行速度（kbps）
rtt	【字段含义】从 SDK 到云端的往返延时（ms）
systemCpu	【字段含义】当前系统的 CPU 使用率（%）。
videoBitrate	【字段含义】视频码率（Kbps）。
width	【字段含义】视频宽度。

V2TXLivePlayerStatistics

V2TXLivePlayerStatistics

播放器的统计数据

枚举类型	描述
appCpu	【字段含义】当前 App 的 CPU 使用率（%）。

audioBitrate	【字段含义】音频码率（Kbps）。
audioBlockRate	【字段含义】音频播放卡顿率，单位（%）。 音频播放卡顿率（audioBlockRate）= 音频播放的累计卡顿时长（audioTotalBlockTime）/ 音频播放的区间时长（2000ms）。
audioPacketLoss	【字段含义】网络音频丢包率（%），注：仅支持前缀为 [trtc://] 或 [webrtc://] 的播放地址。
audioTotalBlockTime	【字段含义】音频播放的累计卡顿时长（ms）。 该时长为区间（2s）内的卡顿时长。
fps	【字段含义】帧率（fps）。
height	【字段含义】视频高度。
jitterBufferDelay	【字段含义】播放延迟（ms）。
netSpeed	【字段含义】下载速度（kbps）
rtt	【字段含义】从 SDK 到云端的往返延时（ms），注：仅支持前缀为 [trtc://] 或 [webrtc://] 的播放地址。
systemCpu	【字段含义】当前系统的 CPU 使用率（%）。
videoBitrate	【字段含义】视频码率（Kbps）。
videoBlockRate	【字段含义】视频播放卡顿率，单位（%）。 视频播放卡顿率（videoBlockRate）= 视频播放的累计卡顿时长（videoTotalBlockTime）/ 视频播放的区间时长（2000ms）。
videoPacketLoss	【字段含义】网络视频丢包率（%），注：仅支持前缀为 [trtc://] 或 [webrtc://] 的播放地址。
videoTotalBlockTime	【字段含义】视频播放的累计卡顿时长（ms）。 该时长为区间（2s）内的卡顿时长。
width	【字段含义】视频宽度。

V2TXLiveMixStream

V2TXLiveMixStream

云端混流中每一路子画面的位置信息

枚举类型	描述
height	【字段含义】图层位置高度（绝对像素值）。
inputType	【字段含义】该直播流的输入类型。
streamId	【字段含义】参与混流的 userId 所在对应的推流 streamId，nil 表示当前推流 streamId。
userId	【字段含义】参与混流的 userId。
width	【字段含义】图层位置宽度（绝对像素值）。
x	【字段含义】图层位置 x 坐标（绝对像素值）。
y	【字段含义】图层位置 y 坐标（绝对像素值）。
zOrder	【字段含义】图层层次（1 – 15）不可重复。

V2TXLiveTranscodingConfig

V2TXLiveTranscodingConfig

云端混流（转码）配置

枚举类型	描述
audioBitrate	【字段含义】最终转码后的音频码率。 【推荐取值】默认值：64kbps，取值范围是 [32，192]，单位：kbps。
audioChannels	【字段含义】最终转码后的音频声道数。 【推荐取值】默认值：1。取值范围为 [1,2] 中的整型。
audioSampleRate	【字段含义】最终转码后的音频采样率。 【推荐取值】默认值：48000Hz。支持12000HZ、16000HZ、22050HZ、24000HZ、32000HZ、44100HZ、48000HZ。
backgroundColor	【字段含义】混合后画面的底色颜色，默认为黑色，格式为十六进制数字，比如：“0x61B9F1” 代表 RGB 分别为(97,158,241)。 【推荐取值】默认值：0x000000，黑色。
backgroundImage	【字段含义】混合后画面的背景图。 【推荐取值】默认值：nil，即不设置背景图。 【特别说明】背景图需要您事先在 “控制台 => 应用管理 => 功能配置 => 素材管理” 中上传，

	上传成功后可以获得对应的“图片ID”，然后将“图片ID”转换成字符串类型并设置到 backgroundImage 里即可。 例如：假设“图片ID” 为 63，可以设置 backgroundImage = "63"。
mixStreams	【字段含义】每一路子画面的位置信息。
outputStreamId	【字段含义】输出到 CDN 上的直播流 ID。 如不设置该参数，SDK 会执行默认逻辑，即房间里的多路流会混合到该接口调用者的视频流上，也就是 A + B => A。 如果设置该参数，SDK 会将房间里的多路流混合到您指定的直播流 ID 上，也就是 A + B => C。 【推荐取值】默认值：nil，即房间里的多路流会混合到该接口调用者的视频流上。
videoBitrate	【字段含义】最终转码后的视频分辨率的码率（kbps）。 【推荐取值】如果填0，后台会根据 videoWidth 和 videoHeight 来估算码率，您也可以参考枚举定义 V2TXLiveVideoResolution 的注释。
videoFramerate	【字段含义】最终转码后的视频分辨率的帧率（FPS）。 【推荐取值】默认值：15fps，取值范围是 (0,30]。
videoGOP	【字段含义】最终转码后的视频分辨率的关键帧间隔（又称为 GOP）。 【推荐取值】默认值：2，单位为秒，取值范围是 [1,8]。
videoHeight	【字段含义】最终转码后的视频分辨率的高度。 【推荐取值】推荐值：640px，如果你是纯音频推流，请将 width × height 设为 0px × 0px，否则混流后会携带一条画布背景的视频流。
videoWidth	【字段含义】最终转码后的视频分辨率的宽度。 【推荐取值】推荐值：360px，如果你是纯音频推流，请将 width × height 设为 0px × 0px，否则混流后会携带一条画布背景的视频流。

V2TXLiveLocalRecordingParams

V2TXLiveLocalRecordingParams

本地录制音视频配置

枚举类型	描述
filePath	【字段含义】录制的文件地址（必填），请确保路径有读写权限且合法，否则录制文件无法生成。 【推荐取值】该路径需精确到文件名及格式后缀，格式后缀用于决定录制出的文件格式，目前支持的格式暂时只有 MP4。

interval	【字段含义】interval 录制信息更新频率，单位毫秒，有效范围：1000-10000。 【推荐取值】`-1`，表示不回调。
recordMode	【字段含义】媒体录制模式。 【推荐取值】`V2TXLiveRecordModeBoth`，即同时录制音频和视频。

V2TXLiveSocks5ProxyConfig

V2TXLiveSocks5ProxyConfig

socks5 代理的协议配置

枚举类型	描述
supportHttps	【字段含义】是否支持 https。 【推荐取值】默认值：true。
supportTcp	【字段含义】是否支持 tcp。 【推荐取值】默认值：true。
supportUdp	【字段含义】是否支持 udp。 【推荐取值】默认值：true。

V2TXLiveLogConfig

V2TXLiveLogConfig

Log配置

枚举类型	描述
enableConsole	【字段含义】是否允许 SDK 在编辑器（XCoder、Android Studio、Visual Studio 等）的控制台上打印 Log。 【推荐取值】默认值：NO。
enableLogFile	【字段含义】是否启用本地 Log 文件。 【特殊说明】如非特殊需要，请不要关闭本地 Log 文件，否则腾讯云技术团队将无法在出现问题时进行跟踪和定位。 【推荐取值】默认值：YES。

enableObserver	【字段含义】是否通过 V2TXLivePremierObserver 接收要打印的 Log 信息。 【特殊说明】如果您希望自己实现 Log 写入，可以打开此开关，Log 信息会通过 V2TXLivePremierObserver#onLog 回调给您。 【推荐取值】默认值：NO。
logLevel	【字段含义】设置 Log 级别。 【推荐取值】默认值：V2TXLiveLogLevelAll。
logPath	【字段含义】设置本地 Log 的存储目录，默认 Log 存储位置： iOS & Mac: sandbox Documents/log。

V2TXLiveStreamInfo

V2TXLiveStreamInfo

支持自适应切换的码流信息

枚举类型	描述
height	**字段含义** 视频高, 默认值：0, 表示未知。
url	**字段含义** 流地址，通过 SwitchStream 接口调用实现多码率质量切换。
width	**字段含义** 视频宽, 默认值：0, 表示未知。

连麦

MLVBLiveRoom

最近更新时间：2023-03-17 14:27:57

功能

腾讯云视立方 · 直播 SDK – 连麦直播间。

> **注意**
> 后台接口限制并发为每秒100次请求，若您有高并发请求请提前 联系我们 处理，避免影响服务调用。

介绍

基于腾讯云直播、点播（VOD）和即时通信（IM）三大 PAAS 服务组合而成，支持：

- 主播创建新的直播间开播，观众进入直播间观看。
- 主播和观众进行视频连麦互动。
- 两个不同房间的主播 PK 互动。
- 每一个直播间都有一个不限制房间人数的聊天室，支持发送各种文本消息和自定义消息，自定义消息可用于实现弹幕、点赞和礼物。

连麦直播间（MLVBLiveRoom）是一个开源的 Class，依赖两个腾讯云的闭源 SDK：

- LiteAVSDK：使用了其中的 TXLivePusher 和 TXLivePlayer 两个组件，前者用于推流，后者用于拉流。
- IM SDK：使用 IM SDK 的 AVChatroom 用于实现直播聊天室的功能，同时，主播间的连麦流程也是依靠 IM 消息串联起来的。

请参见 直播连麦（LiveRoom）。

SDK 基础函数

delegate

MLVBLiveRoom 事件回调，您可以通过 MLVBLiveRoomDelegate 获得 MLVBLiveRoom 的各种状态通知。

```
@property (nonatomic, weak) id< MLVBLiveRoomDelegate > delegate
```

> **说明**
> 默认是在 Main Queue 中回调，如果需要自定义回调线程，可使用 delegateQueue。

delegateQueue

设置驱动回调函数的 GCD 队列。

```
@property (nonatomic, copy) dispatch_queue_t delegateQueue
```

sharedInstance

获取 MLVBLiveRoom 单例对象。

```
+ (instancetype)sharedInstance
```

返回

MLVBLiveRoom 实例。

> **说明**
> 可以调用 MLVBLiveRoom destroySharedInstance 销毁单例对象。

destorySharedInstance

销毁 MLVBLiveRoom 单例对象。

```
+ (void)destorySharedInstance
```

> **说明**
> 销毁实例后，外部缓存的 MLVBLiveRoom 实例不能再使用，需要重新调用 sharedInstance 获取新实例。

loginWithInfo

登录。

```
- (void)loginWithInfo:(MLVBLoginInfo *)loginInfo completion:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
loginInfo	MLVBLoginInfo *	登录信息。
completion	void(^)(int errCode, NSString *errMsg)	登录结果回调。

logout

登出。

```
- (void)logout
```

setSelfProfile

修改个人信息。

```
- (void)setSelfProfile:(NSString *)userName avatarURL:(NSString *)avatarURL
completion:(void(^)(int code, NSString *msg))completion
```

参数

参数	类型	含义
userName	NSString *	昵称。
avatarURL	NSString *	头像地址。
completion	(void(^)(int code, NSString *msg))	修改结果回调。

房间相关接口函数

getRoomList

获取房间列表。

```
- (void)getRoomList:(int)index count:(int)count completion:(void(^)(int errCode,
NSString *errMsg, NSArray< MLVBRoomInfo * > *roomInfoArray))completion
```

参数

参数	类型	含义
index	int	房间开始索引，从0开始计算。
count	int	希望后台返回的房间个数。
completion	void(^)(int errCode, NSString *errMsg, NSArray< MLVBRoomInfo * > *roomInfoArray)	获取房间列表的结果回调。

介绍

该接口支持分页获取房间列表，可以用 index 和 count 两个参数控制列表分页的逻辑，

- index = 0 & count = 10代表获取第一页的10个房间。
- index = 11 & count = 10代表获取第二页的10个房间。

getAudienceList

获取观众列表。

```
- (void)getAudienceList:(NSString *)roomID completion:(void(^)(int errCode, NSString *errMsg, NSArray< MLVBAudienceInfo * > *audienceInfoArray))completion
```

参数

参数	类型	含义
roomID	NSString *	房间标识。
completion	void(^)(int errCode, NSString *errMsg, NSArray< MLVBAudienceInfo * > *audienceInfoArray)	获取观众列表的结果回调。

介绍

当有观众进房时，后台会将其信息加入到指定房间的观众列表中，调入该函数即可返回指定房间的观众列表。

> **说明**
> 观众列表最多只保存30人，因为对于常规的 UI 展示来说这已经足够，保存更多除了浪费存储空间，也会拖慢列表返回的速度。

createRoom

创建房间（主播调用）。

```
- (void)createRoom:(NSString *)roomID roomInfo:(NSString *)roomInfo completion:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
roomID	NSString *	房间标识，推荐做法是用主播的 userID 作为房间的 roomID，这样省去了后台映射的成本。room ID 可以填空，此时由后台生成。

roomInfo	NSString *	房间信息（非必填），用于房间描述的信息，如房间名称，允许使用 JSON 格式作为房间信息。
completion	void(^)(int errCode, NSString *errMsg)	创建房间的结果回调。

介绍

主播开播的正常调用流程是：

1. 主播调用 startLocalPreview 打开摄像头预览，此时可以调整美颜参数。
2. 主播调用 createRoom 创建直播间，房间创建成功与否会通过 completion 通知主播。

enterRoom

进入房间（观众调用）。

```
- (void)enterRoom:(NSString *)roomID view:(UIView *)view completion:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
roomID	NSString *	房间标识。
view	UIView *	承载视频画面的控件。
completion	void(^)(int errCode, NSString *errMsg)	进入房间的结果回调。

介绍

观众观看直播的正常调用流程是：

1. 观众调用 getRoomList 刷新最新的直播房间列表，并通过 completion 回调拿到房间列表。
2. 观众选择一个直播间以后，调用 enterRoom 进入该房间。

exitRoom

离开房间。

```
- (void)exitRoom:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
completion	void(^)(int errCode, NSString *errMsg)	离开房间的结果回调。

setCustomInfo

设置当前房间的扩展信息字段。

```
- (void)setCustomInfo:(MLVBCustomFieldOp)op key:(NSString *)key value:(id)value
completion:(void(^)(int errCode, NSString *custom))completion
```

参数

参数	类型	含义
op	MLVBCustomFieldOp	执行动作。
key	NSString *	自定义键。
value	id	可选类型为 NSNumber 或者 NSString。
completion	void(^)(int errCode, NSString *custom)	操作完成的回调。

介绍

有时候您需要为当前房间设置一些扩展字段，如“点赞人数”和“是否正在连麦”等，这些字段我们很难全都预先定义好，所以提供了如下三种操作接口：

- SET：设置，value 可以是数值或者字符串，例如“是否正在连麦”等。
- INC：增加，value 只能是整数，如“点赞人数”，“人气指数”等，都可以使用该操作接口。
- DEC：减少，value 只能是整数，如“点赞人数”，“人气指数”等，都可以使用该操作接口。

> **说明**
> op 为 MLVBCustomFieldOpSet 或者 MLVBCustomFieldOpDec 时，value 需要是一个数字。

getCustomInfo

获取当前房间的扩展信息字段。

```
- (void)getCustomInfo:(void(^)(int errCode, NSString *errMsg, NSDictionary
*customInfo))completion
```

参数

参数	类型	含义

completion	void(^)(int errCode, NSString *errMsg, NSDictionary *customInfo)	获取自定义值回调。

主播和观众连麦

requestJoinAnchor

观众请求连麦。

```
- (void)requestJoinAnchor:(NSString *)reason completion:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
reason	NSString *	连麦原因。
completion	void(^)(int errCode, NSString *errMsg)	主播响应回调。

介绍

主播和观众的连麦流程可以简单描述为如下几个步骤：

1. 观众调用 requestJoinAnchor 向主播发起连麦请求。
2. 主播会收到 MLVBLiveRoomDelegate.onRequestJoinAnchor 的回调通知。
3. 主播调用 responseJoinAnchor 确定是否接受观众的连麦请求。
4. 观众会收到 requestJoinAnchor 传入的回调通知，可以得知请求是否被同意。
5. 观众如果请求被同意，则调用 startLocalPreview 开启本地摄像头，如果 App 还没有取得摄像头和麦克风权限，会触发 UI 提示。
6. 观众然后调用 joinAnchor 正式进入连麦状态。
7. 主播一旦观众进入连麦状态，主播就会收到 MLVBLiveRoomDelegate onAnchorEnter 通知。
8. 主播调用 startRemoteView 就可以看到连麦观众的视频画面。
9. 观众如果直播间里已经有其他观众正在跟主播进行连麦，那么新加入的这位连麦观众也会收到 MLVBLiveRoomDelegate onAnchorJoin 通知，用于展示（startRemoteView）其他连麦者的视频画面。

responseJoinAnchor

主播处理连麦请求。

```
- (void)responseJoinAnchor:(NSString *)userID agree:(BOOL)agree reason:(NSString *)reason
```

参数

参数	类型	含义
userID	NSString *	观众 ID。
agree	BOOL	YES：同意；NO：拒绝。
reason	NSString *	同意/拒绝连麦的原因描述。

介绍

主播在收到 MLVBLiveRoomDelegate.onRequestJoinAnchor 回调之后会需要调用此接口来处理观众的连麦请求。

joinAnchor

进入连麦状态。

```
- (void)joinAnchor:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
completion	void(^)(int errCode, NSString *errMsg)	进入连麦的结果回调。

介绍

进入连麦成功后，主播和其他连麦观众会收到 MLVBLiveRoomDelegate.onAnchorEnter 通知。

quitJoinAnchor

观众退出连麦。

```
- (void)quitJoinAnchor:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
completion	void(^)(int errCode, NSString *errMsg)	退出连麦的结果回调。

介绍

退出连麦成功后，主播和其他连麦观众会收到 MLVBLiveRoomDelegate.onAnchorExit 通知。

kickoutJoinAnchor

主播踢除连麦观众。

```
- (void)kickoutJoinAnchor:(NSString *)userID
```

参数

参数	类型	含义
userID	NSString *	连麦小主播 ID。

介绍

主播调用此接口踢除连麦观众后，被踢连麦观众会收到 MLVBLiveRoomDelegate.onKickoutJoinAnchor 回调通知。

主播跨房间 PK

requestRoomPK

请求跨房 PK。

```
- (void)requestRoomPK:(NSString *)userID completion:(void(^)(int errCode, NSString *errMsg, NSString *streamUrl))completion
```

参数

参数	类型	含义
userID	NSString *	被邀约主播 ID。
completion	void(^)(int errCode, NSString *errMsg, NSString *streamUrl)	请求跨房 PK 的结果回调。

介绍

主播和主播之间可以跨房间 PK，两个正在直播中的主播 A 和 B，他们之间的跨房 PK 流程如下：

1. 主播 A 调用 requestRoomPK 向主播 B 发起连麦请求。
2. 主播 B 会收到 MLVBLiveRoomDelegate onRequestRoomPK 回调通知。
3. 主播 B 调用 responseRoomPK 确定是否接受主播 A 的 PK 请求。
4. 主播 B 如果接受了主播 A 的要求，可以直接调用 startRemoteView 来显示主播 A 的视频画面。
5. 主播 A 会通过传入的 completion 收到回调通知，可以得知请求是否被同意。
6. 主播 A 如果请求被同意，则可以调用 startRemoteView 显示主播 B 的视频画面。

responseRoomPK

响应跨房 PK 请求。

```
- (void)responseRoomPK:(MLVBAnchorInfo *)anchor agree:(BOOL)agree reason:
(NSString *)reason
```

参数

参数	类型	含义
anchor	MLVBAnchorInfo *	发起 PK 请求的主播。
agree	BOOL	YES：同意；NO：拒绝。
reason	NSString *	同意或拒绝 PK 的原因描述。

介绍

主播响应其他房间主播的 PK 请求，发起 PK 请求的主播会收到 MLVBLiveRoomDelegate.onRequestRoomPK 回调通知。

quitRoomPK

退出跨房 PK。

```
- (void)quitRoomPK:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
completion	void(^)(int errCode, NSString *errMsg)	退出跨房 PK 的结果回调。

介绍

当两个主播中的任何一个退出跨房 PK 状态后，另一个主播会收到 MLVBLiveRoomDelegate.onQuitRoomPK 回调通知。

视频相关接口函数

startLocalPreview

开启本地视频的预览画面。

```
- (void)startLocalPreview:(BOOL)frontCamera view:(UIView *)view
```

参数

参数	类型	含义
frontCamera	BOOL	YES：前置摄像头；NO：后置摄像头。
view	UIView *	承载视频画面的控件。

stopLocalPreview

停止本地视频采集及预览。

```
- (void)stopLocalPreview
```

startRemoteView

启动渲染远端视频画面。

```
- (void)startRemoteView:(MLVBAnchorInfo *)anchorInfo view:(UIView *)view
onPlayBegin:(IPlayBegin)onPlayBegin onPlayError:(IPlayError)onPlayError playEvent:
(IPlayEventBlock)onPlayEvent
```

参数

参数	类型	含义
anchorInfo	MLVBAnchorInfo *	对方的用户信息。
view	UIView *	承载视频画面的控件。
onPlayBegin	IPlayBegin	播放器开始回调。
onPlayError	IPlayError	播放出错回调。
onPlayEvent	IPlayEventBlock	其它播放事件回调。

> **说明**
> 在 onUserVideoAvailable 回调时，调用这个接口。

stopRemoteView

停止渲染远端视频画面。

```
- (void)stopRemoteView:(MLVBAnchorInfo *)anchor
```

参数

参数	类型	含义
anchor	MLVBAnchorInfo *	对方的用户。

setMirror

设置观众端镜像效果。

```
- (void)setMirror:(BOOL)isMirror
```

参数

参数	类型	含义
isMirror	BOOL	YES：播放端看到的是镜像画面；NO：播放端看到的是非镜像画面。

介绍

由于前置摄像头采集的画面是取自手机的观察视角，将采集到的画面直接展示给观众是没有问题的，但如果将采集到的画面也直接显示给主播，会让主播感受到和照镜子时完全相反的体验，主播会感到很奇怪。 因此，SDK 会默认开启本地摄像头预览画面的镜像效果，让主播直播时感受到和照镜子一样的体验效果。
setMirror 所影响的是观众端看到的视频效果，如果想要保持观众端看到的效果跟主播端保持一致，需要开启镜像； 如果想要让观众端看到正常的未经处理过的画面（如主播弹吉他的时候有类似需求），则可以关闭镜像。

> **说明**
> 仅当前使用前置摄像头时，setMirror 接口才会生效，**在使用后置摄像头时此接口无效**。

音频相关接口函数

muteLocalAudio

是否屏蔽本地音频。

```
- (void)muteLocalAudio:(BOOL)mute
```

参数

参数	类型	含义

mute	BOOL	YES：屏蔽；NO：开启。

muteRemoteAudio

设置指定用户是否静音。

```
- (void)muteRemoteAudio:(NSString *)userID mute:(BOOL)mute
```

参数

参数	类型	含义
userID	NSString *	对方的用户标识。
mute	BOOL	YES：静音；NO：非静音。

muteAllRemoteAudio

设置所有远端用户是否静音。

```
- (void)muteAllRemoteAudio:(BOOL)mute
```

参数

参数	类型	含义
mute	BOOL	YES：静音；NO：非静音。

摄像头相关接口函数

switchCamera

切换前后摄像头。

```
- (void)switchCamera
```

setCameraMuteImage

主播屏蔽摄像头期间需要显示的等待图片。

```
- (void)setCameraMuteImage:(UIImage *)image
```

参数

参数	类型	含义
image	UIImage *	等待图片。

介绍

当主播屏蔽摄像头，或者由于 App 切入后台无法使用摄像头的时候，我们需要使用一张等待图片来提示观众“主播暂时离开，请不要走开”。

setZoom

调整焦距。

```
- (void)setZoom:(CGFloat)distance
```

参数

参数	类型	含义
distance	CGFloat	焦距大小，取值范围：1 – 5。

> **说明**
> 当为1的时候为最远视角（正常镜头），当为5的时候为最近视角（放大镜头），这里最大值推荐为5，超过5后视频数据会变得模糊不清。

enableTorch

打开闪关灯。

```
- (BOOL)enableTorch:(BOOL)bEnable
```

参数

参数	类型	含义
bEnable	BOOL	YES：打开；NO：关闭。

返回

YES：打开成功；NO：打开失败。

setFocusPosition

设置手动对焦区域。

```
- (void)setFocusPosition:(CGPoint)touchPoint
```

介绍

SDK 默认使用摄像头自动对焦功能，您也可以通过 TXLivePushConfig 中的 touchFocus 选项关闭自动对焦，改用手动对焦。改用手动对焦之后，需要由主播自己单击摄像头预览画面上的某个区域，来手动指导摄像头对焦。

美颜滤镜相关接口函数

getBeautyManager

获取美颜管理对象 TXBeautyManager 。

```
- (TXBeautyManager *)getBeautyManager
```

> **说明**
>
> 通过美颜管理，您可以使用以下功能：
>
> - 设置”美颜风格”、”美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小鼻”、“亮眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。
> - 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”。
> - 设置人脸挂件（素材）等动态效果。
> - 添加美妆。
> - 进行手势识别。

setFilter

设置指定素材滤镜特效。

```
- (void)setFilter:(UIImage *)image
```

参数

参数	类型	含义
image	UIImage *	指定素材，即颜色查找表图片。

> **说明**

滤镜素材请使用 png 格式，不能使用 jpg 格式。友情提示：Windows 里直接改文件的后缀名不能改变图片的格式，需要用 Photoshop 进行转换。

setSpecialRatio

设置滤镜浓度。

```
- (void)setSpecialRatio:(float)specialValue
```

参数

参数	类型	含义
specialValue	float	从0到1，越大滤镜效果越明显，默认取值0.5。

setGreenScreenFile

设置绿幕背景视频（商业版有效，其它版本设置此参数无效）。

```
- (void)setGreenScreenFile:(NSURL *)file
```

参数

参数	类型	含义
file	NSURL *	视频文件路径。支持 MP4；nil 表示关闭特效。

说明

此处的绿幕功能并非智能抠背，它需要被拍摄者的背后有一块绿色的幕布来辅助产生特效。

消息发送接口函数

sendRoomTextMsg

发送文本消息。

```
- (void)sendRoomTextMsg:(NSString *)message completion:(void(^)(int errCode,
NSString *errMsg))completion
```

参数

参数	类型	含义

message	NSString *	文本消息。
completion	void(^)(int errCode, NSString *errMsg)	发送结果回调。

sendRoomCustomMsg

发送自定义文本消息。

```
- (void)sendRoomCustomMsg:(NSString *)cmd msg:(NSString *)message completion:(void(^)(int errCode, NSString *errMsg))completion
```

参数

参数	类型	含义
cmd	NSString *	命令字，由开发者自定义，主要用于区分不同消息类型。
message	NSString *	文本消息。
completion	void(^)(int errCode, NSString *errMsg)	发送结果回调。

背景混音相关接口函数

playBGM

播放背景音乐。

```
- (BOOL)playBGM:(NSString *)path
```

参数

参数	类型	含义
path	NSString *	音乐文件路径，一定要是 `app` 对应的 `document` 目录下面的路径，否则文件会读取失败。

返回

YES：成功；NO：失败。

playBGM

播放背景音乐（高级版本）。

```
- (BOOL)playBGM:(NSString *)path withBeginNotify:(void(^)(NSInteger errCode))beginNotify withProgressNotify:(void(^)(NSInteger progressMS, NSInteger durationMS))progressNotify andCompleteNotify:(void(^)(NSInteger errCode))completeNotify
```

参数

参数	类型	含义
path	NSString *	音乐文件路径，一定要是 `app` 对应的 `document` 目录下面的路径，否则文件会读取失败。
beginNotify	void(^)(NSInteger errCode)	音乐播放开始的回调通知。
progressNotify	void(^)(NSInteger progressMS, NSInteger durationMS)	音乐播放的进度通知，单位：毫秒。
completeNotify	void(^)(NSInteger errCode)	音乐播放结束的回调通知。

返回

YES：成功；NO：失败。

stopBGM

停止播放背景音乐。

```
- (BOOL)stopBGM
```

pauseBGM

暂停播放背景音乐。

```
- (BOOL)pauseBGM
```

resumeBGM

继续播放背景音乐。

```
- (BOOL)resumeBGM
```

getMusicDuration

获取音乐文件总时长，单位毫秒。

```
- (int)getMusicDuration:(NSString *)path
```

参数

参数	类型	含义
path	NSString *	音乐文件路径，如果 path 为 nil，那么返回当前正在播放的背景音乐时长。

返回

成功返回时长，单位毫秒，失败返回-1。

setMicVolume

设置麦克风的音量大小，播放背景音乐混音时使用，用来控制麦克风音量大小。

```
- (BOOL)setMicVolume:(float)volume
```

参数

参数	类型	含义
volume	float	音量大小，1.0 为正常音量，建议值为0.0 - 2.0。

setBGMVolume

设置背景音乐的音量大小，播放背景音乐混音时使用，用来控制背景音音量大小。

```
- (BOOL)setBGMVolume:(float)volume
```

参数

参数	类型	含义
volume	float	音量大小，1.0为正常音量，建议值为0.0 - 2.0。

setBGMPitch

调整背景音乐的音调高低。

```
- (BOOL)setBGMPitch:(float)pitch
```

参数

参数	类型	含义
pitch	float	音调，默认值是0.0f，范围：-1 – 1之间的浮点数。

返回

YES：成功；NO：失败。

setReverbType

设置混响效果。

```
- (BOOL)setReverbType:(TXReverbType)reverbType
```

参数

参数	类型	含义
reverbType	TXReverbType	混响类型，详见 TXLiveSDKTypeDef.h 中的 TXReverbType 定义。

返回

YES：成功；NO：失败。

setVoiceChangerType

设置变声类型。

```
- (BOOL)setVoiceChangerType:(TXVoiceChangerType)voiceChangerType
```

参数

参数	类型	含义
voiceChangerType	TXVoiceChangerType	混响类型，详见 TXLiveSDKTypeDef.h 中的 voiceChangerType 定义。

返回

YES：成功；NO：失败。

调试相关接口函数

showVideoDebugLog

在渲染 view 上显示播放或推流状态统计及事件消息浮层。

```
- (void)showVideoDebugLog:(BOOL)isShow
```

MLVBLiveRoomDelegate

最近更新时间：2022-12-20 18:20:58

功能

MLVBLiveRoom 事件回调。

介绍

包括房间关闭、Debug 事件信息及出错说明等。

通用事件回调

onError

错误回调。

```
- (void)onError:(int)errCode errMsg:(NSString *)errMsg extraInfo:(NSDictionary *)extraInfo
```

参数

参数	类型	含义
errCode	int	错误码。
errMsg	NSString *	错误信息。
extraInfo	NSDictionary *	额外信息，如错误发生的用户，一般不需要关注，默认是本地错误。

介绍

SDK 不可恢复的错误，一定要监听，并分情况给用户适当的界面提示。

onWarning

警告回调。

```
- (void)onWarning:(int)warningCode warningMsg:(NSString *)warningMsg extraInfo:(NSDictionary *)extraInfo
```

参数

参数	类型	含义
warningCode	int	错误码 TRTCWarningCode。

warningMsg	NSString *	警告信息。
extraInfo	NSDictionary *	额外信息，如警告发生的用户，一般不需要关注，默认是本地错误。

onDebugLog

Log 回调。

```
- (void)onDebugLog:(NSString *)log
```

参数

参数	类型	含义
log	NSString *	LOG 信息。

房间事件回调

onRoomDestroy

房间被销毁的回调。

```
- (void)onRoomDestroy:(NSString *)roomID
```

参数

参数	类型	含义
roomID	NSString *	房间 ID。

介绍

主播退房时，房间内的所有用户都会收到此通知。

主播和观众的进出事件回调

onAnchorEnter

收到新主播进房通知。

```
- (void)onAnchorEnter:(MLVBAnchorInfo *)anchorInfo
```

参数

参数	类型	含义
anchorInfo	MLVBAnchorInfo *	新进房用户信息。

介绍

房间内的主播和连麦中的观众会收到新主播的进房事件，您可以调用 MLVBLiveRoom startRemoteView 显示该主播的视频画面。

> **说明**
> 直播间里的普通观众不会收到主播加入和推出的通知。

onAnchorExit

收到主播退房通知。

```
- (void)onAnchorExit:(MLVBAnchorInfo *)anchorInfo
```

参数

参数	类型	含义
anchorInfo	MLVBAnchorInfo *	退房用户信息。

介绍

房间内的主播（和连麦中的观众）会收到新主播的退房事件，您可以调用 MLVBLiveRoom stopRemoteView 关闭该主播的视频画面。

> **说明**
> 直播间里的普通观众不会收到主播加入和推出的通知。

onAudienceEnter

收到观众进房通知。

```
- (void)onAudienceEnter:(MLVBAudienceInfo *)audienceInfo
```

参数

参数	类型	含义
audienceInfo	MLVBAudienceInfo *	进房观众信息。

onAudienceExit

收到观众退房通知。

```
- (void)onAudienceExit:(MLVBAudienceInfo *)audienceInfo
```

参数

参数	类型	含义
audienceInfo	MLVBAudienceInfo *	退房观众信息。

主播和观众连麦事件回调

onRequestJoinAnchor

主播收到观众连麦请求时的回调。

```
- (void)onRequestJoinAnchor:(MLVBAnchorInfo *)anchorInfo reason:(NSString *)reason
```

参数

参数	类型	含义
anchorInfo	MLVBAnchorInfo *	观众信息。
reason	NSString *	连麦原因描述。

onKickoutJoinAnchor

连麦观众收到被踢出连麦的通知。

```
- (void)onKickoutJoinAnchor
```

介绍

连麦观众收到被主播踢除连麦的消息，您需要调用 MLVBLiveRoom kickoutJoinAnchor 来退出连麦。

主播 PK 事件回调

onRequestRoomPK

收到请求跨房 PK 通知。

```
- (void)onRequestRoomPK:(MLVBAnchorInfo *)anchorInfo
```

参数

参数	类型	含义
anchorInfo	MLVBAnchorInfo *	发起跨房连麦的主播信息。

介绍

主播收到其他房间主播的 PK 请求，如果同意 PK ，您需要调用 MLVBLiveRoom startRemoteView 接口播放邀约主播的流。

onQuitRoomPK

收到断开跨房 PK 通知。

```
- (void)onQuitRoomPK
```

消息事件回调

onRecvRoomTextMsg

收到文本消息。

```
- (void)onRecvRoomTextMsg:(NSString *)roomID userID:(NSString *)userID userName:
(NSString *)userName userAvatar:(NSString *)userAvatar message:(NSString
*)message
```

参数

参数	类型	含义
roomID	NSString *	房间 ID。
userID	NSString *	发送者 ID。
userName	NSString *	发送者昵称。
userAvatar	NSString *	发送者头像。
message	NSString *	文本消息。

onRecvRoomCustomMsg

收到自定义消息。

```
- (void)onRecvRoomCustomMsg:(NSString *)roomID userID:(NSString *)userID
userName:(NSString *)userName userAvatar:(NSString *)userAvatar cmd:(NSString
*)cmd message:(NSString *)message
```

参数

参数	类型	含义
roomID	NSString *	房间 ID。
userID	NSString *	发送者 ID。
userName	NSString *	发送者昵称。
userAvatar	NSString *	发送者头像。
cmd	NSString *	自定义 cmd。
message	NSString *	自定义消息内容。

示例

最近更新时间：2022-12-22 10:51:38

针对开发者的接入反馈的高频问题，腾讯云提供有直观易懂的 API-Example 工程，方便开发者可以快速的了解相关 API 的使用，欢迎使用。

工程地址

所属平台	GitHub 地址
iOS	Github

目录说明

在这个示例项目中包含了以下场景：

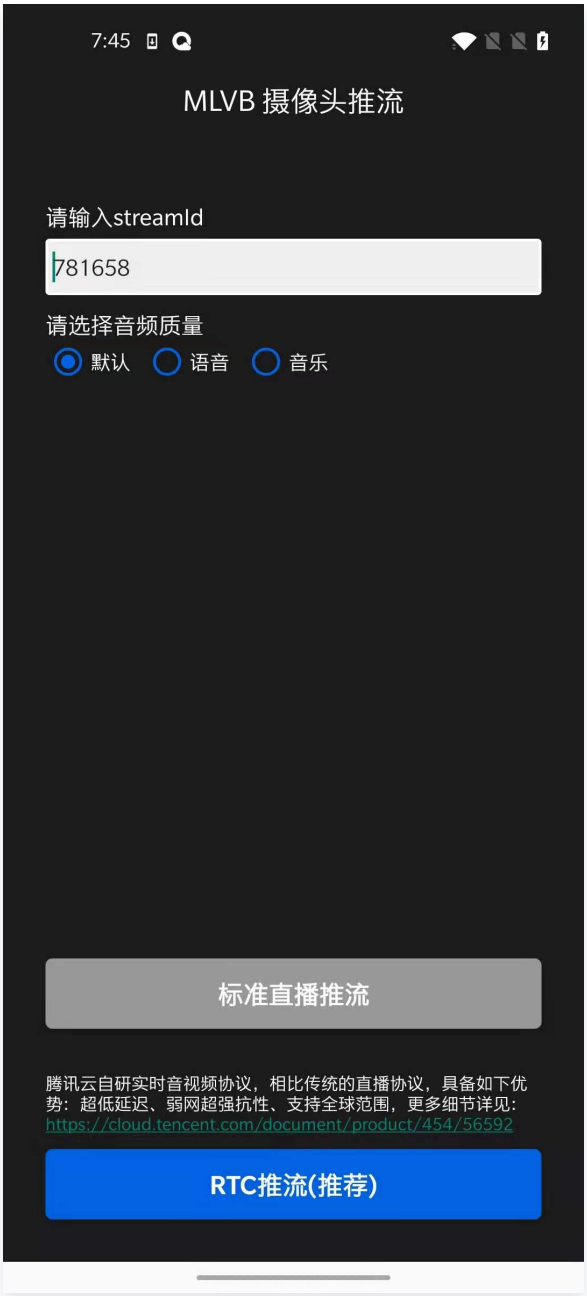
- **基础功能：**
  - 摄像头推流
  - 录屏推流
  - 直播拉流
  - 连麦互动
  - 连麦 PK
- **进阶功能：**
  - 自定义视频采集
  - 第三方美颜
  - RTC 连麦 + 超低延时播放

> **说明**
> 目前的工程结构跟标准的 xcproject 工程在名称大小写上可能有略微的差异，主要目的是方便大家在网页上看到此工程时，名称意义更加清晰。

应用图示

示例主页	推流示例	拉流示例


7:45
腾讯云MLVB API Example
基础功能
摄像头推流
录屏推流
直播拉流
连麦互动
PK互动
进阶功能
动态切换渲染组件
自定义视频采集
第三方美颜
RTC连麦+超低延时播放


7:45
MLVB 摄像头推流
请输入streamId
781658
请选择音频质量
默认
语音
音乐
标准直播推流
腾讯云自研实时音视频协议，相比传统的直播协议，具备如下优势：超低延迟、弱网超强抗性、支持全球范围，更多细节详见：
https://cloud.tencent.com/document/product/454/56592
RTC推流(推荐)

Android API 概览

最近更新时间：2024-01-16 14:35:22

API OVERVIEW

V2 推流器相关接口

函数列表	描述
release	释放 V2TXLivePusher 资源
setObserver	设置推流器回调
setRenderView	设置本地摄像头预览 View
setRenderMirror	设置本地摄像头预览镜像
setEncoderMirror	设置视频编码镜像
setRenderRotation	设置本地摄像头预览画面的旋转角度
setRenderFillMode	设置本地摄像头预览画面的填充模式
startCamera	打开本地摄像头
stopCamera	关闭本地摄像头
startMicrophone	打开麦克风
stopMicrophone	关闭麦克风
startVirtualCamera	开启图片推流
stopVirtualCamera	关闭图片推流
startScreenCapture	开启屏幕采集
stopScreenCapture	关闭屏幕采集
pauseAudio	暂停推流器的音频流
resumeAudio	恢复推流器的音频流

pauseVideo	暂停推流器的视频流
resumeVideo	恢复推流器的视频流
startPush	开始音视频数据推流
stopPush	停止推送音视频数据
isPushing	当前推流器是否正在推流中
setAudioQuality	设置推流音频质量
setVideoQuality	设置推流视频编码参数
getAudioEffectManager	获取音效管理对象
getBeautyManager	获取美颜管理对象
getDeviceManager	获取设备管理对象
snapshot	截取推流过程中的本地画面
setWatermark	设置推流器水印。默认情况下，水印不开启
enableVolumeEvaluation	启用采集音量大小提示
enableCustomVideoProcess	开启/关闭自定义视频处理
enableCustomVideoCapture	开启/关闭自定义视频采集
enableCustomAudioCapture	开启/关闭自定义音频采集
sendCustomVideoFrame	在自定义视频采集模式下，将采集的视频数据发送到SDK
sendCustomAudioFrame	在自定义音频采集模式下，将采集的音频数据发送到SDK
enableAudioProcessObserver	开启/关闭对经过前处理后的本地音频帧的监听回调
sendSeiMessage	发送 SEI 消息
startSystemAudioLo	打开系统声音采集

opback	
stopSystemAudioLoopback	关闭系统声音采集
showDebugView	显示仪表盘。
setProperty	调用 V2TXLivePusher 的高级 API 接口。
setMixTranscodingConfig	设置云端的混流转码参数
startLocalRecording	开始录制音视频流
stopLocalRecording	停止录制音视频流

直播推流器事件回调

函数列表	描述
onError	直播推流器错误通知，推流器出现错误时，会回调该通知
onWarning	直播推流器警告通知
onCaptureFirstAudioFrame	首帧音频采集完成的回调通知
onCaptureFirstVideoFrame	首帧视频采集完成的回调通知
onMicrophoneVolumeUpdate	麦克风采集音量值回调
onPushStatusUpdate	推流器连接状态回调通知
onStatisticsUpdate	直播推流器统计数据回调
onSnapshotComplete	截图回调
onGLContextCreated	SDK 内部的 OpenGL 环境的创建通知
onProcessAudioFrame	本地采集并经过音频模块前处理、音效处理和混 BGM 后的音频数据回调
onProcessVideoFra	自定义视频处理回调

me	
onGLContextDestroyed	SDK 内部的 OpenGL 环境的销毁通知
onSetMixTranscodingConfig	设置云端的混流转码参数的回调，对应于 setMixTranscodingConfig 接口
onScreenCaptureStarted	当屏幕分享开始时，SDK 会通过此回调通知
onScreenCaptureStopped	当屏幕分享停止时，SDK 会通过此回调通知
onLocalRecordBegin	录制任务开始的事件回调
onLocalRecording	录制任务正在进行中的进展事件回调
onLocalRecordComplete	录制任务已经结束的事件回调

直播播放器事件回调

函数列表	描述
onError	直播播放器错误通知，播放器出现错误时，会回调该通知
onWarning	直播播放器警告通知
onVideoResolutionChanged	直播播放器分辨率变化通知
onConnected	已经成功连接到服务器
onVideoPlaying	视频播放事件
onAudioPlaying	音频播放事件
onVideoLoading	视频加载事件
onAudioLoading	音频加载事件
onPlayoutVolumeUpdate	播放器音量大小回调
onStatisticsUpdate	直播播放器统计数据回调
onSnapshotComple	截图回调

te	
onRenderVideoFrame	自定义视频渲染回调
onPlayoutAudioFrame	音频数据回调
onReceiveSeiMessage	收到 SEI 消息的回调，发送端通过 V2TXLivePusher 中的 `sendSeiMessage` 来发送 SEI 消息
onStreamSwitched	分辨率无缝切换回调
onLocalRecordBegin	录制任务开始的事件回调
onLocalRecording	录制任务正在进行中的进展事件回调
onLocalRecordComplete	录制任务已经结束的事件回调

V2 播放器相关接口

函数列表	描述
setObserver	设置播放器回调
setRenderView	设置播放器的视频渲染 View，该控件负责显示视频内容
setRenderRotation	设置播放器画面的旋转角度
setRenderFillMode	设置画面的填充模式
startLivePlay	开始播放音视频流
stopPlay	停止播放音视频流
isPlaying	播放器是否正在播放中
pauseAudio	暂停播放器的音频流
resumeAudio	恢复播放器的音频流
pauseVideo	暂停播放器的视频流
resumeVideo	恢复播放器的视频流
setPlayoutVolume	设置播放器音量

setCacheParams	设置播放器缓存自动调整的最小和最大时间（单位：秒）
switchStream	直播流无缝切换，支持 FLV 和 LEB
getStreamList	获取码流信息
enableVolumeEvaluation	启用播放音量大小提示
snapshot	截取播放过程中的视频画面
enableObserveVideoFrame	开启/关闭对视频帧的监听回调
enableObserveAudioFrame	开启/关闭对音频数据的监听回调
enableReceiveSeiMessage	开启接收 SEI 消息
showDebugView	是否显示播放器状态信息的调试浮层
setProperty	调用 V2TXLivePlayer 的高级 API 接口
startLocalRecording	开始录制音视频流
stopLocalRecording	停止录制音视频流

V2TXLive 高级接口

函数列表	描述
getSDKVersionStr	获取 SDK 版本号
setObserver	设置 V2TXLivePremier 回调接口
setLogConfig	设置 Log 的配置信息
setEnvironment	设置 SDK 接入环境
setLicence	设置 SDK 的授权 License
setSocks5Proxy	设置 SDK socks5 代理配置
enableAudioCaptureObserver	开启/关闭对音频采集数据的监听回调（可读写）

enableAudioPlayoutObserver	开启/关闭对最终系统要播放出的音频数据的监听回调
enableVoiceEarMonitorObserver	开启/关闭耳返音频数据的监听回调
setUserId	设置 userId
callExperimentalAPI	调用实验性 API 接口

V2TXLive 高级回调接口

函数列表	描述
onLog	自定义 Log 输出回调接口
onLicenceLoaded	setLicence 接口回调
onCaptureAudioFrame	本地麦克风采集到的音频数据回调
onPlayoutAudioFrame	将各路待播放音频混合之后并在最终提交系统播放之前的数据回调
onVoiceEarMonitorAudioFrame	耳返的音频数据

背景音乐预加载事件回调

函数列表	描述
onLoadProgress	背景音乐预加载进度
onLoadError	背景音乐预加载出错

背景音乐的播放事件回调

函数列表	描述
onStart	背景音乐开始播放
onPlayProgress	背景音乐的播放进度
onComplet	背景音乐已经播放完毕

e	

人声相关的特效接口

函数列表	描述
enableVoiceEarMonitor	开启耳返
setVoiceEarMonitorVolume	设置耳返音量
setVoiceReverbType	设置人声的混响效果
setVoiceChangerType	设置人声的变声特效
setVoiceCaptureVolume	设置语音音量
setVoicePitch	设置语音音调

背景音乐的相关接口

函数列表	描述
setMusicObserver	设置背景音乐的事件回调接口
startPlayMusic	开始播放背景音乐
stopPlayMusic	停止播放背景音乐
pausePlayMusic	暂停播放背景音乐
resumePlayMusic	恢复播放背景音乐
setAllMusicVolume	设置所有背景音乐的本地音量和远端音量的大小
setMusicPublishVolume	设置某一首背景音乐的远端音量的大小
setMusicPlayoutVolume	设置某一首背景音乐的本地音量的大小
setMusicPitch	调整背景音乐的音调高低
setMusicSpeedRate	调整背景音乐的变速效果

getMusicCurrentPosInMS	获取背景音乐的播放进度（单位：毫秒）
getMusicDurationInMS	获取背景音乐的总时长（单位：毫秒）
seekMusicToPosInMS	设置背景音乐的播放进度（单位：毫秒）
setMusicScratchSpeedRate	调整搓碟的变速效果
setPreloadObserver	设置预加载事件回调
preloadMusic	预加载背景音乐
getMusicTrackCount	获取背景音乐的音轨数量
setMusicTrack	指定背景音乐的播放音轨

美颜相关接口

函数列表	描述
setBeautyStyle	设置美颜（磨皮）算法
setBeautyLevel	设置美颜级别
setWhitenessLevel	设置美白级别
enableSharpnessEnhancement	开启清晰度增强
setRuddyLevel	设置红润级别
setFilter	设置色彩滤镜效果
setFilterStrength	设置色彩滤镜的强度
setGreenScreenFile	设置绿幕背景视频
setEyeScaleLevel	设置大眼级别
setFaceSlimLevel	设置瘦脸级别
setFaceVLevel	设置 V 脸级别

setChinLevel	设置下巴拉伸或收缩
setFaceShortLevel	设置短脸级别
setFaceNarrowLevel	设置窄脸级别
setNoseSlimLevel	设置瘦鼻级别
setEyeLightenLevel	设置亮眼级别
setToothWhitenLevel	设置牙齿美白级别
setWrinkleRemoveLevel	设置祛皱级别
setPounchRemoveLevel	设置祛眼袋级别
setSmileLinesRemoveLevel	设置法令纹去除级别
setForeheadLevel	设置发际线调整级别
setEyeDistanceLevel	设置眼距
setEyeAngleLevel	设置眼角调整级别
setMouthShapeLevel	设置嘴型调整级别
setNoseWingLevel	设置鼻翼调整级别
setNosePositionLevel	设置鼻子位置
setLipsThicknessLevel	设置嘴唇厚度
setFaceBeautyLevel	设置脸型
setMotionTmpl	选择 AI 动效挂件
setMotionMute	是否在动效素材播放时静音

设备操作接口

函数列表	描述

isFrontCamera	判断当前是否为前置摄像头（仅适用于移动端）
switchCamera	切换前置或后置摄像头（仅适用于移动端）
getCameraZoomMaxRatio	获取摄像头的最大缩放倍数（仅适用于移动端）
setCameraZoomRatio	设置摄像头的缩放倍数（仅适用于移动端）
isAutoFocusEnabled	查询是否支持自动识别人脸位置（仅适用于移动端）
enableCameraAutoFocus	开启自动对焦功能（仅适用于移动端）
setCameraFocusPosition	设置摄像头的对焦位置（仅适用于移动端）
enableCameraTorch	开启/关闭闪光灯，也就是手电筒模式（仅适用于移动端）
setAudioRoute	设置音频路由（仅适用于移动端）
setExposureCompensation	设置摄像头的曝光参数，取值范围从-1到1
setCameraCapturerParam	设置摄像头采集偏好

弃用接口

函数列表	描述
setSystemVolumeType	设置系统音量类型（仅适用于移动端）

V2TXLivePusher

最近更新时间：2024-01-16 14:35:22

Module: V2TXLivePusher @ TXLiteAVSDK

Function: 腾讯云直播推流器

功能

腾讯云直播推流器

介绍

主要负责将本地的音频和视频画面进行编码，并推送到指定的推流地址，支持任意的推流服务端。
推流器包含如下能力：

- 自定义的视频采集，让您可以根据项目需要定制自己的音视频数据源。
- 美颜、滤镜、贴纸，包含多套美颜磨皮算法（自然&光滑）和多款色彩空间滤镜（支持自定义滤镜）。
- Qos 流量控制技术，具备上行网络自适应能力，可以根据主播端网络的具体情况实时调节音视频数据量。
- 脸型调整、动效挂件，支持基于优图 AI 人脸识别技术的大眼、瘦脸、隆鼻等脸型微调以及动效挂件效果，只需要购买 优图 License 就可以轻松实现丰富的直播效果。

V2TXLivePusher

V2TXLivePusher

函数列表	描述
release	释放 V2TXLivePusher 资源
setObserver	设置推流器回调
setRenderView	设置本地摄像头预览 View
setRenderView	设置本地摄像头预览 View
setRenderView	设置本地摄像头预览 View

setRenderMirror	设置本地摄像头预览镜像
setEncoderMirror	设置视频编码镜像
setRenderRotation	设置本地摄像头预览画面的旋转角度
setRenderFillMode	设置本地摄像头预览画面的填充模式
startCamera	打开本地摄像头
stopCamera	关闭本地摄像头
startMicrophone	打开麦克风
stopMicrophone	关闭麦克风
startVirtualCamera	开启图片推流
stopVirtualCamera	关闭图片推流
startScreenCapture	开启屏幕采集
stopScreenCapture	关闭屏幕采集
pauseAudio	暂停推流器的音频流
resumeAudio	恢复推流器的音频流
pauseVideo	暂停推流器的视频流
resumeVideo	恢复推流器的视频流
startPush	开始音视频数据推流
stopPush	停止推送音视频数据
isPushing	当前推流器是否正在推流中
setAudioQuality	设置推流音频质量
setVideoQuality	设置推流视频编码参数
getAudioEffectManager	获取音效管理对象
getBeautyManager	获取美颜管理对象
getDeviceManager	获取设备管理对象
snapshot	截取推流过程中的本地画面

setWatermark	设置推流器水印。默认情况下，水印不开启
enableVolumeEvaluation	启用采集音量大小提示
enableCustomVideoProcess	开启/关闭自定义视频处理
enableCustomVideoCapture	开启/关闭自定义视频采集
enableCustomAudioCapture	开启/关闭自定义音频采集
sendCustomVideoFrame	在自定义视频采集模式下，将采集的视频数据发送到SDK
sendCustomAudioFrame	在自定义音频采集模式下，将采集的音频数据发送到SDK
enableAudioProcessObserver	开启/关闭对经过前处理后的本地音频帧的监听回调
sendSeiMessage	发送 SEI 消息
startSystemAudioLoopback	打开系统声音采集
stopSystemAudioLoopback	关闭系统声音采集
showDebugView	显示仪表盘。
setProperty	调用 V2TXLivePusher 的高级 API 接口。
setMixTranscodingConfig	设置云端的混流转码参数
startLocalRecording	开始录制音视频流
stopLocalRecording	停止录制音视频流

release

release

释放 V2TXLivePusher 资源

setObserver

setObserver

void setObserver （V2TXLivePusherObserver observer）

设置推流器回调

通过设置回调，可以监听 V2TXLivePusher 推流器的一些回调事件，
包括推流器状态、音量回调、统计数据、警告和错误信息等。

参数	描述
observer	推流器的回调目标对象，更多信息请查看 V2TXLivePusherObserver 。

setRenderView

setRenderView

int setRenderView （TXCloudVideoView view）

设置本地摄像头预览 View

本地摄像头采集到的画面，经过美颜、脸型调整、滤镜等多种效果叠加之后，最终会显示到传入的 View 上。

参数	描述
view	本地摄像头预览 View。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK：成功。

setRenderView

setRenderView

int setRenderView (TextureView view)

设置本地摄像头预览 View

本地摄像头采集到的画面，经过美颜、脸型调整、滤镜等多种效果叠加之后，最终会显示到传入的 View 上。

参数	描述
view	本地摄像头预览 View。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK：成功。

setRenderView

setRenderView

int setRenderView (SurfaceView view)

设置本地摄像头预览 View

本地摄像头采集到的画面，经过美颜、脸型调整、滤镜等多种效果叠加之后，最终会显示到传入的 View 上。

参数	描述
view	本地摄像头预览 View。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK：成功。

setRenderMirror

setRenderMirror

int setRenderMirror (V2TXLiveMirrorType mirrorType)

设置本地摄像头预览镜像

本地摄像头分为前置摄像头和后置摄像头，系统默认情况下，是前置摄像头镜像，后置摄像头不镜像，这里可以修改前置后置摄像头的默认镜像类型。

参数	描述
mirrorType	摄像头镜像类型 V2TXLiveMirrorType 。 • V2TXLiveMirrorTypeAuto 【默认值】: 默认镜像类型. 在这种情况下，前置摄像头的画面是镜像的，后置摄像头的画面不是镜像的。 • V2TXLiveMirrorTypeEnable: 前置摄像头 和 后置摄像头，都切换为镜像模式。 • V2TXLiveMirrorTypeDisable: 前置摄像头 和 后置摄像头，都切换为非镜像模式。

返回值说明:

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setEncoderMirror

setEncoderMirror

int setEncoderMirror (boolean mirror)

设置视频编码镜像

参数	描述
mirror	是否镜像。 • false【默认值】: 播放端看到的是非镜像画面。 • true: 播放端看到的是镜像画面。

> **注意**
> 编码镜像只影响观众端看到的视频效果。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setRenderRotation

setRenderRotation

int setRenderRotation （V2TXLiveRotation rotation）

设置本地摄像头预览画面的旋转角度

参数	描述
rotation	预览画面的旋转角度 V2TXLiveRotation 。 • V2TXLiveRotation0【默认值】: 0度, 不旋转。 • V2TXLiveRotation90: 顺时针旋转90度。 • V2TXLiveRotation180: 顺时针旋转180度。 • V2TXLiveRotation270: 顺时针旋转270度。

注意
只旋转本地预览画面，不影响推流出去的画面。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setRenderFillMode

setRenderFillMode

int setRenderFillMode （V2TXLiveFillMode mode）

设置本地摄像头预览画面的填充模式

参数	描述

mode	画面填充模式 V2TXLiveFillMode 。 • V2TXLiveFillModeFill【默认值】: 图像铺满屏幕，不留黑边，如果图像宽高比不同于屏幕宽高比，部分画面内容会被裁剪掉。 • V2TXLiveFillModeFit: 图像适应屏幕，保持画面完整，但如果图像宽高比不同于屏幕宽高比，会有黑边的存在。 • V2TXLiveFillModeScaleFill: 图像拉伸铺满，因此长度和宽度可能不会按比例变化。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startCamera

startCamera

int startCamera (boolean frontCamera)

打开本地摄像头

参数	描述
frontCamera	指定摄像头方向是否为前置。 • true【默认值】: 切换到前置摄像头。 • false: 切换到后置摄像头。

注意
startVirtualCamera，startCamera，startScreenCapture，同一 Pusher 实例下，仅有一个采集源可以上行，不同采集源之间切换，请先关闭前一采集源，再开启后一采集源，保证同一采集源的开启和关闭是成对调用的。比如：采集源从Camera切换到VirtualCamera，调用顺序是 startCamera -> stopCamera -> startVirtualCamera。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

stopCamera

stopCamera

关闭本地摄像头

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startMicrophone

startMicrophone

打开麦克风

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

stopMicrophone

stopMicrophone

关闭麦克风

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startVirtualCamera

startVirtualCamera

int startVirtualCamera (Bitmap image)

开启图片推流

参数	描述
image	图片。

> **注意**
> startVirtualCamera，startCamera，startScreenCapture，同一 Pusher 实例下，仅有一个采集源可以上行，不同采集源之间切换，请先关闭前一采集源，再开启后一采集源，保证同一采集源的开启和关闭是成对调用的。比如：采集源从Camera切换到VirtualCamera，调用顺序是 startCamera -> stopCamera -> startVirtualCamera。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

stopVirtualCamera

stopVirtualCamera

关闭图片推流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startScreenCapture

startScreenCapture

开启屏幕采集

> **注意**

> startVirtualCamera，startCamera，startScreenCapture，同一 Pusher 实例下，仅有一个采集源可以上行，不同采集源之间切换，请先关闭前一采集源，再开启后一采集源，保证同一采集源的开启和关闭是成对调用的。比如：采集源从Camera切换到ScreenCapture，调用顺序是 startCamera -> stopCamera -> startScreenCapture。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

stopScreenCapture

stopScreenCapture

关闭屏幕采集

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

pauseAudio

pauseAudio

暂停推流器的音频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

resumeAudio

resumeAudio

恢复推流器的音频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

pauseVideo

pauseVideo

暂停推流器的视频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

resumeVideo

resumeVideo

恢复推流器的视频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startPush

startPush

int startPush (String url)

开始音视频数据推流

参数	描述
url	推流的目标地址，支持任意推流服务端。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 操作成功，开始连接推流目标地址。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，url 不合法。
- V2TXLIVE_ERROR_INVALID_LICENSE: 操作失败，license 不合法，鉴权失败。
- V2TXLIVE_ERROR_REFUSED: 操作失败，RTC 不支持同一设备上同时推拉同一个 StreamId。

stopPush

stopPush

停止推送音视频数据

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

isPushing

isPushing

当前推流器是否正在推流中

返回值说明：

是否正在推流。

- 1: 正在推流中。
- 0: 已经停止推流。

setAudioQuality

setAudioQuality

int setAudioQuality (V2TXLiveAudioQuality quality)

设置推流音频质量

参数	描述
quality	音频质量 V2TXLiveAudioQuality 。 • V2TXLiveAudioQualityDefault【默认值】: 通用。 • V2TXLiveAudioQualitySpeech: 语音。 • V2TXLiveAudioQualityMusic: 音乐。

返回值说明:

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 推流过程中，不允许调整音质。

setVideoQuality

setVideoQuality

int setVideoQuality (V2TXLiveVideoEncoderParam param)

设置推流视频编码参数

参数	描述
param	视频编码参数 V2TXLiveVideoEncoderParam 。

返回值说明:

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

getAudioEffectManager

getAudioEffectManager

获取音效管理对象

通过音效管理，您可以使用以下功能：

- 调整麦克风收集的人声音量。
- 设置混响和变声效果。
- 开启耳返，设置耳返音量。
- 添加背景音乐，调整背景音乐的播放效果。

参考 TXAudioEffectManager

getBeautyManager

getBeautyManager

获取美颜管理对象

通过美颜管理，您可以使用以下功能：

- 设置”美颜风格”、”美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小鼻”、“亮眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。
- 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”。
- 设置人脸挂件（素材）等动态效果。
- 添加美妆。
- 进行手势识别。

参考 TXBeautyManager

getDeviceManager

getDeviceManager

获取设备管理对象

通过设备管理，您可以使用以下功能：

- 切换前后摄像头。
- 设置自动聚焦。

- 设置摄像头缩放倍数。
- 打开或关闭闪光灯。
- 切换耳机或者扬声器。
- 修改音量类型(媒体音量或者通话音量)。

参考 TXDeviceManager

snapshot

snapshot

截取推流过程中的本地画面

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 已经停止推流，不允许调用截图操作。

setWatermark

setWatermark

```
int setWatermark (Bitmap image
                  float x
                  float y
                  float scale)
```

设置推流器水印。默认情况下，水印不开启

参数	描述
image	水印图片。如果该值为 null，则等效于禁用水印。
scale	水印图片的缩放比例，取值范围为0 – 1的浮点数。

x	水印的横坐标，取值范围为0 – 1的浮点数。
y	水印的纵坐标，取值范围为0 – 1的浮点数。

返回值说明：

返回值 V2TXLiveCode
- V2TXLIVE_OK: 成功

enableVolumeEvaluation

enableVolumeEvaluation

int enableVolumeEvaluation (int intervalMs)

启用采集音量大小提示

开启后可以在 onMicrophoneVolumeUpdate 回调中获取到 SDK 对音量大小值的评估。

参数	描述
intervalMs	决定了 onMicrophoneVolumeUpdate 回调的触发间隔，单位为ms，最小间隔为100ms，如果小于等于0则会关闭回调，建议设置为300ms；【默认值】：0，不开启。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

enableCustomVideoProcess

enableCustomVideoProcess

int enableCustomVideoProcess (boolean enable

V2TXLivePixelFormat pixelFormat

V2TXLiveBufferType bufferType)

开启/关闭自定义视频处理

参数	描述
enable	true: 开启; false: 关闭。【默认值】: false。

> **注意**
> 支持的格式组合：
> V2TXLivePixelFormatTexture2D+V2TXLiveBufferTypeTexture
> V2TXLivePixelFormatI420+V2TXLiveBufferTypeByteBuffer

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_NOT_SUPPORTED: 不支持的格式。

enableCustomVideoCapture

enableCustomVideoCapture

int enableCustomVideoCapture (boolean enable)

开启/关闭自定义视频采集

在自定义视频采集模式下，SDK 不再从摄像头采集图像，只保留编码和发送能力。

参数	描述
enable	true：开启自定义采集；false：关闭自定义采集。【默认值】：false。

> **注意**
> 需要在 startPush 之前调用，才会生效。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

enableCustomAudioCapture

enableCustomAudioCapture

int enableCustomAudioCapture (boolean enable)

开启/关闭自定义音频采集

@brief 开启/关闭自定义音频采集。
在自定义音频采集模式下，SDK 不再从麦克风采集声音，只保留编码和发送能力。

参数	描述
enable	true: 开启自定义采集; false: 关闭自定义采集。【默认值】: false。

注意
需要在 startPush 前调用才会生效。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

sendCustomVideoFrame

sendCustomVideoFrame

int sendCustomVideoFrame (V2TXLiveVideoFrame videoFrame)

在自定义视频采集模式下，将采集的视频数据发送到SDK

在自定义视频采集模式下，SDK不再采集摄像头数据，仅保留编码和发送功能。

参数	描述
videoFrame	向 SDK 发送的 视频帧数据 V2TXLiveVideoFrame 。

注意
需要在 startPush 之前调用 enableCustomVideoCapture 开启自定义采集。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 发送失败，视频帧数据不合法。

sendCustomAudioFrame

sendCustomAudioFrame

int sendCustomAudioFrame (V2TXLiveAudioFrame audioFrame)

在自定义音频采集模式下，将采集的音频数据发送到SDK

@info 在自定义音频采集模式下，将采集的音频数据发送到SDK，SDK不再采集麦克风数据，仅保留编码和发送功能。

@param audioFrame 向 SDK 发送的 音频帧数据 V2TXLiveAudioFrame 。
@return 返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 发送失败，音频帧数据不合法。

@note 需要在 startPush 之前调用 enableCustomAudioCapture 开启自定义采集。

enableAudioProcessObserver

enableAudioProcessObserver

int enableAudioProcessObserver	(boolean enable V2TXLiveDef. V2TXLiveAudioFrameObserverFormat fo

开启/关闭对经过前处理后的本地音频帧的监听回调

参数	描述
enable	是否开启。【默认值】：false。
format	设置回调出的 AudioFrame 的格式。

注意
需要在 startPush 之前调用，才会生效。

sendSeiMessage

sendSeiMessage

int sendSeiMessage	(int payloadType byte[] data)

发送 SEI 消息

播放端 V2TXLivePlayer 通过 V2TXLivePlayerObserver 中的 `onReceiveSeiMessage` 回调来接收该消息。

参数	描述
data	待发送的数据。
payloadType	数据类型，支持 5、242。推荐填：242。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

startSystemAudioLoopback

startSystemAudioLoopback

打开系统声音采集

开启后可以采集整个操作系统的播放声音，并将其混入到当前麦克风采集的声音中一起发送到云端。

> **注意**
> 1. 该接口只在 Android API 29 及以上的版本生效。
> 2. 您需要先使用该接口来开启系统声音采集，当使用屏幕分享接口开启屏幕分享时才会真正生效。
> 3. 您需要添加一个前台服务来确保系统声音采集不被静默，并设置 android:foregroundServiceType="mediaProjection"。
> 4. SDK 只采集满足捕获策略和音频用法的应用音频，目前采集的音频用法包括USAGE_MEDIA, USAGE_GAME。

stopSystemAudioLoopback

stopSystemAudioLoopback

关闭系统声音采集

showDebugView

showDebugView

void showDebugView (boolean isShow)

显示仪表盘。

参数	描述

isShow	是否显示。【默认值】：false。

setProperty

setProperty

int setProperty (String key

Object value)

调用 V2TXLivePusher 的高级 API 接口。

参数	描述
key	高级 API 对应的 key，详情请参考 V2TXLiveProperty 定义。
value	调用 key 所对应的高级 API 时，需要的参数。

> **注意**
> 该接口用于调用一些高级功能。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，key 不允许为空。

setMixTranscodingConfig

setMixTranscodingConfig

int setMixTranscodingConfig (V2TXLiveTranscodingConfig config)

设置云端的混流转码参数

如果您在实时音视频 控制台 中的功能配置页开启了“启用旁路推流”功能，

房间里的每一路画面都会有一个默认的直播 CDN 地址 。

一个直播间中可能有不止一位主播，而且每个主播都有自己的画面和声音，但对于 CDN 观众来说，他们只需要一路直播流，
所以您需要将多路音视频流混成一路标准的直播流，这就需要混流转码。

当您调用 setMixTranscodingConfig() 接口时，SDK 会向腾讯云的转码服务器发送一条指令，目的是将房间里的多路音视频流混合为一路，
您可以通过 mixStreams 参数来调整每一路画面的位置，以及是否只混合声音，也可以通过 videoWidth、videoHeight、videoBitrate 等参数控制混合音视频流的编码参数。

【画面1】=> 解码 ====> \
\
【画面2】=> 解码 => 画面混合 => 编码 => 【混合后的画面】
/
【画面3】=> 解码 ====> /

【声音1】=> 解码 ====> \
\
【声音2】=> 解码 => 声音混合 => 编码 => 【混合后的声音】
/
【声音3】=> 解码 ====> /

参考文档： 云端混流转码 。

参数	描述
config	请参考 V2TXLiveDef.java 中关于 V2TXLiveTranscodingConfig 的介绍。如果传入 null 则取消云端混流转码。

注意

关于云端混流的注意事项：

- 仅支持 RTC 模式混流。
- 云端转码会引入一定的 CDN 观看延时，大概会增加1 – 2秒。

- 调用该函数的用户，会将连麦中的多路画面混合到自己当前这路画面或者 config 中指定的 streamId 上。
- 请注意，若您还在房间中且不再需要混流，请务必传入 null 进行取消，因为当您发起混流后，云端混流模块就会开始工作，不及时取消混流可能会引起不必要的计费损失。
- 请放心，您退房时会自动取消混流状态。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 未开启推流时，不允许设置混流转码参数。

startLocalRecording

startLocalRecording

int startLocalRecording (V2TXLiveLocalRecordingParams params)

开始录制音视频流

注意

推流开启后才能开始录制，非推流状态下开启录制无效。

- 录制过程中不要动态切换分辨率和软/硬剪辑，生成的视频极有可能出现异常。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK：成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER：参数不合法，比如filePath 为空。
- V2TXLIVE_ERROR_REFUSED：API被拒绝，推流尚未开始。

stopLocalRecording

stopLocalRecording

停止录制音视频流

注意

当停止推流后，如果视频还在录制中，SDK 内部会自动结束录制。

V2TXLivePusherObserver

最近更新时间：2024-01-16 14:35:22

Module: V2TXLivePusherObserver @ TXLiteAVSDK

Function: 腾讯云直播推流的回调通知

功能

腾讯云直播的推流回调通知。

介绍

可以接收 V2TXLivePusher 推流器的一些推流通知，包括推流器连接状态、音视频首帧回调、统计数据、警告和错误信息等。

V2TXLivePusherObserver

V2TXLivePusherObserver

函数列表	描述
onError	直播推流器错误通知，推流器出现错误时，会回调该通知
onWarning	直播推流器警告通知
onCaptureFirstAudioFrame	首帧音频采集完成的回调通知
onCaptureFirstVideoFrame	首帧视频采集完成的回调通知
onMicrophoneVolumeUpdate	麦克风采集音量值回调
onPushStatusUpdate	推流器连接状态回调通知
onStatisticsUpdate	直播推流器统计数据回调

onSnapshotComplete	截图回调
onGLContextCreated	SDK 内部的 OpenGL 环境的创建通知
onProcessAudioFrame	本地采集并经过音频模块前处理、音效处理和混 BGM 后的音频数据回调
onProcessVideoFrame	自定义视频处理回调
onGLContextDestroyed	SDK 内部的 OpenGL 环境的销毁通知
onSetMixTranscodingConfig	设置云端的混流转码参数的回调，对应于 setMixTranscodingConfig 接口
onScreenCaptureStarted	当屏幕分享开始时，SDK 会通过此回调通知
onScreenCaptureStopped	当屏幕分享停止时，SDK 会通过此回调通知
onLocalRecordBegin	录制任务开始的事件回调
onLocalRecording	录制任务正在进行中的进展事件回调
onLocalRecordComplete	录制任务已经结束的事件回调

onError

onError

```
void onError (int code
              String msg
              Bundle extraInfo)
```

直播推流器错误通知，推流器出现错误时，会回调该通知

参数	描述
code	错误码 V2TXLiveCode 。
extraInfo	扩展信息。
msg	错误信息。

onWarning

onWarning

```
void
onWarning
(int code
String msg
Bundle extraInfo)
```

直播推流器警告通知

参数	描述
code	警告码 V2TXLiveCode 。
extraInfo	扩展信息。
msg	警告信息。

onCaptureFirstAudioFrame

onCaptureFirstAudioFrame

首帧音频采集完成的回调通知

onCaptureFirstVideoFrame

onCaptureFirstVideoFrame

首帧视频采集完成的回调通知

onMicrophoneVolumeUpdate

onMicrophoneVolumeUpdate

```
void onMicrophoneVolumeUpdate (int volume)
```

麦克风采集音量值回调

参数	描述
volume	音量大小。

注意
调用 enableVolumeEvaluation 开启采集音量大小提示之后，会收到这个回调通知。

onPushStatusUpdate

onPushStatusUpdate

```
void onPushStatusUpdate (V2TXLivePushStatus status
                         String msg
                         Bundle extraInfo)
```

推流器连接状态回调通知

参数	描述
extraInfo	扩展信息。
msg	连接状态信息。

status	推流器连接状态 V2TXLivePushStatus 。

onStatisticsUpdate

onStatisticsUpdate

void onStatisticsUpdate (V2TXLivePusherStatistics statistics)

直播推流器统计数据回调

参数	描述
statistics	推流器统计数据 V2TXLivePusherStatistics

onSnapshotComplete

onSnapshotComplete

void onSnapshotComplete (Bitmap image)

截图回调

参数	描述
image	已截取的视频画面。

注意

调用 snapshot 截图之后，会收到这个回调通知。

onGLContextCreated

onGLContextCreated

SDK 内部的 OpenGL 环境的创建通知

onProcessAudioFrame

onProcessAudioFrame

```
void onProcessAudioFrame (V2TXLiveDef.V2TXLiveAudioFrame frame)
```

本地采集并经过音频模块前处理、音效处理和混 BGM 后的音频数据回调

当您设置完音频数据自定义回调之后，SDK 内部会把刚采集到并经过前处理、音效处理和混 BGM 之后的数据，在最终进行网络编码之前，以 PCM 格式的形式通过本接口回调给您。

- 此接口回调出的音频时间帧长固定为 0.02s，格式为 PCM 格式。
- 由时间帧长转化为字节帧长的公式为 `采样率 × 时间帧长 × 声道数 × 采样点位宽` 。
- 以 SDK 默认的音频录制格式 48000 采样率、单声道、16采样点位宽为例，字节帧长为 `48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节` 。

参数	描述
frame	PCM 格式的音频数据帧。

注意

1. 请不要在此回调函数中做任何耗时操作，由于 SDK 每隔 20ms 就要处理一帧音频数据，如果您的处理时间超过 20ms，就会导致声音异常。
2. 此接口回调出的音频数据是可读写的，也就是说您可以在回调函数中同步修改音频数据，但请保证处理耗时。

onProcessVideoFrame

onProcessVideoFrame

```
int onProcessVideoFrame (V2TXLiveVideoFrame srcFrame,
                         V2TXLiveVideoFrame dstFrame)
```

自定义视频处理回调

参数	描述
dstFrame	用于承载处理过的视频画面。
srcFrame	用于承载未处理的视频画面。

注意

需要调用 enableCustomVideoProcess 开启自定义视频处理，才会收到这个回调通知。

【情况一】美颜组件会产生新的纹理

如果您使用的美颜组件会在处理图像的过程中产生一帧全新的纹理（用于承载处理后的图像），那请您在回调函数中将 dstFrame.textureId 设置为新纹理的 ID。

```java
@Override
public void onGLContextCreated() {
mFURenderer.onSurfaceCreated();
mFURenderer.setUseTexAsync(true);
}
@Override
public int onProcessVideoFrame(V2TXLiveVideoFrame srcFrame,
V2TXLiveVideoFrame dstFrame) {
dstFrame.texture.textureId = mFURenderer.onDrawFrameSingleInput(
srcFrame.texture.textureId, srcFrame.width, srcFrame.height);
return 0;
}
@Override
public void onGLContextDestroyed() {
mFURenderer.onSurfaceDestroyed();
}
```

【情况二】美颜组件并不自身产生新纹理

如果您使用的第三方美颜模块并不生成新的纹理，而是需要您设置给该模块一个输入纹理和一个输出纹理，则可以考虑如下方案：

```java
int onProcessVideoFrame(V2TXLiveVideoFrame srcFrame, V2TXLiveVideoFrame
dstFrame) {
```

```
thirdparty_process(srcFrame.texture.textureId, srcFrame.width, srcFrame.height,
dstFrame.texture.textureId);
return 0;
}
```

onGLContextDestroyed

onGLContextDestroyed

SDK 内部的 OpenGL 环境的销毁通知

onSetMixTranscodingConfig

onSetMixTranscodingConfig

void onSetMixTranscodingConfig	(int code String msg)

设置云端的混流转码参数的回调，对应于 {@link setMixTranscodingConfig} 接口

参数	描述
code	0表示成功，其余值表示失败。
msg	具体错误原因。

onScreenCaptureStarted

onScreenCaptureStarted

当屏幕分享开始时，SDK 会通过此回调通知

onScreenCaptureStopped

onScreenCaptureStopped

```
void onScreenCaptureStopped(int reason)
```

当屏幕分享停止时，SDK 会通过此回调通知

参数	描述
reason	停止原因 • 0：表示用户主动停止。 • 1：iOS 表示录屏被系统中断；Mac、Windows 表示屏幕分享窗口被关闭。 • 2：Windows 表示屏幕分享的显示屏状态变更（如接口被拔出、投影模式变更等）；其他平台不抛出。

onLocalRecordBegin

onLocalRecordBegin

```
void onLocalRecordBegin(int code
                        String storagePath)
```

录制任务开始的事件回调

参数	描述
code	状态码。 • 0：录制任务启动成功。 • -1：内部错误导致录制任务启动失败。 • -2：文件后缀名有误（比如不支持的录制格式）。 • -6：录制已经启动，需要先停止录制。 • -7：录制文件已存在，需要先删除文件。 • -8：录制目录无写入权限，请检查目录权限问题。

storagePath	录制的文件地址。

onLocalRecording

onLocalRecording

```
void onLocalRecording (long durationMs
                       String storagePath)
```

录制任务正在进行中的进展事件回调

参数	描述
durationMs	录制时长。
storagePath	录制的文件地址。

onLocalRecordComplete

onLocalRecordComplete

```
void onLocalRecordComplete (int code
                            String storagePath)
```

录制任务已经结束的事件回调

参数	描述
code	状态码。 • 0：结束录制任务成功。 • -1：录制失败。 • -2：切换分辨率或横竖屏导致录制结束。 • -3：录制时间太短，或未采集到任何视频或音频数据，请检查录制时长，或是否已开启音、视频采集。

storagePath	录制的文件地址。

V2TXLivePlayerObserver

最近更新时间：2024-01-16 14:35:22

Module: V2TXLivePlayerObserver @ TXLiteAVSDK

Function: 腾讯云直播的播放器回调通知

功能

腾讯云直播的播放器回调通知。

介绍

可以接收 V2TXLivePlayer 播放器的一些回调通知，包括播放器状态、播放音量回调、音视频首帧回调、统计数据、警告和错误信息等。

V2TXLivePlayerObserver

V2TXLivePlayerObserver

函数列表	描述
onError	直播播放器错误通知，播放器出现错误时，会回调该通知
onWarning	直播播放器警告通知
onVideoResolutionChanged	直播播放器分辨率变化通知
onConnected	已经成功连接到服务器
onVideoPlaying	视频播放事件
onAudioPlaying	音频播放事件
onVideoLoading	视频加载事件
onAudioLoading	音频加载事件

onPlayoutVolumeUpdate	播放器音量大小回调
onStatisticsUpdate	直播播放器统计数据回调
onSnapshotComplete	截图回调
onRenderVideoFrame	自定义视频渲染回调
onPlayoutAudioFrame	音频数据回调
onReceiveSeiMessage	收到 SEI 消息的回调，发送端通过 V2TXLivePusher 中的 `sendSeiMessage` 来发送 SEI 消息
onStreamSwitched	分辨率无缝切换回调
onLocalRecordBegin	录制任务开始的事件回调
onLocalRecording	录制任务正在进行中的进展事件回调
onLocalRecordComplete	录制任务已经结束的事件回调

onError

onError

```
void onError (V2TXLivePlayer player
              int code
              String msg
              Bundle extraInfo)
```

直播播放器错误通知，播放器出现错误时，会回调该通知

参数	描述
code	错误码 V2TXLiveCode 。

extraInfo	扩展信息。
msg	错误信息。
player	回调该通知的播放器对象。

onWarning

onWarning

```
void
onWarning
```
(V2TXLivePlayer player

int code

String msg

Bundle extraInfo)

直播播放器警告通知

参数	描述
code	警告码 V2TXLiveCode 。
extraInfo	扩展信息。
msg	警告信息。
player	回调该通知的播放器对象。

onVideoResolutionChanged

onVideoResolutionChanged

```
void
onVideoResolutionChanged
```
(V2TXLivePlayer player

int width

int height)

直播播放器分辨率变化通知

参数	描述
height	视频高。
player	回调该通知的播放器对象。
width	视频宽。

onConnected

onConnected

```
void onConnected (V2TXLivePlayer player
                  Bundle extraInfo)
```

已经成功连接到服务器

参数	描述
extraInfo	扩展信息。
player	回调该通知的播放器对象。

onVideoPlaying

onVideoPlaying

```
void onVideoPlaying (V2TXLivePlayer player
                     boolean firstPlay
                     Bundle extraInfo)
```

视频播放事件

参数	描述
extraInfo	扩展信息。
firstPlay	第一次播放标志。
player	回调该通知的播放器对象。

onAudioPlaying

onAudioPlaying

```
void onAudioPlaying (V2TXLivePlayer player
                     boolean firstPlay
                     Bundle extraInfo)
```

音频播放事件

参数	描述
extraInfo	扩展信息。
firstPlay	第一次播放标志。
player	回调该通知的播放器对象。

onVideoLoading

onVideoLoading

```
void onVideoLoading (V2TXLivePlayer player
                     Bundle extraInfo)
```

视频加载事件

参数	描述
extraInfo	扩展信息。
player	回调该通知的播放器对象。

onAudioLoading

onAudioLoading

```
void
onAudioLoading
```
(V2TXLivePlayer player

Bundle extraInfo)

音频加载事件

参数	描述
extraInfo	扩展信息。
player	回调该通知的播放器对象。

onPlayoutVolumeUpdate

onPlayoutVolumeUpdate

```
void
onPlayoutVolumeUpdate
```
(V2TXLivePlayer player

int volume)

播放器音量大小回调

参数	描述
player	回调该通知的播放器对象。
volu	音量大小。

me	

> **注意**
> 调用 enableVolumeEvaluation 开启播放音量大小提示之后，会收到这个回调通知。

onStatisticsUpdate

onStatisticsUpdate

```
void onStatisticsUpdate (V2TXLivePlayer player
                         V2TXLivePlayerStatistics statistics)
```

直播播放器统计数据回调

参数	描述
player	回调该通知的播放器对象。
statistics	播放器统计数据 V2TXLivePlayerStatistics 。

onSnapshotComplete

onSnapshotComplete

```
void onSnapshotComplete (V2TXLivePlayer player
                         Bitmap image)
```

截图回调

参数	描述
image	已截取的视频画面。
player	回调该通知的播放器对象。

onRenderVideoFrame

onRenderVideoFrame

```
void onRenderVideoFrame(V2TXLivePlayer player, V2TXLiveVideoFrame videoFrame)
```

自定义视频渲染回调

参数	描述
player	回调该通知的播放器对象。
videoFrame	视频帧数据 V2TXLiveVideoFrame 。

注意
需要您调用 enableObserveVideoFrame 开启回调开关。

onPlayoutAudioFrame

onPlayoutAudioFrame

```
void onPlayoutAudioFrame(V2TXLivePlayer player, V2TXLiveAudioFrame audioFrame)
```

音频数据回调

参数	描述
audioFrame	音频帧数据 V2TXLiveAudioFrame 。
player	回调该通知的播放器对象。

> **注意**
> 需要您调用 enableObserveAudioFrame 开启回调开关。请在当前回调中使用 audioFrame 的 data。

onReceiveSeiMessage

onReceiveSeiMessage

```
void
onReceiveSeiMessage
```
(V2TXLivePlayer player
int payloadType
byte[] data)

收到 SEI 消息的回调，发送端通过 {@link V2TXLivePusher} 中的 `sendSeiMessage` 来发送 SEI 消息

参数	描述
data	数据。
payloadType	回调数据的SEI payloadType。
player	回调该通知的播放器对象。

> **注意**
> 调用 V2TXLivePlayer 中的 `enableReceiveSeiMessage` 开启接收 SEI 消息之后，会收到这个回调通知。

onStreamSwitched

onStreamSwitched

```
void
onStreamSwitched
```
(V2TXLivePlayer player

```
String url
int code)
```

分辨率无缝切换回调

参数	描述
code	状态码，0：成功，-1：切换超时，-2：切换失败，服务端错误，-3：切换失败，客户端错误。
player	回调该通知的播放器对象。
url	切换的播放地址。

注意

调用 V2TXLivePlayer 中的 `switchStream` 切换分辨率，会收到这个回调通知。

onLocalRecordBegin

onLocalRecordBegin

```
void onLocalRecordBegin (V2TXLivePlayer player
                         int code
                         String storagePath)
```

录制任务开始的事件回调

参数	描述
code	状态码。 • 0：录制任务启动成功。 • -1：内部错误导致录制任务启动失败。 • -2：文件后缀名有误（比如不支持的录制格式）。 • -6：录制已经启动，需要先停止录制。 • -7：录制文件已存在，需要先删除文件。

	• -8：录制目录无写入权限，请检查目录权限问题。
player	回调该通知的播放器对象。
storagePath	录制的文件地址。

onLocalRecording

onLocalRecording

```
void onLocalRecording (V2TXLivePlayer player
                       long durationMs
                       String storagePath)
```

录制任务正在进行中的进展事件回调

参数	描述
durationMs	录制时长。
player	回调该通知的播放器对象。
storagePath	录制的文件地址。

onLocalRecordComplete

onLocalRecordComplete

```
void onLocalRecordComplete (V2TXLivePlayer player
                            int code
                            String storagePath)
```

录制任务已经结束的事件回调

参数	描述
code	状态码。 • 0：结束录制任务成功。 • -1：录制失败。 • -2：切换分辨率或横竖屏导致录制结束。 • -3：录制时间太短，或未采集到任何视频或音频数据，请检查录制时长，或是否已开启音、视频采集。
player	回调该通知的播放器对象。
storagePath	录制的文件地址。

V2TXLivePlayer

最近更新时间：2024-01-16 14:35:22

Module: V2TXLivePlayer @ TXLiteAVSDK

Function: 腾讯云直播播放器

功能

腾讯云直播播放器。
主要负责从指定的直播流地址拉取音视频数据，并进行解码和本地渲染播放。

介绍

播放器包含如下能力：

- 支持 RTMP、HTTP-FLV、HLS、TRTC、WebRTC 协议。
- 屏幕截图，可以截取当前直播流的视频画面。
- 延时调节，可以设置播放器缓存自动调整的最小和最大时间。
- 自定义的视频数据处理，您可以根据项目需要处理直播流中的视频数据后，再进行渲染以及播放。

V2TXLivePlayer

V2TXLivePlayer

函数列表	描述
setObserver	设置播放器回调
setRenderView	设置播放器的视频渲染 View，该控件负责显示视频内容
setRenderView	设置播放器的视频渲染 View。该控件负责显示视频内容
setRenderView	设置播放器的视频渲染 View。该控件负责显示视频内容
setRenderRotation	设置播放器画面的旋转角度
setRenderFillMode	设置画面的填充模式

startLivePlay	开始播放音视频流
stopPlay	停止播放音视频流
isPlaying	播放器是否正在播放中
pauseAudio	暂停播放器的音频流
resumeAudio	恢复播放器的音频流
pauseVideo	暂停播放器的视频流
resumeVideo	恢复播放器的视频流
setPlayoutVolume	设置播放器音量
setCacheParams	设置播放器缓存自动调整的最小和最大时间（单位：秒）
switchStream	直播流无缝切换，支持 FLV 和 LEB
getStreamList	获取码流信息
enableVolumeEvaluation	启用播放音量大小提示
snapshot	截取播放过程中的视频画面
enableObserveVideoFrame	开启/关闭对视频帧的监听回调
enableObserveAudioFrame	开启/关闭对音频数据的监听回调
enableReceiveSeiMessage	开启接收 SEI 消息
showDebugView	是否显示播放器状态信息的调试浮层
setProperty	调用 V2TXLivePlayer 的高级 API 接口
startLocalRecording	开始录制音视频流
stopLocalRecording	停止录制音视频流

setObserver

setObserver

void setObserver （V2TXLivePlayerObserver observer)

设置播放器回调

通过设置回调，可以监听 V2TXLivePlayer 播放器的一些回调事件，
包括播放器状态、播放音量回调、音视频首帧回调、统计数据、警告和错误信息等。

参数	描述
observer	播放器的回调目标对象，更多信息请查看 V2TXLivePlayerObserver

setRenderView

setRenderView

int setRenderView （TXCloudVideoView view)

设置播放器的视频渲染 View，该控件负责显示视频内容

参数	描述
view	播放器渲染 View。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK：成功。

setRenderView

setRenderView

int setRenderView （TextureView view)

设置播放器的视频渲染 View。 该控件负责显示视频内容

参数	描述
view	播放器渲染 View。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK：成功。

setRenderView

setRenderView

int setRenderView (SurfaceView view)

设置播放器的视频渲染 View。该控件负责显示视频内容

参数	描述
view	播放器渲染 View。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK：成功。

setRenderRotation

setRenderRotation

int setRenderRotation (V2TXLiveRotation rotation)

设置播放器画面的旋转角度

参数	描述
rotation	旋转角度 V2TXLiveRotation 。 • V2TXLiveRotation0【默认值】：0度，不旋转。 • V2TXLiveRotation90：顺时针旋转90度。

	• V2TXLiveRotation180: 顺时针旋转180度。 • V2TXLiveRotation270: 顺时针旋转270度。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setRenderFillMode

setRenderFillMode

int setRenderFillMode (V2TXLiveFillMode mode)

设置画面的填充模式

参数	描述
mode	画面填充模式 V2TXLiveFillMode 。 • V2TXLiveFillModeFill【默认值】: 图像铺满屏幕，不留黑边，如果图像宽高比不同于屏幕宽高比，部分画面内容会被裁剪掉。 • V2TXLiveFillModeFit: 图像适应屏幕，保持画面完整，但如果图像宽高比不同于屏幕宽高比，会有黑边的存在。 • V2TXLiveFillModeScaleFill: 图像拉伸铺满，因此长度和宽度可能不会按比例变化。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

startLivePlay

startLivePlay

int startLivePlay (String url)

开始播放音视频流

参数	描述

url	音视频流的播放地址，支持 RTMP，HTTP-FLV，TRTC。

注意

10.7 版本开始，需要通过 setLicence 或者 setLicence 设置 Licence 后方可成功播放，否则将播放失败（黑屏），全局仅设置一次即可。直播 Licence、短视频 Licence 和视频播放 Licence 均可使用，若您暂未获取上述 Licence，可 快速免费申请测试版 Licence 以正常播放，正式版 Licence 需 购买 。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 操作成功，开始连接并播放。
- V2TXLIVE_ERROR_INVALID_PARAMETER：操作失败，url 不合法。
- V2TXLIVE_ERROR_REFUSED：RTC 不支持同一设备上同时推拉同一个 StreamId。
- V2TXLIVE_ERROR_INVALID_LICENSE：licence 不合法，播放失败。

stopPlay

stopPlay

停止播放音视频流

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

isPlaying

isPlaying

播放器是否正在播放中

返回值说明：

是否正在播放。

- 1: 正在播放中。
- 0: 已经停止播放。

pauseAudio

pauseAudio

暂停播放器的音频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

resumeAudio

resumeAudio

恢复播放器的音频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

pauseVideo

pauseVideo

暂停播放器的视频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

resumeVideo

resumeVideo

恢复播放器的视频流

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setPlayoutVolume

setPlayoutVolume

int setPlayoutVolume (int volume)

设置播放器音量

参数	描述
volume	音量大小，取值范围0 - 100。【默认值】: 100。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

setCacheParams

setCacheParams

int setCacheParams (float minTime
float maxTime)

设置播放器缓存自动调整的最小和最大时间（单位：秒）

参数	描述
maxTime	缓存自动调整的最大时间，取值需要大于0。【默认值】：5。

minTime	缓存自动调整的最小时间，取值需要大于0。【默认值】：1。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，minTime 和 maxTime 需要大于0。
- V2TXLIVE_ERROR_REFUSED: 播放器处于播放状态，不支持修改缓存策略。

switchStream

switchStream

int switchStream (String newUrl)

直播流无缝切换，支持 FLV 和 LEB

参数	描述
newUrl	新的拉流地址。

getStreamList

getStreamList

获取码流信息

enableVolumeEvaluation

enableVolumeEvaluation

int enableVolumeEvaluation (int intervalMs)

启用播放音量大小提示

开启后可以在 V2TXLivePlayerObserver 回调中获取到 SDK 对音量大小值的评估。

参数	描述
intervalMs	决定了 onPlayoutVolumeUpdate 回调的触发间隔，单位为ms，最小间隔为100ms，如果小于等于0则会关闭回调，建议设置为300ms；【默认值】：0，不开启。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。

snapshot

snapshot

截取播放过程中的视频画面

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_REFUSED: 播放器处于停止状态，不允许调用截图操作。

enableObserveVideoFrame

enableObserveVideoFrame

int enableObserveVideoFrame(boolean enable, V2TXLivePixelFormat pixelFormat, V2TXLiveBufferType bufferType)

开启/关闭对视频帧的监听回调

SDK 在您开启此开关后将不再渲染视频画面，您可以通过 V2TXLivePlayerObserver 获得视频帧，并执行自定义的渲染逻辑。

参数	描述
bufferType	自定义渲染回调的视频数据格式 V2TXLiveBufferType 。
enable	是否开启自定义渲染。【默认值】：false。
pixelFormat	自定义渲染回调的视频像素格式 V2TXLivePixelFormat 。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_NOT_SUPPORTED: 像素格式或者数据格式不支持。

enableObserveAudioFrame

enableObserveAudioFrame

int enableObserveAudioFrame (boolean enable)

开启/关闭对音频数据的监听回调

如果您开启此开关，您可以通过 V2TXLivePlayerObserver 获得音频数据，并执行自定义的逻辑。

参数	描述
enable	是否开启音频数据回调。【默认值】：false。

返回值说明：

返回值 V2TXLiveCode

- V2TXLIVE_OK: 成功

enableReceiveSeiMessage

enableReceiveSeiMessage

int enableReceiveSeiMessage (boolean enable

int payloadType)

开启接收 SEI 消息

参数	描述
enable	true: 开启接收 SEI 消息; false: 关闭接收 SEI 消息。【默认值】: false。
payloadType	指定接收 SEI 消息的 payloadType，支持 5、242，请与发送端的 payloadType 保持一致。

返回值说明:

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。

showDebugView

showDebugView

```
void showDebugView (boolean isShow)
```

是否显示播放器状态信息的调试浮层

参数	描述
isShow	是否显示。【默认值】: false。

setProperty

setProperty

```
int setProperty (String key
                 Object value)
```

调用 V2TXLivePlayer 的高级 API 接口

参数	描述
key	高级 API 对应的 key。
value	调用 key 所对应的高级 API 时，需要的参数。

注意
该接口用于调用一些高级功能。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，key 不允许为 null。

startLocalRecording

startLocalRecording

int startLocalRecording (V2TXLiveLocalRecordingParams params)

开始录制音视频流

注意
拉流开启后才能开始录制，非拉流状态下开启录制无效。
- 录制过程中不要动态切换软/硬解，生成的视频极有可能出现异常。

返回值说明：

返回值 V2TXLiveCode 。
- V2TXLIVE_OK：成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER：参数不合法，比如filePath 为空。
- V2TXLIVE_ERROR_REFUSED：API被拒绝，拉流尚未开始。

stopLocalRecording

stopLocalRecording

停止录制音视频流

> **注意**
> 当停止拉流后，如果视频还在录制中，SDK 内部会自动结束录制。

V2TXLivePremier

最近更新时间：2024-03-07 15:14:11

Module: V2TXLivePremier @ TXLiteAVSDK

Function: V2TXLive 高级接口

V2TXLivePremier

V2TXLivePremierObserver

函数列表	描述
onLog	自定义 Log 输出回调接口
onLicenceLoaded	setLicence 接口回调
onCaptureAudioFrame	本地麦克风采集到的音频数据回调
onPlayoutAudioFrame	将各路待播放音频混合之后并在最终提交系统播放之前的数据回调
onVoiceEarMonitorAudioFrame	耳返的音频数据

V2TXLivePremier

函数列表	描述
getSDKVersionStr	获取 SDK 版本号
setObserver	设置 V2TXLivePremier 回调接口
setLogConfig	设置 Log 的配置信息
setEnvironment	设置 SDK 接入环境
setLicence	设置 SDK 的授权 License

setSocks5Proxy	设置 SDK socks5 代理配置
enableAudioCaptureObserver	开启/关闭对音频采集数据的监听回调（可读写）
enableAudioPlayoutObserver	开启/关闭对最终系统要播放出的音频数据的监听回调
enableVoiceEarMonitorObserver	开启/关闭耳返音频数据的监听回调
setUserId	设置 userId
callExperimentalAPI	调用实验性 API 接口

onLog

onLog

```
void onLog (int level, String log)
```

自定义 Log 输出回调接口

onLicenceLoaded

onLicenceLoaded

```
void onLicenceLoaded (int result, String reason)
```

setLicence 接口回调

参数	描述
reason	设置 licence 失败原因。

result	设置 licence 结果 0 成功，负数失败。

onCaptureAudioFrame

onCaptureAudioFrame

void onCaptureAudioFrame (V2TXLiveDef.V2TXLiveAudioFrame frame)

本地麦克风采集到的音频数据回调

参数	描述
frame	音频数据。

注意

- 请不要在此回调函数中做任何耗时操作，建议直接拷贝到另一线程进行处理，否则会导致各种声音问题。
- 此接口回调出的音频数据支持修改。
- 此接口回调出的音频时间帧长固定为0.02s。

由时间帧长转化为字节帧长的公式为【采样率 × 时间帧长 × 声道数 × 采样点位宽】。
以SDK默认的音频录制格式48000采样率、单声道、16采样点位宽为例，字节帧长为【48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节】。

- 此接口回调出的音频数据**不包含**背景音、音效、混响等前处理效果，延迟极低。
- 需要您调用 enableAudioCaptureObserver 开启回调开关。

onPlayoutAudioFrame

onPlayoutAudioFrame

void onPlayoutAudioFrame (V2TXLiveDef.V2TXLiveAudioFrame frame)

将各路待播放音频混合之后并在最终提交系统播放之前的数据回调

当您设置完音频数据自定义回调之后，SDK 内部会把各路待播放的音频混合之后的音频数据，在提交系统播放之前，以 PCM 格式的形式通过本接口回调给您。

- 此接口回调出的音频时间帧长固定为 0.02s，格式为 PCM 格式。
- 由时间帧长转化为字节帧长的公式为 `采样率 × 时间帧长 × 声道数 × 采样点位宽` 。
- 以 SDK 默认的音频录制格式 48000 采样率、单声道、16 采样点位宽为例，字节帧长为 `48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节` 。

参数	描述
frame	PCM 格式的音频数据帧。

注意

1. 请不要在此回调函数中做任何耗时操作，由于 SDK 每隔 20ms 就要处理一帧音频数据，如果您的处理时间超过 20ms，就会导致声音异常。
2. 此接口回调出的音频数据是可读写的，也就是说您可以在回调函数中同步修改音频数据，但请保证处理耗时。
3. 此接口回调出的是对各路待播放音频数据的混合，但其中并不包含耳返的音频数据。

onVoiceEarMonitorAudioFrame

onVoiceEarMonitorAudioFrame

```
void onVoiceEarMonitorAudioFrame (V2TXLiveDef.V2TXLiveAudioFrame frame)
```

耳返的音频数据

当您设置完音频数据自定义回调之后，SDK 内部会把耳返的音频数据在播放之前以 PCM 格式的形式通过本接口回调给您。

- 此接口回调出的音频时间帧长不固定，格式为 PCM 格式。
- 由时间帧长转化为字节帧长的公式为 `采样率 × 时间帧长 × 声道数 × 采样点位宽` 。
- 以 TRTC 默认的音频录制格式 48000 采样率、单声道、16采样点位宽为例，0.02s 的音频数据字节帧长为 `48000 × 0.02s × 1 × 16bit = 15360bit = 1920字节` 。

参数	描述
frame	PCM 格式的音频数据帧。

注意

1. 请不要在此回调函数中做任何耗时操作，否则会导致声音异常。
2. 此接口回调出的音频数据是可读写的，也就是说您可以在回调函数中同步修改音频数据，但请保证处理耗时。

getSDKVersionStr

getSDKVersionStr

获取 SDK 版本号

setObserver

setObserver

void setObserver (V2TXLivePremierObserver observer)

设置 V2TXLivePremier 回调接口

setLogConfig

setLogConfig

void setLogConfig (V2TXLiveDef.V2TXLiveLogConfig config)

设置 Log 的配置信息

setEnvironment

setEnvironment

void setEnvironment (String env)

设置 SDK 接入环境

参数	描述
env	目前支持 “default” 和 “GDPR” 两个参数。 • default：默认环境，SDK 会在全球寻找最佳接入点进行接入。 • GDPR：所有音视频数据和质量统计数据都不会经过中国大陆地区的服务器。

注意

如您的应用无特殊需求，请不要调用此接口进行设置。

setLicence

setLicence

void setLicence (Context context
String url
String key)

设置 SDK 的授权 License

文档地址：https://cloud.tencent.com/document/product/454/34750。

参数	描述
context	
key	licence的秘钥。
url	licence的地址。

setSocks5Proxy

setSocks5Proxy

```
void
setSocks5Proxy
(String host
int port
String username
String password
V2TXLiveDef.V2TXLiveSocks5ProxyConfig config)
```

设置 SDK socks5 代理配置

参数	描述
config	配置使用 socks5 代理服务器的协议。
host	socks5 代理服务器的地址。
password	socks5 代理服务器的验证的密码。
port	socks5 代理服务器的端口。
username	socks5 代理服务器的验证的用户名。

enableAudioCaptureObserver

enableAudioCaptureObserver

```
void
enableAudioCaptureObserver
(boolean enable
V2TXLiveDef.V2TXLiveAudioFrameObserverFormat
```

开启/关闭对音频采集数据的监听回调（可读写）

参数	描述
enable	是否开启。【默认值】：false。
format	设置回调出的 AudioFrame 的格式。

注意
需要在 startPush 之前调用，才会生效。

enableAudioPlayoutObserver

enableAudioPlayoutObserver

```
void enableAudioPlayoutObserver (boolean enable
                                 V2TXLiveDef.V2TXLiveAudioFrameObserverFormat
```

开启/关闭对最终系统要播放出的音频数据的监听回调

参数	描述
enable	是否开启。【默认值】：false。
format	设置回调出的 AudioFrame 的格式。

enableVoiceEarMonitorObserver

enableVoiceEarMonitorObserver

```
void enableVoiceEarMonitorObserver (boolean enable)
```

开启/关闭耳返音频数据的监听回调

参数	描述

enable	是否开启。【默认值】：false。

setUserId

setUserId

void setUserId (String userId)

设置 userId

参数	描述
userId	业务侧自身维护的用户/设备id。

callExperimentalAPI

callExperimentalAPI

int callExperimentalAPI (String jsonStr)

调用实验性 API 接口

参数	描述
jsonStr	接口及参数描述的 JSON 字符串。

注意

该接口用于调用一些实验性功能。

返回值说明：

返回值 V2TXLiveCode 。

- V2TXLIVE_OK: 成功。
- V2TXLIVE_ERROR_INVALID_PARAMETER: 操作失败，参数非法。

TXAudioEffectManager

最近更新时间：2024-01-16 14:35:22

Module: 背景音乐、短音效和人声特效的管理类

Function: 用于对背景音乐、短音效和人声特效进行设置的管理类

TXAudioEffectManager

TXMusicPreloadObserver

函数列表	描述
onLoadProgress	背景音乐预加载进度
onLoadError	背景音乐预加载出错

TXMusicPlayObserver

函数列表	描述
onStart	背景音乐开始播放
onPlayProgress	背景音乐的播放进度
onComplete	背景音乐已经播放完毕

TXAudioEffectManager

函数列表	描述
enableVoiceEarMonitor	开启耳返
setVoiceEarMonitor	设置耳返音量

Volume	
setVoiceReverbType	设置人声的混响效果
setVoiceChangerType	设置人声的变声特效
setVoiceCaptureVolume	设置语音音量
setVoicePitch	设置语音音调
setMusicObserver	设置背景音乐的事件回调接口
startPlayMusic	开始播放背景音乐
stopPlayMusic	停止播放背景音乐
pausePlayMusic	暂停播放背景音乐
resumePlayMusic	恢复播放背景音乐
setAllMusicVolume	设置所有背景音乐的本地音量和远端音量的大小
setMusicPublishVolume	设置某一首背景音乐的远端音量的大小
setMusicPlayoutVolume	设置某一首背景音乐的本地音量的大小
setMusicPitch	调整背景音乐的音调高低
setMusicSpeedRate	调整背景音乐的变速效果
getMusicCurrentPosInMS	获取背景音乐的播放进度（单位：毫秒）
getMusicDurationInMS	获取背景音乐的总时长（单位：毫秒）
seekMusicToPosInMS	设置背景音乐的播放进度（单位：毫秒）
setMusicScratchSpeedRate	调整搓碟的变速效果
setPreloadObserver	设置预加载事件回调

preloadMusic	预加载背景音乐
getMusicTrackCount	获取背景音乐的音轨数量
setMusicTrack	指定背景音乐的播放音轨

结构体类型

函数列表	描述
AudioMusicParam	背景音乐的播放控制信息

枚举类型

枚举类型	描述
TXVoiceReverbType	混响特效
TXVoiceChangerType	变声特效

onLoadProgress

onLoadProgress

```
void onLoadProgress (int id
                     int progress)
```

背景音乐预加载进度

onLoadError

onLoadError

```
void (int id
```

onLoadError

int errorCode)

背景音乐预加载出错

参数	描述
errorCode	-4001:打开文件失败，如音频文件格式不支持，本地音频文件不存在，网络音频文件无法访问等；-4002:解码失败，如音频文件损坏，网络音频文件服务器无法访问等；-4003:预加载数量超上限，请先调用 stopPlayMusic 释放无用的预加载。

onStart

onStart

void onStart (int id

int errCode)

背景音乐开始播放

在背景音乐开始播放成功或者失败时调用。

参数	描述
errCode	错误码。0: 开始播放成功；-4001: 打开文件失败，如音频文件格式不支持，本地音频文件不存在，网络音频文件无法访问等。
id	音乐 ID。

onPlayProgress

onPlayProgress

void onPlayProgress (int id

long curPtsMS

long durationMS)

背景音乐的播放进度

onComplete

onComplete

void onComplete (int id int errCode)

背景音乐已经播放完毕

在背景音乐播放完毕或播放错误时调用。

参数	描述
errCode	错误码。0: 播放结束；-4002: 解码失败，如音频文件损坏，网络音频文件服务器无法访问等。
id	音乐 ID。

enableVoiceEarMonitor

enableVoiceEarMonitor

void enableVoiceEarMonitor (boolean enable)

开启耳返

主播开启耳返后，可以在耳机里听到麦克风采集到的自己发出的声音，该特效适用于主播唱歌的应用场景中。

需要您注意的是，由于蓝牙耳机的硬件延迟非常高，所以在主播佩戴蓝牙耳机时无法开启此特效，请尽量在用户界面上提示主播佩戴有线耳机。

同时也需要注意，并非所有的手机开启此特效后都能达到优秀的耳返效果，我们已经对部分耳返效果不佳的手机屏蔽了该特效。

参数	描述
enable	true：开启；false：关闭。

注意
仅在主播佩戴耳机时才能开启此特效，同时请您提示主播佩戴有线耳机。

setVoiceEarMonitorVolume

setVoiceEarMonitorVolume

void setVoiceEarMonitorVolume (int volume)

设置耳返音量

通过该接口您可以设置耳返特效中声音的音量大小。

参数	描述
volume	音量大小，取值范围为 0 – 100，默认值：100。

注意
如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setVoiceReverbType

setVoiceReverbType

void setVoiceReverbType (TXVoiceReverbType type)

设置人声的混响效果

通过该接口您可以设置人声的混响效果，具体特效请参见枚举定义 TXVoiceReverbType 。

> **注意**
> 设置的效果在退出房间后会自动失效，如果下次进房还需要对应特效，需要调用此接口再次进行设置。

setVoiceChangerType

setVoiceChangerType

```
void setVoiceChangerType (TXVoiceChangerType type)
```

设置人声的变声特效

通过该接口您可以设置人声的变声特效，具体特效请参见枚举定义 TXVoiceChangeType 。

> **注意**
> 设置的效果在退出房间后会自动失效，如果下次进房还需要对应特效，需要调用此接口再次进行设置。

setVoiceCaptureVolume

setVoiceCaptureVolume

```
void setVoiceCaptureVolume (int volume)
```

设置语音音量

该接口可以设置语音音量的大小，一般配合音乐音量的设置接口 setAllMusicVolume 协同使用，用于调谐语音和音乐在混音前各自的音量占比。

参数	描述
volu	音量大小，取值范围为0 - 100，默认值：100。

me	

注意

如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setVoicePitch

setVoicePitch

```
void setVoicePitch (double pitch)
```

设置语音音调

该接口可以设置语音音调，用于实现变调不变速的目的。

参数	描述
pitch	音调，取值范围为-1.0f~1.0f，默认值：0.0f。

setMusicObserver

setMusicObserver

```
void setMusicObserver (int id
                       TXMusicPlayObserver observer)
```

设置背景音乐的事件回调接口

请在播放背景音乐之前使用该接口设置播放事件回调，以便感知背景音乐的播放进度。

参数	描述
musicId	音乐 ID。

observer	具体参考 ITXMusicPlayObserver 中定义接口。

注意

1. 如果该 ID 不需要使用，observer 可设置为 NULL 彻底释放。

startPlayMusic

startPlayMusic

boolean startPlayMusic (final AudioMusicParam musicParam)

开始播放背景音乐

每个音乐都需要您指定具体的 ID，您可以通过该 ID 对音乐的开始、停止、音量等进行设置。

参数	描述
musicParam	音乐参数。

注意

1. 如果要多次播放同一首背景音乐，请不要每次播放都分配一个新的 ID，我们推荐使用相同的 ID。
2. 若您希望同时播放多首不同的音乐，请为不同的音乐分配不同的 ID 进行播放。
3. 如果使用同一个 ID 播放不同音乐，SDK 会先停止播放旧的音乐，再播放新的音乐。

stopPlayMusic

stopPlayMusic

void stopPlayMusic (int id)

停止播放背景音乐

参数	描述

id	音乐 ID。

pausePlayMusic

pausePlayMusic

void pausePlayMusic (int id)

暂停播放背景音乐

参数	描述
id	音乐 ID。

resumePlayMusic

resumePlayMusic

void resumePlayMusic (int id)

恢复播放背景音乐

参数	描述
id	音乐 ID。

setAllMusicVolume

setAllMusicVolume

void setAllMusicVolume (int volume)

设置所有背景音乐的本地音量和远端音量的大小

该接口可以设置所有背景音乐的本地音量和远端音量。

- 本地音量：即主播本地可以听到的背景音乐的音量大小。
- 远端音量：即观众端可以听到的背景音乐的音量大小。

参数	描述
volume	音量大小，取值范围为0 - 100，默认值：60。

注意
如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setMusicPublishVolume

setMusicPublishVolume

```
void setMusicPublishVolume(int id, int volume)
```

设置某一首背景音乐的远端音量的大小

该接口可以细粒度地控制每一首背景音乐的远端音量，也就是观众端可听到的背景音乐的音量大小。

参数	描述
id	音乐 ID。
volume	音量大小，取值范围为0 - 100；默认值：60。

注意
如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setMusicPlayoutVolume

setMusicPlayoutVolume

```
void setMusicPlayoutVolume(int id, int volume)
```

设置某一首背景音乐的本地音量的大小

该接口可以细粒度地控制每一首背景音乐的本地音量，也就是主播本地可以听到的背景音乐的音量大小。

参数	描述
id	音乐 ID。
volume	音量大小，取值范围为0 - 100，默认值：60。

注意
如果将 volume 设置成 100 之后感觉音量还是太小，可以将 volume 最大设置成 150，但超过 100 的 volume 会有爆音的风险，请谨慎操作。

setMusicPitch

setMusicPitch

```
void setMusicPitch(int id, float pitch)
```

调整背景音乐的音调高低

参数	描述
id	音乐 ID。

pitch	音调，默认值是0.0f，范围是：[-1 ~ 1] 之间的浮点数。

setMusicSpeedRate

setMusicSpeedRate

```
void setMusicSpeedRate(int id, float speedRate)
```

调整背景音乐的变速效果

参数	描述
id	音乐 ID。
speedRate	速度，默认值是1.0f，范围是：[0.5 ~ 2] 之间的浮点数。

getMusicCurrentPosInMS

getMusicCurrentPosInMS

```
long getMusicCurrentPosInMS(int id)
```

获取背景音乐的播放进度（单位：毫秒）

参数	描述
id	音乐 ID。

返回值说明：

成功返回当前播放时间，单位：毫秒，失败返回 -1。

getMusicDurationInMS

getMusicDurationInMS

long getMusicDurationInMS (String path)

获取背景音乐的总时长（单位：毫秒）

参数	描述
path	音乐文件路径。

返回值说明：

成功返回时长，失败返回 -1。

seekMusicToPosInMS

seekMusicToPosInMS

void seekMusicToPosInMS (int id

int pts)

设置背景音乐的播放进度（单位：毫秒）

参数	描述
id	音乐 ID。
pts	单位: 毫秒。

注意

请尽量避免过度频繁地调用该接口，因为该接口可能会再次读写音乐文件，耗时稍高。
因此，当用户拖拽音乐的播放进度条时，请在用户完成拖拽操作后再调用本接口。
因为 UI 上的进度条控件往往会以很高的频率反馈用户的拖拽进度，如不做频率限制，会导致较差的用户体验。

setMusicScratchSpeedRate

setMusicScratchSpeedRate

```
void
setMusicScratchSpeedRate
(int id
float scratchSpeedRate)
```

调整搓碟的变速效果

参数	描述
id	音乐 ID。
scratchSpeedRate	搓碟速度，默认值是1.0f，范围是：[-12.0 ~ 12.0] 之间的浮点数，速度值正/负表示方向正/反，绝对值大小表示速度快慢。

注意
前置条件 preloadMusic 成功。

setPreloadObserver

setPreloadObserver

```
void
setPreloadObserver
(TXMusicPreloadObserver observer)
```

设置预加载事件回调

请在预加载背景音乐之前使用该接口设置回调，以便感知背景音乐的预加载进度。

参数	描述
observer	具体参考 ITXMusicPreloadObserver 中定义接口。

preloadMusic

preloadMusic

boolean preloadMusic (final AudioMusicParam preloadParam)

预加载背景音乐

每个音乐都需要您指定具体的 ID，您可以通过该 ID 对音乐的开始、停止、音量等进行设置。

参数	描述
preloadParam	预加载音乐参数。

注意

1. 预先加载最多同时支持2个不同 ID 的预加载，且预加载时长不超过10分钟，使用完需 stopPlayMusic，否则内存不释放。
2. 若该ID对应的音乐正在播放中，预加载会失败，需先调用 stopPlayMusic。
3. 当 musicParam 和传入 startPlayMusic 的 musicParam 完全相同时，预加载有效。

getMusicTrackCount

getMusicTrackCount

int getMusicTrackCount (int id)

获取背景音乐的音轨数量

参数	描述
id	音乐 ID。

setMusicTrack

setMusicTrack

void setMusicTrack	(int id int trackIndex)

指定背景音乐的播放音轨

参数	描述
id	音乐 ID。
index	默认播放第一个音轨。取值范围[0, 音轨总数)。

注意
音轨总数量可通过 getMusicTrackCount 接口获取。

TXVoiceReverbType

TXVoiceReverbType

混响特效

混响特效可以作用于人声之上，通过声学算法对声音进行叠加处理，模拟出各种不同环境下的临场感受，目前支持如下几种混响效果：
0：关闭；1：KTV；2：小房间；3：大会堂；4：低沉；5：洪亮；6：金属声；7：磁性；8：空灵；9：录音棚；10：悠扬 11：录音棚2。

枚举	取值	描述
TXLiveVoiceReverbType_0	0	关闭特效
TXLiveVoiceReverbType_1	1	KTV
TXLiveVoiceReverbType_2	2	小房间
TXLiveVoiceReverbType_3	3	大会堂

TXLiveVoiceReverbType_4	4	低沉
TXLiveVoiceReverbType_5	5	洪亮
TXLiveVoiceReverbType_6	6	金属声
TXLiveVoiceReverbType_7	7	磁性
TXLiveVoiceReverbType_8	8	空灵
TXLiveVoiceReverbType_9	9	录音棚
TXLiveVoiceReverbType_10	10	悠扬
TXLiveVoiceReverbType_11	11	录音棚2

TXVoiceChangeType

TXVoiceChangeType

变声特效

变声特效可以作用于人声之上，通过声学算法对人声进行二次处理，以获得与原始声音所不同的音色，目前支持如下几种变声特效：
0：关闭；1：熊孩子；2：萝莉；3：大叔；4：重金属；5：感冒；6：外语腔；7：困兽；8：肥宅；9：强电流；10：重机械；11：空灵。

枚举	取值	描述
TXLiveVoiceChangerType_0	0	关闭
TXLiveVoiceChangerType_1	1	熊孩子
TXLiveVoiceChangerTy	2	萝莉

pe_2		
TXLiveVoiceChangerType_3	3	大叔
TXLiveVoiceChangerType_4	4	重金属
TXLiveVoiceChangerType_5	5	感冒
TXLiveVoiceChangerType_6	6	外语腔
TXLiveVoiceChangerType_7	7	困兽
TXLiveVoiceChangerType_8	8	肥宅
TXLiveVoiceChangerType_9	9	强电流
TXLiveVoiceChangerType_10	10	重机械
TXLiveVoiceChangerType_11	11	空灵

TXAudioMusicParam

TXAudioMusicParam

背景音乐的播放控制信息

该信息用于在接口 startPlayMusic 中指定背景音乐的相关信息，包括播放 ID、文件路径和循环次数等：
1. 如果要多次播放同一首背景音乐，请不要每次播放都分配一个新的 ID，我们推荐使用相同的 ID。
2. 若您希望同时播放多首不同的音乐，请为不同的音乐分配不同的 ID 进行播放。
3. 如果使用同一个 ID 播放不同音乐，SDK 会先停止播放旧的音乐，再播放新的音乐。

枚举类型	描述
endTimeMS	【字段含义】音乐结束播放时间点，单位毫秒，0表示播放至文件结尾。

id	【字段含义】音乐 ID。 【特殊说明】SDK 允许播放多路音乐，因此需要使用 ID 进行标记，用于控制音乐的开始、停止、音量等。
isShortFile	【字段含义】播放的是否为短音乐文件。 【推荐取值】true：需要重复播放的短音乐文件；false：正常的音乐文件。默认值：false。
loopCount	【字段含义】音乐循环播放的次数。 【推荐取值】取值范围为0 - 任意正整数，默认值：0。0 表示播放音乐一次；1 表示播放音乐两次；以此类推。
path	【字段含义】音效文件的完整路径或 URL 地址。支持的音频格式包括 MP3、AAC、M4A、WAV。
publish	【字段含义】是否将音乐传到远端。 【推荐取值】true：音乐在本地播放的同时，远端用户也能听到该音乐；false：主播只能在本地听到该音乐，远端观众听不到。默认值：false。
startTimeMS	【字段含义】音乐开始播放时间点，单位：毫秒。

TXBeautyManager

最近更新时间：2024-03-07 15:14:11

Module: 美颜与图像处理参数设置类

Function: 修改美颜、滤镜、绿幕等参数

TXBeautyManager

TXBeautyManager

函数列表	描述
setBeautyStyle	设置美颜（磨皮）算法
setBeautyLevel	设置美颜级别
setWhitenessLevel	设置美白级别
enableSharpnessEnhancement	开启清晰度增强
setRuddyLevel	设置红润级别
setFilter	设置色彩滤镜效果
setFilterStrength	设置色彩滤镜的强度
setGreenScreenFile	设置绿幕背景视频
setEyeScaleLevel	设置大眼级别
setFaceSlimLevel	设置瘦脸级别
setFaceVLevel	设置 V 脸级别
setChinLevel	设置下巴拉伸或收缩
setFaceShortLevel	设置短脸级别
setFaceNarrowLevel	设置窄脸级别

setNoseSlimLevel	设置瘦鼻级别
setEyeLightenLevel	设置亮眼级别
setToothWhitenLevel	设置牙齿美白级别
setWrinkleRemoveLevel	设置祛皱级别
setPounchRemoveLevel	设置祛眼袋级别
setSmileLinesRemoveLevel	设置法令纹去除级别
setForeheadLevel	设置发际线调整级别
setEyeDistanceLevel	设置眼距
setEyeAngleLevel	设置眼角调整级别
setMouthShapeLevel	设置嘴型调整级别
setNoseWingLevel	设置鼻翼调整级别
setNosePositionLevel	设置鼻子位置
setLipsThicknessLevel	设置嘴唇厚度
setFaceBeautyLevel	设置脸型
setMotionTmpl	选择 AI 动效挂件
setMotionMute	是否在动效素材播放时静音

枚举类型

枚举类型	描述
TXBeautyStyle	美颜（磨皮）算法

setBeautyStyle

setBeautyStyle

void setBeautyStyle	(int beautyStyle)

设置美颜（磨皮）算法

TRTC 内置多种不同的磨皮算法，您可以选择最适合您产品定位的方案：

参数	描述
beautyStyle	美颜风格，TXBeautyStyleSmooth：光滑；TXBeautyStyleNature：自然；TXBeautyStylePitu：优图。

setBeautyLevel

setBeautyLevel

void setBeautyLevel	(float beautyLevel)

设置美颜级别

参数	描述
beautyLevel	美颜级别，取值范围 0 - 9； 0 表示关闭，9 表示效果最明显。

setWhitenessLevel

setWhitenessLevel

void setWhitenessLevel	(float whitenessLevel)

设置美白级别

参数	描述
whitenessLevel	美白级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

enableSharpnessEnhancement

enableSharpnessEnhancement

void enableSharpnessEnhancement	(boolean enable)

开启清晰度增强

setRuddyLevel

setRuddyLevel

void setRuddyLevel	(float ruddyLevel)

设置红润级别

参数	描述
ruddyLevel	红润级别，取值范围0 - 9；0 表示关闭，9 表示效果最明显。

setFilter

setFilter

void setFilter	(Bitmap image)

设置色彩滤镜效果

色彩滤镜，是一副包含色彩映射关系的颜色查找表图片，您可以在我们提供的官方 Demo 中找到预先准备好的几张滤镜图片。
SDK 会根据该查找表中的映射关系，对摄像头采集出的原始视频画面进行二次处理，以达到预期的滤镜效果。

参数	描述
image	包含色彩映射关系的颜色查找表图片，必须是 png 格式。

setFilterStrength

setFilterStrength

void setFilterStrength (float strength)

设置色彩滤镜的强度

该数值越高，色彩滤镜的作用强度越明显，经过滤镜处理后的视频画面跟原画面的颜色差异越大。
我默认的滤镜浓度是 0.5，如果您觉得默认的滤镜效果不明显，可以设置为 0.5 以上的数字，最大值为 1。

参数	描述
strength	从 0 到 1，数值越大滤镜效果越明显，默认值为 0.5。

setGreenScreenFile

setGreenScreenFile

int setGreenScreenFile (String path)

设置绿幕背景视频

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见 腾讯美颜特效SDK ）中生效。此接口所开启的绿幕功能不具备智能去除背景的能力，需要被拍摄者的背后有一块绿色的幕布来辅助产生特效。

参数	描述

path	MP4格式的视频文件路径; 设置空值表示关闭特效。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeScaleLevel

setEyeScaleLevel

int setEyeScaleLevel (float eyeScaleLevel)

设置大眼级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeScaleLevel	大眼级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceSlimLevel

setFaceSlimLevel

int setFaceSlimLevel (float faceSlimLevel)

设置瘦脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述

faceSlimLevel	瘦脸级别，取值范围0 – 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceVLevel

setFaceVLevel

int setFaceVLevel (float faceVLevel)

设置 V 脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
faceVLevel	V 脸级别，取值范围 0 – 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setChinLevel

setChinLevel

int setChinLevel (float chinLevel)

设置下巴拉伸或收缩

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
chinLevel	下巴拉伸或收缩级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示收缩，大于 0 表示拉伸。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceShortLevel

setFaceShortLevel

```
int setFaceShortLevel(float faceShortLevel)
```

设置短脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
faceShortLevel	短脸级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceNarrowLevel

setFaceNarrowLevel

```
int setFaceNarrowLevel(float faceNarrowLevel)
```

设置窄脸级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
level	窄脸级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setNoseSlimLevel

setNoseSlimLevel

```
int setNoseSlimLevel (float noseSlimLevel)
```

设置瘦鼻级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
noseSlimLevel	瘦鼻级别，取值范围0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeLightenLevel

setEyeLightenLevel

```
int setEyeLightenLevel (float eyeLightenLevel)
```

设置亮眼级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeLightenLevel	亮眼级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setToothWhitenLevel

setToothWhitenLevel

int setToothWhitenLevel (float toothWhitenLevel)

设置牙齿美白级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
toothWhitenLevel	白牙级别，取值范围 0 - 9；0表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setWrinkleRemoveLevel

setWrinkleRemoveLevel

int setWrinkleRemoveLevel (float wrinkleRemoveLevel)

设置祛皱级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
wrinkleRemoveLevel	祛皱级别，取值范围0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setPounchRemoveLevel

setPounchRemoveLevel

int setPounchRemoveLevel (float pounchRemoveLevel)

设置祛眼袋级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
pounchRemoveLevel	祛眼袋级别，取值范围 0 - 9；0 表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setSmileLinesRemoveLevel

setSmileLinesRemoveLevel

int setSmileLinesRemoveLevel (float smileLinesRemoveLevel)

设置法令纹去除级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
smileLinesRemoveLevel	法令纹级别，取值范围 0 - 9；0表示关闭，9 表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setForeheadLevel

setForeheadLevel

int setForeheadLevel (float foreheadLevel)

设置发际线调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
foreheadLevel	发际线级别，取值范围-9 – 9；0表示关闭，9表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeDistanceLevel

setEyeDistanceLevel

int setEyeDistanceLevel (float eyeDistanceLevel)

设置眼距

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeDistanceLevel	眼距级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setEyeAngleLevel

setEyeAngleLevel

int setEyeAngleLevel (float eyeAngleLevel)

设置眼角调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
eyeAngleLevel	眼角调整级别，取值范围-9 – 9；0表示关闭，9表示效果最明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setMouthShapeLevel

setMouthShapeLevel

int setMouthShapeLevel (float mouthShapeLevel)

设置嘴型调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
mouthShapeLevel	嘴型级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setNoseWingLevel

setNoseWingLevel

int setNoseWingLevel (float noseWingLevel)

设置鼻翼调整级别

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
noseWingLevel	鼻翼调整级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setNosePositionLevel

setNosePositionLevel

int setNosePositionLevel (float nosePositionLevel)

设置鼻子位置

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
nosePositionLevel	鼻子位置级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示抬高，大于 0 表示降低。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setLipsThicknessLevel

setLipsThicknessLevel

int setLipsThicknessLevel (float lipsThicknessLevel)

设置嘴唇厚度

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
lipsThicknessLevel	嘴唇厚度级别，取值范围 -9 – 9；0 表示关闭，小于 0 表示拉伸，大于 0 表示收缩。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setFaceBeautyLevel

setFaceBeautyLevel

int setFaceBeautyLevel (float faceBeautyLevel)

设置脸型

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
faceBeautyLevel	美型级别，取值范围 0 – 9；0 表示关闭，1 – 9 值越大，效果越明显。

返回值说明：

0：成功；-5：当前 License 对应 feature 不支持。

setMotionTmpl

setMotionTmpl

void setMotionTmpl	(String tmplPath)

选择 AI 动效挂件

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。

参数	描述
tmplPath	动效素材文件所在目录。

setMotionMute

setMotionMute

void setMotionMute	(boolean motionMute)

是否在动效素材播放时静音

该接口仅在 企业版 SDK（旧版已下线，新版本SDK如需使用高级美颜功能请参见腾讯美颜特效SDK）中生效。
有些挂件本身会有声音特效，通过此 API 可以关闭这些特效播放时所带的声音效果。

参数	描述
motionMute	true：静音；false：不静音。

TXBeautyStyle

TXBeautyStyle

美颜（磨皮）算法

TRTC 内置多种不同的磨皮算法，您可以选择最适合您产品定位的方案。

枚举	取值	描述
TXBeautyStyleSmooth	0	光滑，算法比较激进，磨皮效果比较明显，适用于秀场直播。
TXBeautyStyleNature	1	自然，算法更多地保留了面部细节，磨皮效果更加自然，适用于绝大多数直播场景。
TXBeautyStylePitu	2	优图，由优图实验室提供，磨皮效果介于光滑和自然之间，比光滑保留更多皮肤细节，比自然磨皮程度更高。

TXDeviceManager

最近更新时间：2024-03-07 15:14:11

Module: 音视频设备管理模块

Function: 用于管理摄像头、麦克风和扬声器等音视频相关的硬件设备

TXDeviceManager

TXDeviceManager

函数列表	描述
isFrontCamera	判断当前是否为前置摄像头（仅适用于移动端）
switchCamera	切换前置或后置摄像头（仅适用于移动端）
getCameraZoomMaxRatio	获取摄像头的最大缩放倍数（仅适用于移动端）
setCameraZoomRatio	设置摄像头的缩放倍数（仅适用于移动端）
isAutoFocusEnabled	查询是否支持自动识别人脸位置（仅适用于移动端）
enableCameraAutoFocus	开启自动对焦功能（仅适用于移动端）
setCameraFocusPosition	设置摄像头的对焦位置（仅适用于移动端）
enableCameraTorch	开启/关闭闪光灯，也就是手电筒模式（仅适用于移动端）
setAudioRoute	设置音频路由（仅适用于移动端）
setExposureCompensation	设置摄像头的曝光参数，取值范围从-1到1

setCameraCaptureParam	设置摄像头采集偏好
setSystemVolumeType	设置系统音量类型（仅适用于移动端）

结构体类型

函数列表	描述
TXCameraCaptureParam	摄像头采集参数

枚举类型

枚举类型	描述
TXSystemVolumeType	系统音量类型（仅适用于移动设备）
TXAudioRoute	音频路由（即声音的播放模式）
TXCameraCaptureMode	摄像头采集偏好

isFrontCamera

isFrontCamera

判断当前是否为前置摄像头（仅适用于移动端）

switchCamera

switchCamera

int switchCamera (boolean frontCamera)

切换前置或后置摄像头（仅适用于移动端）

getCameraZoomMaxRatio

getCameraZoomMaxRatio

获取摄像头的最大缩放倍数（仅适用于移动端）

setCameraZoomRatio

setCameraZoomRatio

int setCameraZoomRatio (float zoomRatio)

设置摄像头的缩放倍数（仅适用于移动端）

参数	描述
zoomRatio	取值范围1 - 5，取值为1表示最远视角（正常镜头），取值为5表示最近视角（放大镜头）。最大值推荐为5，若超过5，视频数据会变得模糊不清。

isAutoFocusEnabled

isAutoFocusEnabled

查询是否支持自动识别人脸位置（仅适用于移动端）

enableCameraAutoFocus

enableCameraAutoFocus

int enableCameraAutoFocus (boolean enabled)

开启自动对焦功能（仅适用于移动端）

开启后，SDK 会自动检测画面中的人脸位置，并将摄像头的焦点始终对焦在人脸位置上。

setCameraFocusPosition

setCameraFocusPosition

```
int setCameraFocusPosition(int x, int y)
```

设置摄像头的对焦位置（仅适用于移动端）

您可以通过该接口实现如下交互：
1. 在本地摄像头的预览画面上，允许用户单击操作。
2. 在用户的单击位置显示一个矩形方框，以示摄像头会在此处对焦。
3. 随后将用户点击位置的坐标通过本接口传递给 SDK，之后 SDK 会操控摄像头按照用户期望的位置进行对焦。

参数	描述
position	对焦位置，请传入期望对焦点的坐标值

> **注意**
> 使用该接口的前提是先通过 enableCameraAutoFocus 关闭自动对焦功能。

返回值说明：

0：操作成功；负数：操作失败。

enableCameraTorch

enableCameraTorch

```
boolean enableCameraTorch(boolean enable)
```

开启/关闭闪光灯，也就是手电筒模式（仅适用于移动端）

setAudioRoute

setAudioRoute

int setAudioRoute (TXAudioRoute route)

设置音频路由（仅适用于移动端）

手机有两个音频播放设备：一个是位于手机顶部的听筒，一个是位于手机底部的立体声扬声器。
设置音频路由为听筒时，声音比较小，只有将耳朵凑近才能听清楚，隐私性较好，适合用于接听电话。
设置音频路由为扬声器时，声音比较大，不用将手机贴脸也能听清，因此可以实现“免提”的功能。

setExposureCompensation

setExposureCompensation

int setExposureCompensation (float value)

设置摄像头的曝光参数，取值范围从-1到1

setCameraCapturerParam

setCameraCapturerParam

void setCameraCapturerParam (TXCameraCaptureParam params)

设置摄像头采集偏好

setSystemVolumeType

setSystemVolumeType

int (TXSystemVolumeType type)

setSystemVolumeType

设置系统音量类型（仅适用于移动端）

@deprecated v9.5 版本开始不推荐使用，建议使用 TRTCCloud 中的 startLocalAudio(quality) 接口替代之，通过 quality 参数来决策音质。

TXSystemVolumeType(Deprecated)

TXSystemVolumeType(Deprecated)

系统音量类型（仅适用于移动设备）

@deprecated v9.5 版本开始不推荐使用。

现代智能手机中一般都具备两套系统音量类型，即“通话音量”和“媒体音量”。

- 通话音量：手机专门为接打电话所设计的音量类型，自带回声抵消（AEC）功能，并且支持通过蓝牙耳机上的麦克风进行拾音，缺点是音质比较一般。

当您通过手机侧面的音量按键下调手机音量时，如果无法将其调至零（也就是无法彻底静音），说明您的手机当前处于通话音量。

- 媒体音量：手机专门为音乐场景所设计的音量类型，无法使用系统的 AEC 功能，并且不支持通过蓝牙耳机的麦克风进行拾音，但具备更好的音乐播放效果。

当您通过手机侧面的音量按键下调手机音量时，如果能够将手机音量调至彻底静音，说明您的手机当前处于媒体音量。

SDK 目前提供了三种系统音量类型的控制模式：自动切换模式、全程通话音量模式、全程媒体音量模式。

枚举	取值	描述
TXSystemVolumeTypeAuto	Not Defined	自动切换模式
TXSystemVolumeTypeMedia	Not Defined	全程媒体音量
TXSystemVolumeTypeVOIP	Not Defined	全程通话音量

TXAudioRoute

TXAudioRoute

音频路由（即声音的播放模式）

音频路由，即声音是从手机的扬声器还是从听筒中播放出来，因此该接口仅适用于手机等移动端设备。
手机有两个扬声器：一个是位于手机顶部的听筒，一个是位于手机底部的立体声扬声器。

- 设置音频路由为听筒时，声音比较小，只有将耳朵凑近才能听清楚，隐私性较好，适合用于接听电话。
- 设置音频路由为扬声器时，声音比较大，不用将手机贴脸也能听清，因此可以实现“免提”的功能。

枚举	取值	描述
TXAudioRouteSpeakerphone	Not Defined	Speakerphone：使用扬声器播放（即“免提”），扬声器位于手机底部，声音偏大，适合外放音乐。
TXAudioRouteEarpiece	Not Defined	Earpiece：使用听筒播放，听筒位于手机顶部，声音偏小，适合需要保护隐私的通话场景。

TXCameraCaptureMode

TXCameraCaptureMode

摄像头采集偏好

该枚举类型用于摄像头采集参数设置。

枚举	取值	描述
TXCameraResolutionStrategyAuto	Not Defined	自动调整采集参数。 SDK 根据实际的采集设备性能及网络情况，选择合适的摄像头输出参数，在设备性能及视频预览质量之间，维持平衡。
TXCameraResolutionStrategyPerformance	Not Defined	优先保证设备性能。 SDK 根据用户设置编码器的分辨率和帧率，选择最接近的摄像头输出参数，从而保证设备性能。

TXCameraResolutionStrategyHighQuality	Not Defined	优先保证视频预览质量。 SDK选择较高的摄像头输出参数，从而提高预览视频的质量。在这种情况下，会消耗更多的 CPU 及内存做视频前处理。
TXCameraCaptureManual	Not Defined	允许用户设置本地摄像头采集的视频宽高。

TXCameraCaptureParam

TXCameraCaptureParam

摄像头采集参数

该设置能决定本地预览图像画质。

枚举类型	描述
height	**字段含义：** 采集图像宽度
mode	**字段含义：** 摄像头采集偏好，请参见 TXCameraCaptureMode
width	**字段含义：** 采集图像长度

错误码表

最近更新时间：2024-03-07 15:14:11

Module: V2TXLiveCode @ TXLiteAVSDK

Function: 腾讯云直播服务(LVB)错误码和警告码的定义。

错误码表

枚举类型

枚举类型	描述
V2TXLiveCode	V2 错误码和警告码

V2TXLiveCode

V2TXLiveCode

V2 错误码和警告码

枚举	取值	描述
V2TXLIVE_OK	0	没有错误。
V2TXLIVE_ERROR_FAILED	-1	暂未归类的通用错误。
V2TXLIVE_ERROR_INVALID_PARAMETER	-2	调用 API 时，传入的参数不合法。
V2TXLIVE_ERROR_REFUSED	-3	API 调用被拒绝。
V2TXLIVE_ERROR_NOT_SUPPORTED	-4	当前 API 不支持调用。
V2TXLIVE_ERROR_INVALID_LICENSE	-5	license 不合法，调用失败。

V2TXLIVE_ERROR_REQUEST_TIMEOUT	-6	请求服务器超时。
V2TXLIVE_ERROR_SERVER_PROCESS_FAILED	-7	服务器无法处理您的请求。
V2TXLIVE_ERROR_DISCONNECTED	-8	连接断开。
V2TXLIVE_ERROR_NO_AVAILABLE_HEVC_DECODERS	-2304	找不到可用的 HEVC 解码器。
V2TXLIVE_WARNING_NETWORK_BUSY	1101	网络状况不佳：上行带宽太小，上传数据受阻。
V2TXLIVE_WARNING_VIDEO_BLOCK	2105	当前视频播放出现卡顿。
V2TXLIVE_WARNING_CAMERA_START_FAILED	-1301	摄像头打开失败。
V2TXLIVE_WARNING_CAMERA_OCCUPIED	-1316	摄像头正在被占用中，可尝试打开其他摄像头。
V2TXLIVE_WARNING_CAMERA_NO_PERMISSION	-1314	摄像头设备未授权，通常在移动设备出现，可能是权限被用户拒绝了。
V2TXLIVE_WARNING_MICROPHONE_START_FAILED	-1302	麦克风打开失败。
V2TXLIVE_WARNING_MICROPHONE_OCCUPIED	-1319	麦克风正在被占用中，例如移动设备正在通话时，打开麦克风会失败。
V2TXLIVE_WARNING_MICROPHONE_NO_PERMISSION	-1317	麦克风设备未授权，通常在移动设备出现，可能是权限被用户拒绝了。
V2TXLIVE_WARNING_SCREEN_CAPTURE_NOT_SUPPORTED	-1309	当前系统不支持屏幕分享。
V2TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	-1308	开始录屏失败，如果在移动设备出现，可能是权限被用户拒绝了。
V2TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	-7001	录屏被系统中断。
V2TXLIVE_WARNING_CURRENT_ENCODE_TYPE_CHANGED	1104	表示编码器发生改变，可以通过 onWarning 函数的扩展信息中的 codec_type 字段来获取当前的编码格式。

		其中 1 代表 265 编码，0 代表 264 编码。注意 Windows 端不支持此错误码的扩展信息。
V2TXLIVE_WARNING_CURRENT_DECODE_TYPE_CHANGED	2008	表示解码器发生改变，可以通过 onWarning 函数的扩展信息中的 codec_type 字段来获取当前的解码格式。 其中 1 代表 265 解码，0 代表 264 解码。注意 Windows 端不支持此错误码的扩展信息。

类型定义

最近更新时间：2024-01-16 14:35:22

Module: V2TXLiveDef @ TXLiteAVSDK

Function: 腾讯云直播服务(LVB)关键类型定义

类型定义

结构体类型

函数列表	描述
V2TXLiveVideoEncoderParam	视频编码参数
V2TXLiveVideoFrame	视频帧信息
V2TXLiveAudioFrame	音频帧数据
V2TXLiveAudioFrameObserverFormat	音频帧回调格式
V2TXLivePusherStatistics	推流器的统计数据
V2TXLivePlayerStatistics	播放器的统计数据
V2TXLiveMixStream	云端混流中每一路子画面的位置信息
V2TXLiveTranscodingConfig	云端混流（转码）配置
V2TXLiveLocalRecordingParams	本地录制音视频配置
V2TXLiveSocks5ProxyConfig	socks5 代理的协议配置
V2TXLiveLogConfig	Log配置
V2TXLiveStreamInfo	支持自适应切换的码流信息

枚举类型

枚举类型	描述
V2TXLiveMode	支持协议
V2TXLiveVideoResolution	视频分辨率
V2TXLiveVideoResolutionMode	视频宽高比模式
V2TXLiveMirrorType	本地摄像头镜像类型
V2TXLiveFillMode	视频画面填充模式
V2TXLiveRotation	视频画面顺时针旋转角度
V2TXLivePixelFormat	视频帧的像素格式
V2TXLiveBufferType	视频数据包装格式
V2TXLiveTexture	视频纹理包装类
V2TXLiveAudioQuality	声音音质
V2TXLiveAudioFrameOperationMode	音频回调数据读写模式
V2TXLivePushStatus	直播流的连接状态
V2TXLiveMixInputType	混流输入类型配置
V2TXLiveRecordMode	本地音视频录制模式
V2TXLiveLogLevel	日志级别枚举值

V2TXLiveMode

V2TXLiveMode

支持协议

枚举	取值	描述

TXLiveMode_RTMP	Not Defined	支持协议: RTMP。
TXLiveMode_RTC	Not Defined	支持协议: TRTC。

V2TXLiveVideoResolution

V2TXLiveVideoResolution

视频分辨率

枚举	取值	描述
V2TXLiveVideoResolution160x160	Not Defined	分辨率 160*160，码率范围：100Kbps ~ 150Kbps，帧率：15fps。
V2TXLiveVideoResolution270x270	Not Defined	分辨率 270*270，码率范围：200Kbps ~ 300Kbps，帧率：15fps。
V2TXLiveVideoResolution480x480	Not Defined	分辨率 480*480，码率范围：350Kbps ~ 525Kbps，帧率：15fps。
V2TXLiveVideoResolution320x240	Not Defined	分辨率 320*240，码率范围：250Kbps ~ 375Kbps，帧率：15fps。
V2TXLiveVideoResolution480x360	Not Defined	分辨率 480*360，码率范围：400Kbps ~ 600Kbps，帧率：15fps。
V2TXLiveVideoResolution640x480	Not Defined	分辨率 640*480，码率范围：600Kbps ~ 900Kbps，帧率：15fps。
V2TXLiveVideoResolution320x180	Not Defined	分辨率 320*180，码率范围：250Kbps ~ 400Kbps，帧率：15fps。

V2TXLiveVideoResolution480x270	Not Defined	分辨率 480*270，码率范围：350Kbps ~ 550Kbps，帧率：15fps。
V2TXLiveVideoResolution640x360	Not Defined	分辨率 640*360，码率范围：500Kbps ~ 900Kbps，帧率：15fps。
V2TXLiveVideoResolution960x540	Not Defined	分辨率 960*540，码率范围：800Kbps ~ 1500Kbps，帧率：15fps。
V2TXLiveVideoResolution1280x720	Not Defined	分辨率 1280*720，码率范围：1000Kbps ~ 1800Kbps，帧率：15fps。
V2TXLiveVideoResolution1920x1080	Not Defined	分辨率 1920*1080，码率范围：2500Kbps ~ 3000Kbps，帧率：15fps。

V2TXLiveVideoResolutionMode

V2TXLiveVideoResolutionMode

视频宽高比模式

注意

- 横屏模式下的分辨率: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModeLandscape = 640 × 360。
- 竖屏模式下的分辨率: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModePortrait = 360 × 640。

枚举	取值	描述
V2TXLiveVideoResolutionModeLandscape	Not Defined	横屏模式。
V2TXLiveVideoResolutionModePortrait	Not Defined	竖屏模式。

V2TXLiveMirrorType

V2TXLiveMirrorType

本地摄像头镜像类型

枚举	取值	描述
V2TXLiveMirrorTypeAuto	Not Defined	系统默认镜像类型，前置摄像头镜像，后置摄像头不镜像。
V2TXLiveMirrorTypeEnable	Not Defined	前置摄像头和后置摄像头，都切换为镜像模式。
V2TXLiveMirrorTypeDisable	Not Defined	前置摄像头和后置摄像头，都切换为非镜像模式。

V2TXLiveFillMode

V2TXLiveFillMode

视频画面填充模式

枚举	取值	描述
V2TXLiveFillModeFill	Not Defined	图像铺满屏幕，超出显示视窗的视频部分将被裁剪，画面显示可能不完整。
V2TXLiveFillModeFit	Not Defined	图像长边填满屏幕，短边区域会被填充黑色，画面的内容完整。
V2TXLiveFillModeScaleFill	Not Defined	图像拉伸铺满，因此长度和宽度可能不会按比例变化。

V2TXLiveRotation

V2TXLiveRotation

视频画面顺时针旋转角度

枚举	取值	描述
V2TXLiveRotation0	Not Defined	不旋转。
V2TXLiveRotation90	Not Defined	顺时针旋转90度。
V2TXLiveRotation180	Not Defined	顺时针旋转180度。
V2TXLiveRotation270	Not Defined	顺时针旋转270度。

V2TXLivePixelFormat

V2TXLivePixelFormat

视频帧的像素格式

枚举	取值	描述
V2TXLivePixelFormatUnknown	Not Defined	未知。
V2TXLivePixelFormatI420	Not Defined	YUV420P I420。
V2TXLivePixelFormatText	Not	OpenGL 2D 纹理。

ure2D	Defined	

V2TXLiveBufferType

V2TXLiveBufferType

视频数据包装格式

> **注意**
> 在自定义采集和自定义渲染功能，您需要用到下列枚举值来指定您希望以什么类型的容器来包装视频数据。
> - Texture: 直接使用时效率最高。

枚举	取值	描述
V2TXLiveBufferTypeUnknown	Not Defined	未知。
V2TXLiveBufferTypeByteBuffer	Not Defined	DirectBuffer，装载 I420 等 buffer，在 native 层使用。
V2TXLiveBufferTypeByteArray	Not Defined	byte[]，装载 I420 等 buffer，在 Java 层使用。
V2TXLiveBufferTypeTexture	Not Defined	直接操作纹理 ID，性能最好，画质损失最少。

V2TXLiveTexture

V2TXLiveTexture

视频纹理包装类

枚举	取值	描述
public int textureId	Not Defined	视频纹理 ID。
public javax.microedition.khronos.egl.EGLContext eglContext10	Not Defined	使用 (javax.microedition.khronos.egl.*) 定义的 OpenGL 接口。
public android.opengl.EGLContext eglContext14	Not Defined	使用 (android.opengl.*) 定义的 OpenGL 接口。

V2TXLiveAudioQuality

V2TXLiveAudioQuality

声音音质

枚举	取值	描述
V2TXLiveAudioQualitySpeech	Not Defined	语音音质：采样率：16k；单声道；音频码率：16kbps；适合语音通话为主的场景，比如在线会议，语音通话。
V2TXLiveAudioQualityDefault	Not Defined	默认音质：采样率：48k；单声道；音频码率：50kbps；SDK 默认的音频质量，如无特殊需求推荐选择之。
V2TXLiveAudioQualityMusic	Not Defined	音乐音质：采样率：48k；双声道 + 全频带；音频码率：128kbps；适合需要高保真传输音乐的场景，比如K歌、音乐直播等。

V2TXLiveAudioFrameOperationMode

V2TXLiveAudioFrameOperationMode

音频回调数据读写模式

SDK 提供了两种音频回调数据的操作模式。

- 读写模式（ReadWrite）：可以获取并修改回调的音频数据，默认模式。
- 只读模式（ReadOnly）：仅从回调中获取音频数据。

枚举	取值	描述
V2TXLiveAudioFrameOperationModeReadWrite	Not Defined	读写模式：可以获取并修改回调的音频数据。
V2TXLiveAudioFrameOperationModeReadOnly	Not Defined	只读模式：仅从回调中获取音频数据。

V2TXLivePushStatus

V2TXLivePushStatus

直播流的连接状态

枚举	取值	描述
V2TXLivePushStatusDisconnected	Not Defined	与服务器断开连接。
V2TXLivePushStatusConnecting	Not Defined	正在连接服务器。
V2TXLivePushStatusConnectSuccess	Not Defined	连接服务器成功。
V2TXLivePushStatusReconnecting	Not Defined	重连服务器中。

V2TXLiveMixInputType

V2TXLiveMixInputType

混流输入类型配置

枚举	取值	描述
V2TXLiveMixInputTypeAudioVideo	Not Defined	混入音视频。
V2TXLiveMixInputTypePureVideo	Not Defined	只混入视频。
V2TXLiveMixInputTypePureAudio	Not Defined	只混入音频。

V2TXLiveRecordMode

V2TXLiveRecordMode

本地音视频录制模式

枚举	取值	描述
V2TXLiveRecordModeBoth	Not Defined	Both mode: 录制音频和视频

V2TXLiveLogLevel

V2TXLiveLogLevel

日志级别枚举值

枚举	取值	描述

V2TXLiveLogLevelAll	0	输出所有级别的 log。
V2TXLiveLogLevelDebug	1	输出 DEBUG，INFO，WARNING，ERROR 和 FATAL 级别的 log。
V2TXLiveLogLevelInfo	2	输出 INFO，WARNING，ERROR 和 FATAL 级别的 log。
V2TXLiveLogLevelWarning	3	只输出 WARNING，ERROR 和 FATAL 级别的 log。
V2TXLiveLogLevelError	4	只输出 ERROR 和 FATAL 级别的 log。
V2TXLiveLogLevelFatal	5	只输出 FATAL 级别的 log。
V2TXLiveLogLevelNULL	6	不输出任何 sdk log。

V2TXLiveVideoEncoderParam

V2TXLiveVideoEncoderParam

视频编码参数

该设置决定远端用户看到的画面质量。

枚举类型	描述
minVideoBitrate	【字段含义】最低视频码率，SDK 会在网络不佳的情况下主动降低视频码率以保持流畅度，最低会降至 minVideoBitrate 所设定的数值。 【推荐取值】您可以通过同时设置 videoBitrate 和 minVideoBitrate 两个参数，用于约束 SDK 对视频码率的调整范围： • 如果您将 videoBitrate 和 minVideoBitrate 设置为同一个值，等价于关闭 SDK 对视频码率的自适应调节能力。
videoBitrate	【字段含义】目标视频码率，SDK 会按照目标码率进行编码，只有在弱网络环境下才会主动降低视频码率。 【推荐取值】请参考 V2TXLiveVideoResolution 在各档位注释的最佳码率，也可以在此基础上适当调高。 比如：V2TXLiveVideoResolution1280x720 对应 1200kbps 的目标码率，您也可以设置为 1500kbps 用来获得更好的观感清晰度。

	【特别说明】您可以通过同时设置 videoBitrate 和 minVideoBitrate 两个参数，用于约束 SDK 对视频码率的调整范围： • 如果您将 videoBitrate 和 minVideoBitrate 设置为同一个值，等价于关闭 SDK 对视频码率的自适应调节能力。
videoFps	【字段含义】视频采集帧率。 【推荐取值】15fps 或 20fps。5fps 以下，卡顿感明显。10fps 以下，会有轻微卡顿感。20fps 以上，会浪费带宽（电影的帧率为 24fps）。
videoResolution	【字段含义】视频分辨率。 【特别说明】如需使用竖屏分辨率，请指定 videoResolutionMode 为 Portrait，例如： 640 × 360 + Portrait = 360 × 640。 【推荐取值】 • 桌面平台（Win + Mac）：建议选择 640 × 360 及以上分辨率，videoResolutionMode 选择 Landscape，即横屏分辨率。
videoResolutionMode	【字段含义】分辨率模式（横屏分辨率 or 竖屏分辨率）。 【推荐取值】桌面平台（Windows、Mac）建议选择 Landscape。 【特别说明】如需使用竖屏分辨率，请指定 resMode 为 Portrait，例如：640 × 360 + Portrait = 360 × 640。

V2TXLiveVideoFrame

V2TXLiveVideoFrame

视频帧信息

注意
自定义采集和自定义渲染时使用。自定义采集时，需要使用 V2TXLiveVideoFrame 来包装待发送的视频帧；自定义渲染时，会返回经过 V2TXLiveVideoFrame 包装的视频帧。

枚举类型	描述
buffer	【字段含义】视频数据。
bufferType	【字段含义】视频数据包装格式。
data	【字段含义】视频数据。
height	【字段含义】视频高度。
pixelForma	【字段含义】视频帧像素格式。

t	
rotation	【字段含义】视频帧的顺时针旋转角度。
texture	【字段含义】视频纹理包装类。
width	【字段含义】视频宽度。

V2TXLiveAudioFrame

V2TXLiveAudioFrame

音频帧数据

枚举类型	描述
channel	【字段含义】声道数。
data	【字段含义】音频数据。
sampleRate	【字段含义】采样率。
timestamp	【字段含义】时间戳，单位ms。

V2TXLiveAudioFrameObserverFormat

V2TXLiveAudioFrameObserverFormat

音频帧回调格式

枚举类型	描述
channel	【字段含义】声道数。 【推荐取值】默认值：1，代表单声道。可设定的数值只有两个数字：1-单声道，2-双声道。
mode	【字段含义】回调数据读写模式。 【推荐取值】V2TXLiveAudioFrameOperationModeReadOnly：仅从回调中获取音频数据。可设定的模式有

	V2TXLiveAudioFrameOperationModeReadOnly， V2TXLiveAudioFrameOperationModeReadWrite。
sampleRate	【字段含义】采样率。 【推荐取值】默认值：48000Hz。支持 16000, 32000, 44100, 48000。
samplesPerCall	【字段含义】采样点数。 【推荐取值】取值必须是 sampleRate/100 的整数倍。

V2TXLivePusherStatistics

V2TXLivePusherStatistics

推流器的统计数据

枚举类型	描述
appCpu	【字段含义】当前 App 的 CPU 使用率（%）。
audioBitrate	【字段含义】音频码率（Kbps）。
fps	【字段含义】帧率（fps）。
height	【字段含义】视频高度。
netSpeed	【字段含义】上行速度（kbps）
rtt	【字段含义】从 SDK 到云端的往返延时（ms）
systemCpu	【字段含义】当前系统的 CPU 使用率（%）。
videoBitrate	【字段含义】视频码率（Kbps）。
width	【字段含义】视频宽度。

V2TXLivePlayerStatistics

V2TXLivePlayerStatistics

播放器的统计数据

枚举类型	描述
appCpu	【字段含义】当前 App 的 CPU 使用率（%）。
audioBitrate	【字段含义】音频码率（Kbps）。
audioBlockRate	【字段含义】音频播放卡顿率，单位（%）。 音频播放卡顿率（audioBlockRate）= 音频播放的累计卡顿时长（audioTotalBlockTime）/ 音频播放的区间时长（2000ms）。
audioPacketLoss	【字段含义】网络音频丢包率（%），注：仅支持前缀为 [trtc://] 或 [webrtc://] 的播放地址。
audioTotalBlockTime	【字段含义】音频播放的累计卡顿时长（ms）。 该时长为区间（2s）内的卡顿时长。
fps	【字段含义】帧率（fps）。
height	【字段含义】视频高度。
jitterBufferDelay	【字段含义】播放延迟（ms）。
netSpeed	【字段含义】下载速度（kbps）
rtt	【字段含义】从 SDK 到云端的往返延时（ms），注：仅支持前缀为 [trtc://] 或 [webrtc://] 的播放地址。
systemCpu	【字段含义】当前系统的 CPU 使用率（%）。
videoBitrate	【字段含义】视频码率（Kbps）。
videoBlockRate	【字段含义】视频播放卡顿率，单位（%）。 视频播放卡顿率（videoBlockRate）= 视频播放的累计卡顿时长（videoTotalBlockTime）/ 视频播放的区间时长（2000ms）。
videoPacketLoss	【字段含义】网络视频丢包率（%），注：仅支持前缀为 [trtc://] 或 [webrtc://] 的播放地址。
videoTotalBlockTime	【字段含义】视频播放的累计卡顿时长（ms）。 该时长为区间（2s）内的卡顿时长。
width	【字段含义】视频宽度。

V2TXLiveMixStream

V2TXLiveMixStream

云端混流中每一路子画面的位置信息

枚举类型	描述
height	【字段含义】图层位置高度（绝对像素值）。
inputType	【字段含义】该直播流的输入类型。
streamId	【字段含义】参与混流的 userId 所在对应的推流 streamId，nil 表示当前推流 streamId。
userId	【字段含义】参与混流的 userId。
width	【字段含义】图层位置宽度（绝对像素值）。
x	【字段含义】图层位置 x 坐标（绝对像素值）。
y	【字段含义】图层位置 y 坐标（绝对像素值）。
zOrder	【字段含义】图层层次（1 – 15）不可重复。

V2TXLiveTranscodingConfig

V2TXLiveTranscodingConfig

云端混流（转码）配置

枚举类型	描述
audioBitrate	【字段含义】最终转码后的音频码率。 【推荐取值】默认值：64kbps，取值范围是 [32，192]，单位：kbps。
audioChannels	【字段含义】最终转码后的音频声道数。 【推荐取值】默认值：1。取值范围为 [1,2] 中的整型。
audioSampleRate	【字段含义】最终转码后的音频采样率。 【推荐取值】默认值：48000Hz。支持12000HZ、16000HZ、22050HZ、24000HZ、32000HZ、44100HZ、48000HZ。
backgroundColor	【字段含义】混合后画面的底色颜色，默认为黑色，格式为十六进制数字，比如：“0x61B9F1”代表 RGB 分别为(97,158,241)。 【推荐取值】默认值：0x000000，黑色。

backgroundImage	【字段含义】混合后画面的背景图。 【推荐取值】默认值：nil，即不设置背景图。 【特别说明】背景图需要您事先在 “控制台 => 应用管理 => 功能配置 => 素材管理” 中上传， 上传成功后可以获得对应的“图片ID”，然后将“图片ID”转换成字符串类型并设置到 backgroundImage 里即可。 例如：假设“图片ID” 为 63，可以设置 backgroundImage = "63"。
mixStreams	【字段含义】每一路子画面的位置信息。
outputStreamId	【字段含义】输出到 CDN 上的直播流 ID。 如不设置该参数，SDK 会执行默认逻辑，即房间里的多路流会混合到该接口调用者的视频流上，也就是 A + B => A。 如果设置该参数，SDK 会将房间里的多路流混合到您指定的直播流 ID 上，也就是 A + B => C。 【推荐取值】默认值：nil，即房间里的多路流会混合到该接口调用者的视频流上。
videoBitrate	【字段含义】最终转码后的视频分辨率的码率（kbps）。 【推荐取值】如果填0，后台会根据 videoWidth 和 videoHeight 来估算码率，您也可以参考枚举定义 V2TXLiveVideoResolution 的注释。
videoFramerate	【字段含义】最终转码后的视频分辨率的帧率（FPS）。 【推荐取值】默认值：15fps，取值范围是 (0,30]。
videoGOP	【字段含义】最终转码后的视频分辨率的关键帧间隔（又称为 GOP）。 【推荐取值】默认值：2，单位为秒，取值范围是 [1,8]。
videoHeight	【字段含义】最终转码后的视频分辨率的高度。 【推荐取值】推荐值：640px，如果你是纯音频推流，请将 width × height 设为 0px × 0px，否则混流后会携带一条画布背景的视频流。
videoWidth	【字段含义】最终转码后的视频分辨率的宽度。 【推荐取值】推荐值：360px，如果你是纯音频推流，请将 width × height 设为 0px × 0px，否则混流后会携带一条画布背景的视频流。

V2TXLiveLocalRecordingParams

V2TXLiveLocalRecordingParams

本地录制音视频配置

枚举类型	描述

filePath	【字段含义】录制的文件地址（必填），请确保路径有读写权限且合法，否则录制文件无法生成。 【推荐取值】该路径需精确到文件名及格式后缀，格式后缀用于决定录制出的文件格式，目前支持的格式暂时只有 MP4。
interval	【字段含义】interval 录制信息更新频率，单位毫秒，有效范围：1000-10000。 【推荐取值】`-1`，表示不回调。
recordMode	【字段含义】媒体录制模式。 【推荐取值】`V2TXLiveRecordModeBoth`，即同时录制音频和视频。

V2TXLiveSocks5ProxyConfig

V2TXLiveSocks5ProxyConfig

socks5 代理的协议配置

枚举类型	描述
supportHttps	【字段含义】是否支持 https。 【推荐取值】默认值：true。
supportTcp	【字段含义】是否支持 tcp。 【推荐取值】默认值：true。
supportUdp	【字段含义】是否支持 udp。 【推荐取值】默认值：true。

V2TXLiveLogConfig

V2TXLiveLogConfig

Log配置

枚举类型	描述
enableConsole	【字段含义】是否允许 SDK 在编辑器（XCoder、Android Studio、Visual Studio 等）的控制台上打印 Log。 【推荐取值】默认值：false。

enableLogFile	【字段含义】是否启用本地 Log 文件。 【特殊说明】如非特殊需要，请不要关闭本地 Log 文件，否则腾讯云技术团队将无法在出现问题时进行跟踪和定位。 【推荐取值】默认值：true。
enableObserver	【字段含义】是否通过 V2TXLivePremierObserver 接收要打印的 Log 信息。 【特殊说明】如果您希望自己实现 Log 写入，可以打开此开关，Log 信息会通过 V2TXLivePremierObserver#onLog 回调给您。 【推荐取值】默认值：false。
logLevel	【字段含义】设置 Log 级别。 【推荐取值】默认值：V2TXLiveLogLevel.V2TXLiveLogLevelAll。
logPath	【字段含义】设置本地 Log 的存储目录，默认 Log 存储位置：Android：/sdcard/Android/data/应用包名/files/log/liteav/。

V2TXLiveStreamInfo

V2TXLiveStreamInfo

支持自适应切换的码流信息

枚举类型	描述
height	**字段含义** 视频高, 默认值：0, 表示未知。
url	**字段含义** 流地址，通过 SwitchStream 接口调用实现多码率质量切换。
width	**字段含义** 视频宽, 默认值：0, 表示未知。

属性定义

最近更新时间：2024-03-07 15:14:11

Module: V2TXLiveProperty @ TXLiteAVSDK

Function: V2TXLive setProperty 支持的 key

属性定义

结构体类型

函数列表	描述
V2TXLiveProperty	V2 直播属性

V2TXLiveProperty

V2TXLiveProperty

V2 直播属性

枚举类型	描述
kV2ClearLastImage	是否清理最后一帧。 默认值：true。 Value：true/false。
kV2EnableHardwareAcceleration	开启/关闭硬件加速【RTMP协议，拉流】。 默认值：true。 Value：true/false。
kV2EnableHevcEncode	开启/关闭 Hevc 编码【RTMP/RTC协议，推流】。 默认值：false。 Value：true/false。

kV2EnableIPMultiplexing	开启/关闭 IP 复用【FLV，拉流】。 默认值：false。 Value：true/false。
kV2MaxNumberOfReconnection	设置重连次数,【RTMP协议，拉流】。 默认值：3。 Value：int。
kV2SecondsBetweenReconnection	设置重连间隔【RTMP协议，拉流】。 单位：秒。 默认值：3。 Value：int。

kV2SetHeaders	设定播放请求头【FLV，拉流】。
	Value：JSON 字符串。
	例如：
	{
	"headers": [
	{
	"key": "key1",
	"value": "value1"
	},
	{
	"key": "key2",
	"value": "value2"
	}
	]
	}

kV2SetMetaData	设置推流 Meta 信息【RTMP，推流】。
	Value：JSON 字符串。
	例如：
	{
	"metadata": [
	{
	"key": "key1",
	"value": "value1"
	},
	{
	"key": "key2",
	"value": "value2"
	}
	]
	}

kV2SetVideoQualityEx	设置自定义编码参数【RTMP/RTC协议，推流】。

	Value：JSON 字符串。
	例如：
	{
	"videoWidth":360,
	"videoHeight":640,
	"videoFps":15,
	"videoBitrate":1000,
	"minVideoBitrate":1000
	}

连麦

MLVBLiveRoom

最近更新时间：2024-01-23 12:38:31

功能

腾讯云视立方 · 直播 SDK – 连麦直播间。

> ⚠ 注意
> 后台接口限制并发为每秒100次请求，若您有高并发请求请提前 联系我们 处理，避免影响服务调用。

介绍

基于腾讯云直播、云点播（VOD）和即时通信（IM）三大 PAAS 服务组合而成，支持：

- 主播创建新的直播间开播，观众进入直播间观看。
- 主播和观众进行视频连麦互动。
- 两个不同房间的主播 PK 互动。
- 一个直播间都有一个不限制房间人数的聊天室，支持发送各种文本消息和自定义消息，自定义消息可用于实现弹幕、点赞和礼物。

连麦直播间（MLVBLiveRoom）是一个开源的 Class，依赖两个腾讯云的闭源 SDK：

- LiteAVSDK：使用了其中的 TXLivePusher 和 TXLivePlayer 两个组件，前者用于推流，后者用于拉流。
- IM SDK：使用 IM SDK 的 AVChatroom 用于实现直播聊天室的功能，同时，主播间的连麦流程也是依靠 IM 消息串联起来的。

参见文档：直播连麦 。

SDK 基础函数

sharedInstance

获取 MLVBLiveRoom 单例对象。

```
MLVBLiveRoom sharedInstance(Context context)
```

参数

参数	类型	含义
context	Context	Android 上下文，内部会转为 ApplicationContext 用于系统 API 调用。

返回

MLVBLiveRoom 实例。

> **说明**
> 可以调用 MLVBLiveRoom#destroySharedInstance() 销毁单例对象。

destroySharedInstance

销毁 MLVBLiveRoom 单例对象。

```
void destroySharedInstance()
```

> **说明**
> 销毁实例后，外部缓存的 MLVBLiveRoom 实例不能再使用，需要重新调用 MLVBLiveRoom#sharedInstance(Context) 获取新实例。

setListener

设置回调接口。

```
abstract void setListener(IMLVBLiveRoomListener listener)
```

参数

参数	类型	含义
listener	IMLVBLiveRoomListener	回调接口。

介绍

您可以通过 IMLVBLiveRoomListener 获得 MLVBLiveRoom 的各种状态通知。

> **说明**
> 默认是在 Main Thread 中回调，如果需要自定义回调线程，可使用 MLVBLiveRoom#setListenerHandler(Handler) 。

setListenerHandler

设置驱动回调的线程。

```
abstract void setListenerHandler(Handler listenerHandler)
```

参数

参数	类型	含义
listenerHandler	Handler	线程。

login

登录。

```
abstract void login(final LoginInfo loginInfo, final IMLVBLiveRoomListener.LoginCallback callback)
```

参数

参数	类型	含义
loginInfo	final LoginInfo	登录信息。
callback	final IMLVBLiveRoomListener.LoginCallback	登录结果回调。

logout

退出登录。

```
abstract void logout()
```

setSelfProfile

修改个人信息。

```
abstract void setSelfProfile(String userName, String avatarURL)
```

参数

参数	类型	含义
userName	String	昵称。
avatarURL	String	头像地址。

房间相关接口函数

getRoomList

获取房间列表。

```
abstract void getRoomList(int index, int count, final
IMLVBLiveRoomListener.GetRoomListCallback callback)
```

参数

参数	类型	含义
index	int	房间开始索引，从0开始计算。
count	int	希望后台返回的房间个数。
callback	final IMLVBLiveRoomListener.GetRoomListCallback	获取房间列表的结果回调。

介绍

该接口支持分页获取房间列表，可以用 index 和 count 两个参数控制列表分页的逻辑：

- index = 0 & count = 10 代表获取第一页的10个房间。
- index = 11 & count = 10 代表获取第二页的10个房间。

getAudienceList

获取观众列表。

```
abstract void getAudienceList(IMLVBLiveRoomListener.GetAudienceListCallback
callback)
```

参数

参数	类型	含义
callback	IMLVBLiveRoomListener.GetAudienceListCallback	获取观众列表的结果回调。

介绍

当有观众进房时，后台会将其信息加入到指定房间的观众列表中，调入该函数即可返回指定房间的观众列表。

> **说明**
> 观众列表最多只保存30人，因为对于常规的 UI 展示来说这已经足够，保存更多除了浪费存储空间，也会拖慢列表返回的速度。

createRoom

创建房间（主播调用）。

```
abstract void createRoom(final String roomID, final String roomInfo, final IMLVBLiveRoomListener.CreateRoomCallback callback)
```

参数

参数	类型	含义
roomID	final String	房间标识，推荐做法是用主播的 userID 作为房间的 roomID，这样省去了后台映射的成本。roomID 可以填空，此时由后台生成。
roomInfo	final String	房间信息（非必填），用于房间描述的信息，如房间名称，允许使用 JSON 格式作为房间信息。
callback	final IMLVBLiveRoomListener.CreateRoomCallback	创建房间的结果回调。

介绍

主播开播的正常调用流程是：

1. 主播调用 startLocalPreview() 打开摄像头预览，此时可以调整美颜参数。
2. 主播调用 createRoom 创建直播间，房间创建成功与否会通过 IMLVBLiveRoomListener.CreateRoomCallback 通知给主播。

enterRoom

进入房间（观众调用）。

```
abstract void enterRoom(final String roomID, final TXCloudVideoView view, final IMLVBLiveRoomListener.EnterRoomCallback callback)
```

参数

参数	类型	含义
roomID	final String	房间标识。
view	final TXCloudVideoView	承载视频画面的控件。
callback	final IMLVBLiveRoomListener.EnterRoomCallback	进入房间的结果回调。

介绍

观众观看直播的正常调用流程是：

1. 观众调用 getRoomList() 刷新最新的直播房间列表，并通过 IMLVBLiveRoomListener.GetRoomListCallback 回调拿到房间列表。
2. 观众选择一个直播间以后，调用 enterRoom() 进入该房间。

exitRoom

离开房间。

```
abstract void exitRoom(IMLVBLiveRoomListener.ExitRoomCallback callback)
```

参数

参数	类型	含义
callback	IMLVBLiveRoomListener.ExitRoomCallback	离开房间的结果回调。

setCustomInfo

设置自定义信息。

```
abstract void setCustomInfo(final MLVBCommonDef.CustomFieldOp op, final String
key, final Object value, final IMLVBLiveRoomListener.SetCustomInfoCallback callback)
```

参数

参数	类型	含义
op	final MLVBCommonDef.CustomFieldOp	执行动作，定义请查看 MLVBCommonDef.CustomFieldOp。
key	final String	自定义键。
value	final Object	数值。
callback	final IMLVBLiveRoomListener.SetCustomInfoCallback	设置自定义信息完成的回调。

介绍

有时候您可能需要为房间产生一些额外的信息，此接口可以将这些信息缓存到服务器。

说明

- op 为 MLVBCommonDef.CustomFieldOp#SET 时，value 可以是 String 或者 Integer 类型。
- op 为 MLVBCommonDef.CustomFieldOp#INC 时，value 是 Integer 类型。
- op 为 MLVBCommonDef.CustomFieldOp#DEC 时，value 是 Integer 类型。

getCustomInfo

获取自定义信息。

```
abstract void getCustomInfo(final IMLVBLiveRoomListener.GetCustomInfoCallback
callback)
```

参数

参数	类型	含义
callback	final IMLVBLiveRoomListener.GetCustomInfoCallback	获取自定义信息回调。

主播和观众连麦

requestJoinAnchor

观众请求连麦。

```
abstract void requestJoinAnchor(String reason,
IMLVBLiveRoomListener.RequestJoinAnchorCallback callback)
```

参数

参数	类型	含义
reason	String	连麦原因。
callback	IMLVBLiveRoomListener.RequestJoinAnchorCallback	请求连麦的回调。

介绍

主播和观众的连麦流程可以简单描述为如下几个步骤：

1. 观众调用 requestJoinAnchor() 向主播发起连麦请求。
2. 主播会收到 IMLVBLiveRoomListener#onRequestJoinAnchor(AnchorInfo， String) 的回调通知。

3. 主播调用 responseJoinAnchor() 确定是否接受观众的连麦请求。
4. 观众会收到 IMLVBLiveRoomListener.RequestJoinAnchorCallback 回调通知，可以得知请求是否被同意。
5. 观众如果请求被同意，则调用 startLocalPreview() 开启本地摄像头，如果 App 还没有取得摄像头和麦克风权限，会触发 UI 提示。
6. 观众然后调用 joinAnchor() 正式进入连麦状态。
7. 主播一旦观众进入连麦状态，主播就会收到 IMLVBLiveRoomListener#onAnchorEnter(AnchorInfo) 通知。
8. 主播调用 startRemoteView() 就可以看到连麦观众的视频画面。
9. 观众如果直播间里已经有其他观众正在跟主播进行连麦，那么新加入的这位连麦观众也会收到 onAnchorJoin() 通知，用于展示（startRemoteView）其他连麦者的视频画面。

responseJoinAnchor

主播处理连麦请求。

```
abstract int responseJoinAnchor(String userID, boolean agree, String reason)
```

参数

参数	类型	含义
userID	String	观众ID。
agree	boolean	true：同意；false：拒绝。
reason	String	同意/拒绝连麦的原因描述。

返回

0：响应成功；非0：响应失败。

介绍

主播在收到 IMLVBLiveRoomListener#onRequestJoinAnchor(AnchorInfo， String) 回调之后会需要调用此接口来处理观众的连麦请求。

joinAnchor

进入连麦状态。

```
abstract void joinAnchor(final IMLVBLiveRoomListener.JoinAnchorCallback callback)
```

参数

参数	类型	含义
callback	final IMLVBLiveRoomListener.JoinAnchorCallback	进入连麦的结果回调。

介绍

进入连麦成功后，主播和其他连麦观众会收到 IMLVBLiveRoomListener#onAnchorEnter(AnchorInfo) 通知。

quitJoinAnchor

观众退出连麦。

```
abstract void quitJoinAnchor(final IMLVBLiveRoomListener.QuitAnchorCallback
callback)
```

参数

参数	类型	含义
callback	final IMLVBLiveRoomListener.QuitAnchorCallback	退出连麦的结果回调。

介绍

退出连麦成功后，主播和其他连麦观众会收到 IMLVBLiveRoomListener#onAnchorExit(AnchorInfo) 通知。

kickoutJoinAnchor

主播踢除连麦观众。

```
abstract void kickoutJoinAnchor(String userID)
```

参数

参数	类型	含义
userID	String	连麦观众 ID。

介绍

主播调用此接口踢除连麦观众后，被踢连麦观众会收到 IMLVBLiveRoomListener#onKickoutJoinAnchor() 回调通知。

主播跨房间 PK

requestRoomPK

请求跨房 PK。

```
abstract void requestRoomPK(String userID, final
IMLVBLiveRoomListener.RequestRoomPKCallback callback)
```

参数

参数	类型	含义
userID	String	被邀约主播 ID。
callback	final IMLVBLiveRoomListener.RequestRoomPKCallback	请求跨房 PK 的结果回调。

介绍

主播和主播之间可以跨房间 PK，两个正在直播中的主播 A 和 B，他们之间的跨房 PK 流程如下：

1. 主播 A 调用 requestRoomPK() 向主播 B 发起连麦请求。
2. 主播 B 会收到 IMLVBLiveRoomListener#onRequestRoomPK(AnchorInfo) 回调通知。
3. 主播 B 调用 responseRoomPK() 确定是否接受主播 A 的 PK 请求。
4. 主播 B 如果接受了主播 A 的要求，可以直接调用 startRemoteView() 来显示主播 A 的视频画面。
5. 主播 A 会收到 IMLVBLiveRoomListener.RequestRoomPKCallback 回调通知，可以得知请求是否被同意。
6. 主播 A 如果请求被同意，则可以调用 startRemoteView() 显示主播 B 的视频画面。

responseRoomPK

响应跨房 PK 请求。

```
abstract int responseRoomPK(String userID, boolean agree, String reason)
```

参数

参数	类型	含义
userID	String	发起 PK 请求的主播 ID。
agree	boolean	true：同意；false：拒绝。
reason	String	同意/拒绝 PK 的原因描述。

返回

0：响应成功；非0：响应失败。

介绍

主播响应其他房间主播的 PK 请求，发起 PK 请求的主播会收到 IMLVBLiveRoomListener.RequestRoomPKCallback 回调通知。

quitRoomPK

退出跨房 PK。

```
abstract void quitRoomPK(final IMLVBLiveRoomListener.QuitRoomPKCallback callback)
```

参数

参数	类型	含义
callback	final IMLVBLiveRoomListener.QuitRoomPKCallback	退出跨房 PK 的结果回调。

介绍

当两个主播中的任何一个退出跨房 PK 状态后，另一个主播会收到 IMLVBLiveRoomListener#onQuitRoomPK(AnchorInfo) 回调通知。

视频相关接口函数

startLocalPreview

开启本地视频的预览画面。

```
abstract void startLocalPreview(boolean frontCamera, TXCloudVideoView view)
```

参数

参数	类型	含义
frontCamera	boolean	YES：前置摄像头；NO：后置摄像头。
view	TXCloudVideoView	承载视频画面的控件。

stopLocalPreview

停止本地视频采集及预览。

```
abstract void stopLocalPreview()
```

startRemoteView

启动渲染远端视频画面。

```
abstract void startRemoteView(final AnchorInfo anchorInfo, final TXCloudVideoView
view, final IMLVBLiveRoomListener.PlayCallback callback)
```

参数

参数	类型	含义
anchorInfo	final AnchorInfo	对方的用户信息。
view	final TXCloudVideoView	承载视频画面的控件。
callback	final IMLVBLiveRoomListener.PlayCallback	播放器监听器。

> **说明**
> 在 onUserVideoAvailable 回调时，调用这个接口。

stopRemoteView

停止渲染远端视频画面。

```
abstract void stopRemoteView(final AnchorInfo anchorInfo)
```

参数

参数	类型	含义
anchorInfo	final AnchorInfo	对方的用户信息。

startScreenCapture

启动录屏。

```
abstract void startScreenCapture()
```

stopScreenCapture

结束录屏。

```
abstract void stopScreenCapture()
```

音频相关接口函数

muteLocalAudio

是否屏蔽本地音频。

```
abstract void muteLocalAudio(boolean mute)
```

参数

参数	类型	含义
mute	boolean	true：屏蔽；false：开启。

muteRemoteAudio

设置指定用户是否静音。

```
abstract void muteRemoteAudio(String userID, boolean mute)
```

参数

参数	类型	含义
userID	String	对方的用户标识。
mute	boolean	true：静音；false：非静音。

muteAllRemoteAudio

设置所有远端用户是否静音。

```
abstract void muteAllRemoteAudio(boolean mute)
```

参数

参数	类型	含义
mute	boolean	true：静音；false：非静音。

摄像头相关接口函数

switchCamera

切换摄像头。

```
abstract void switchCamera()
```

setZoom

设置摄像头缩放因子（焦距）。

```
abstract boolean setZoom(int distance)
```

参数

参数	类型	含义
distance	int	取值范围：1 – 5，当为1的时候为最远视角（正常镜头），当为5的时候为最近视角（放大镜头），这里最大值推荐为5，超过5后视频数据会变得模糊不清。

enableTorch

开关闪光灯。

```
abstract boolean enableTorch(boolean enable)
```

参数

参数	类型	含义
enable	boolean	true：开启；false：关闭。

setCameraMuteImage

主播屏蔽摄像头期间需要显示的等待图片。

```
abstract void setCameraMuteImage(Bitmap bitmap)
```

参数

参数	类型	含义

bitmap	Bitmap	位图。

介绍

当主播屏蔽摄像头，或者由于 App 切入后台无法使用摄像头的时候，我们需要使用一张等待图片来提示观众“主播暂时离开，请不要走开”。

setCameraMuteImage

主播屏蔽摄像头期间需要显示的等待图片。

```
abstract void setCameraMuteImage(final int id)
```

参数

参数	类型	含义
id	final int	设置默认显示图片的资源文件。

介绍

当主播屏蔽摄像头，或者由于 App 切入后台无法使用摄像头的时候，我们需要使用一张等待图片来提示观众“主播暂时离开，请不要走开”。

美颜滤镜相关接口函数

getBeautyManager

获取美颜管理对象 TXBeautyManager。

```
public TXBeautyManager getBeautyManager()
```

通过美颜管理，您可以使用以下功能：

- 设置“美颜风格”、“美白”、“红润”、“大眼”、“瘦脸”、“V脸”、“下巴”、“短脸”、“小鼻”、“亮眼”、“白牙”、“祛眼袋”、“祛皱纹”、“祛法令纹”等美容效果。
- 调整“发际线”、“眼间距”、“眼角”、“嘴形”、“鼻翼”、“鼻子位置”、“嘴唇厚度”、“脸型”。
- 设置人脸挂件（素材）等动态效果。
- 添加美妆。
- 进行手势识别。

setFilter

设置指定素材滤镜特效。

```
abstract void setFilter(Bitmap image)
```

参数

参数	类型	含义
image	Bitmap	指定素材，即颜色查找表图片。注意：**一定要用 png 格式**。

setFilterConcentration

设置滤镜浓度。

```
abstract void setFilterConcentration(float concentration)
```

参数

参数	类型	含义
concentration	float	从0到1，越大滤镜效果越明显，默认取值0.5。

setWatermark

添加水印，height 不用设置，SDK 内部会根据水印宽高比自动计算 height。

```
abstract void setWatermark(Bitmap image, float x, float y, float width)
```

参数

参数	类型	含义
image	Bitmap	水印图片 null 表示清除水印。
x	float	归一化水印位置的 X 轴坐标，取值[0，1]。
y	float	归一化水印位置的 Y 轴坐标，取值[0，1]。
width	float	归一化水印宽度，取值[0，1]。

setGreenScreenFile

设置绿幕文件。

```
abstract boolean setGreenScreenFile(String file)
```

参数

参数	类型	含义
file	String	绿幕文件位置，支持两种方式： • 资源文件放在 assets 目录，path 直接取文件名。 • path 取文件绝对路径。

返回

false：调用失败；true：调用成功。

介绍

目前图片支持 jpg/png，视频支持 mp4/3gp 等 Android 系统支持的格式。

> **说明**
> API 要求18。

setExposureCompensation

调整曝光。

```
abstract void setExposureCompensation(float value)
```

参数

参数	类型	含义
value	float	曝光比例，表示该手机支持最大曝光调整值的比例，取值范围：-1 – 1。负数表示调低曝光，-1是最小值；正数表示调高曝光，1是最大值；0表示不调整曝光。

消息发送接口函数

sendRoomTextMsg

发送文本消息。

```
abstract void sendRoomTextMsg(String message, final
IMLVBLiveRoomListener.SendRoomTextMsgCallback callback)
```

参数

参数	类型	含义
messag	String	文本消息。

e		
callback	final IMLVBLiveRoomListener.SendRoomTextMsgCallback	发送结果回调。

sendRoomCustomMsg

发送自定义文本消息。

```
abstract void sendRoomCustomMsg(String cmd, String message, final IMLVBLiveRoomListener.SendRoomCustomMsgCallback callback)
```

参数

参数	类型	含义
cmd	String	命令字，由开发者自定义，主要用于区分不同消息类型。
message	String	文本消息。
callback	final IMLVBLiveRoomListener.SendRoomCustomMsgCallback	发送结果回调。

背景混音相关接口函数

playBGM

播放背景音乐。

```
abstract boolean playBGM(String path)
```

参数

参数	类型	含义
path	String	背景音乐文件路径。

返回

true：播放成功；false：播放失败。

stopBGM

停止播放背景音乐。

```
abstract void stopBGM()
```

pauseBGM

暂停播放背景音乐。

```
abstract void pauseBGM()
```

resumeBGM

继续播放背景音乐。

```
abstract void resumeBGM()
```

getBGMDuration

获取音乐文件总时长。

```
abstract int getBGMDuration(String path)
```

参数

参数	类型	含义
path	String	音乐文件路径，如果 path 为空，那么返回当前正在播放的 music 时长。

返回
成功返回时长，单位毫秒，失败返回-1。

setMicVolumeOnMixing

设置麦克风的音量大小，播放背景音乐混音时使用，用来控制麦克风音量大小。

```
abstract void setMicVolumeOnMixing(int volume)
```

参数

参数	类型	含义
volume	int	音量大小，100为正常音量，建议值为0 - 200。

setBGMVolume

设置背景音乐的音量大小，播放背景音乐混音时使用，用来控制背景音音量大小。

```
abstract void setBGMVolume(int volume)
```

参数

参数	类型	含义
volume	int	音量大小，100为正常音量，建议值为0 - 200，如果需要调大背景音量可以设置更大的值。

setReverbType

设置混响效果。

```
abstract void setReverbType(int reverbType)
```

参数

参数	类型	含义
reverbType	int	混响类型，详见： • TXLiveConstants#REVERB_TYPE_0（关闭混响）。 • TXLiveConstants#REVERB_TYPE_1（KTV）。 • TXLiveConstants#REVERB_TYPE_2（小房间）。 • TXLiveConstants#REVERB_TYPE_3（大会堂）。 • TXLiveConstants#REVERB_TYPE_4（低沉）。 • TXLiveConstants#REVERB_TYPE_5（洪亮）。 • XLiveConstants#REVERB_TYPE_6（磁性）。

setVoiceChangerType

设置变声类型。

```
abstract void setVoiceChangerType(int voiceChangerType)
```

参数

参数	类型	含义

voiceChangerType	int	变声类型，详见 TXVoiceChangerType。

setBgmPitch

设置背景音乐的音调。

```
abstract void setBgmPitch(float pitch)
```

参数

参数	类型	含义
pitch	float	音调，0为正常音调，范围：-1 - 1。

介绍

该接口用于混音处理，例如将背景音乐与麦克风采集到的声音混合后播放。

IMLVBLiveRoomListener

最近更新时间：2023-09-19 18:43:54

功能

MLVBLiveRoom 事件回调。

介绍

包括房间关闭、Debug 事件信息和出错说明等。

通用事件回调

onError

错误回调。

```
void onError(int errCode, String errMsg, Bundle extraInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errMsg	String	错误信息。
extraInfo	Bundle	额外信息，如错误发生的用户，一般不需要关注，默认是本地错误。

介绍

SDK 不可恢复的错误，一定要监听，并分情况给用户适当的界面提示。

onWarning

警告回调。

```
void onWarning(int warningCode, String warningMsg, Bundle extraInfo)
```

参数

参数	类型	含义
warningCode	int	错误码 TRTCWarningCode。

warningMsg	String	警告信息。
extraInfo	Bundle	额外信息，如警告发生的用户，一般不需要关注，默认是本地错误。

onDebugLog

```
void onDebugLog(String log)
```

房间事件回调

onRoomDestroy

房间被销毁的回调。

```
void onRoomDestroy(String roomID)
```

参数

参数	类型	含义
roomID	String	房间 ID。

介绍

主播退房时，房间内的所有用户都会收到此通知。

onAnchorEnter

收到新主播进房通知。

```
void onAnchorEnter(AnchorInfo anchorInfo)
```

参数

参数	类型	含义
anchorInfo	AnchorInfo	新进房用户信息。

介绍

房间内的主播（和连麦中的观众）会收到新主播的进房事件，您可以调用 MLVBLiveRoom#startRemoteView(AnchorInfo， TXCloudVideoView， PlayCallback) 显示该主播的视频画面。

> **说明**
> 直播间里的普通观众不会收到主播加入和退出的通知。

onAnchorExit

收到主播退房通知。

```
void onAnchorExit(AnchorInfo anchorInfo)
```

参数

参数	类型	含义
anchorInfo	AnchorInfo	退房用户信息。

介绍

房间内的主播（和连麦中的观众）会收到新主播的退房事件，您可以调用 MLVBLiveRoom#stopRemoteView(AnchorInfo) 关闭该主播的视频画面。

> **说明**
> 直播间里的普通观众不会收到主播加入和退出的通知。

onAudienceEnter

收到观众进房通知。

```
void onAudienceEnter(AudienceInfo audienceInfo)
```

参数

参数	类型	含义
audienceInfo	AudienceInfo	进房观众信息。

onAudienceExit

收到观众退房通知。

```
void onAudienceExit(AudienceInfo audienceInfo)
```

参数

参数	类型	含义
audienceInfo	AudienceInfo	退房观众信息。

onRequestJoinAnchor

主播收到观众连麦请求时的回调。

```
void onRequestJoinAnchor(AnchorInfo anchorInfo, String reason)
```

参数

参数	类型	含义
anchorInfo	AnchorInfo	观众信息。
reason	String	连麦原因描述。

onKickoutJoinAnchor

连麦观众收到被踢出连麦的通知。

```
void onKickoutJoinAnchor()
```

介绍

连麦观众收到被主播踢除连麦的消息，您需要调用 MLVBLiveRoom#kickoutJoinAnchor(String) 来退出连麦。

onRequestRoomPK

收到请求跨房 PK 通知。

```
void onRequestRoomPK(AnchorInfo anchorInfo)
```

参数

参数	类型	含义
anchorInfo	AnchorInfo	发起跨房连麦的主播信息。

介绍

主播收到其他房间主播的 PK 请求，如果同意 PK ，您需要调用 MLVBLiveRoom#startRemoteView(AnchorInfo， TXCloudVideoView， PlayCallback) 接口播放邀约主播的流。

onQuitRoomPK

收到断开跨房 PK 通知。

```
void onQuitRoomPK(AnchorInfo anchorInfo)
```

消息事件回调

onRecvRoomTextMsg

收到文本消息。

```
void onRecvRoomTextMsg(String roomID, String userID, String userName, String
userAvatar, String message)
```

参数

参数	类型	含义
roomID	String	房间 ID。
userID	String	发送者 ID。
userName	String	发送者昵称。
userAvatar	String	发送者头像。
message	String	文本消息。

onRecvRoomCustomMsg

收到自定义消息。

```
void onRecvRoomCustomMsg(String roomID, String userID, String userName, String
userAvatar, String cmd, String message)
```

参数

参数	类型	含义

roomID	String	房间 ID。
userID	String	发送者 ID。
userName	String	发送者昵称。
userAvatar	String	发送者头像。
cmd	String	自定义 cmd。
message	String	自定义消息内容。

LoginCallback

功能

登录结果回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

GetRoomListCallback

功能

获取房间列表回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess(ArrayList< RoomInfo > roomInfoList)
```

参数

参数	类型	含义
roomInfoList	ArrayList< RoomInfo >	房间列表。

GetAudienceListCallback

功能

获取观众列表回调接口。

介绍

观众进房时，后台会将其信息加入观众列表中，观众列表最大保存30名观众信息。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess(ArrayList< AudienceInfo > audienceInfoList)
```

参数

参数	类型	含义
audienceInfoList	ArrayList< AudienceInfo >	观众列表。

CreateRoomCallback

功能

创建房间的结果回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess(String RoomID)
```

参数

参数	类型	含义
RoomID	String	房间号标识。

EnterRoomCallback

功能

进入房间的结果回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

ExitRoomCallback

功能

离开房间的结果回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

RequestJoinAnchorCallback

功能

观众请求连麦的结果回调接口。

onAccept

主播接受连麦。

```
void onAccept()
```

onReject

主播拒绝连麦。

```
void onReject(String reason)
```

参数

参数	类型	含义
reason	String	拒绝原因。

onTimeOut

请求超时。

```
void onTimeOut()
```

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

JoinAnchorCallback

功能

进入连麦的结果回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码 RequestRoomPKCallback。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

QuitAnchorCallback

功能

退出连麦的结果调用接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

RequestRoomPKCallback

功能

请求跨房 PK 的结果回调接口。

onAccept

主播接受连麦。

```
void onAccept(AnchorInfo anchorInfo)
```

参数

参数	类型	含义
anchorInfo	AnchorInfo	被邀请 PK 主播的信息。

onReject

拒绝 PK。

```
void onReject(String reason)
```

参数

参数	类型	含义
reason	String	拒绝原因。

onTimeOut

请求超时。

```
void onTimeOut()
```

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

QuitRoomPKCallback

功能
退出跨房 PK 的结果回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

PlayCallback

功能
播放器回调接口。

onBegin

开始回调。

```
void onBegin()
```

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onEvent

其他事件回调。

```
void onEvent(int event, Bundle param)
```

参数

参数	类型	含义
event	int	事件 ID。
param	Bundle	事件附加信息。

SendRoomTextMsgCallback

功能

发送文本消息回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

SendRoomCustomMsgCallback

功能

发送自定义消息回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

SetCustomInfoCallback

功能

设置自定义信息回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onSuccess

成功回调。

```
void onSuccess()
```

GetCustomInfoCallback

功能
获取自定义信息回调接口。

onError

错误回调。

```
void onError(int errCode, String errInfo)
```

参数

参数	类型	含义
errCode	int	错误码。
errInfo	String	错误信息。

onGetCustomInfo

获取自定义信息的回调。

```
void onGetCustomInfo(Map< String, Object > customInfo)
```

参数

参数	类型	含义
customInfo	Map< String, Object >	自定义信息。

示例

最近更新时间：2023-09-19 18:43:54

针对开发者的接入反馈的高频问题，腾讯云提供有直观易懂的 API-Example 工程，方便开发者可以快速的了解相关 API 的使用，欢迎使用。

工程地址

所属平台	GitHub 地址
Android	Github

目录说明

在这个示例项目中包含了以下场景：

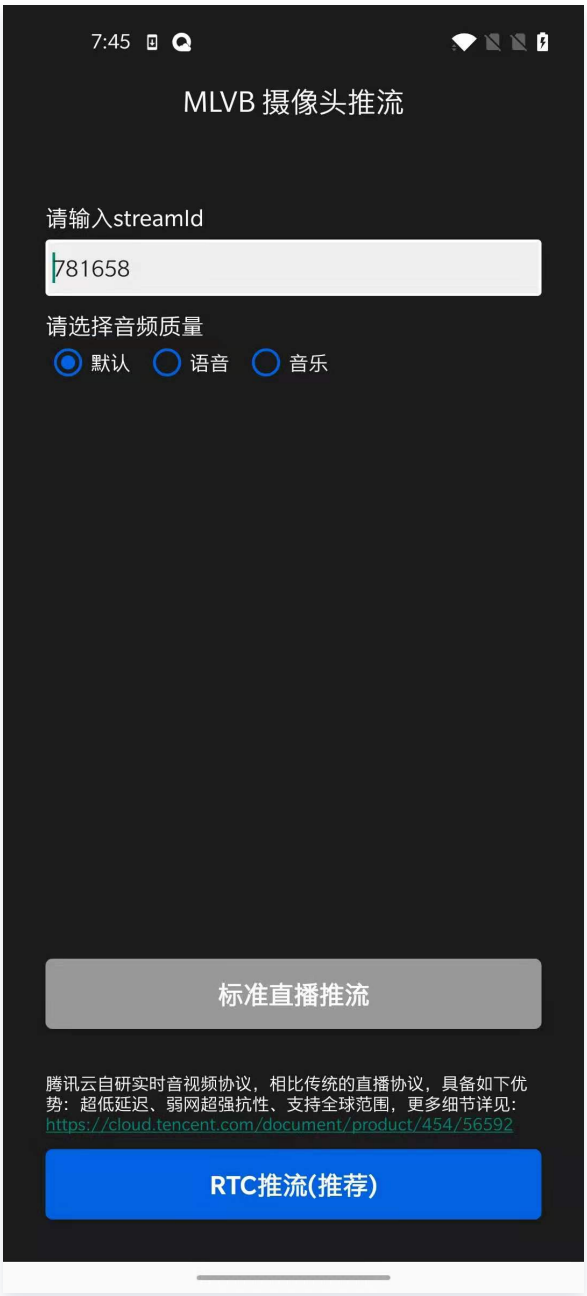
- **基础功能：**
  - 摄像头推流
  - 录屏推流
  - 直播拉流
  - 连麦互动
  - 连麦 PK
- **进阶功能：**
  - 动态切换渲染组件
  - 自定义视频采集
  - 第三方美颜
  - RTC 连麦 + 超低延时播放

> **说明**
> 目前的工程结构跟标准的 Android Studio 工程在名称大小写上可能有略微的差异，主要目的是方便大家在网页上看到此工程时，名称意义更加清晰。

应用图示

主页示例	推流示例	拉流示例


7:45
腾讯云MLVB API Example
基础功能
摄像头推流
录屏推流
直播拉流
连麦互动
PK互动
进阶功能
动态切换渲染组件
自定义视频采集
第三方美颜
RTC连麦+超低延时播放


7:45
MLVB 摄像头推流
请输入streamId
781658
请选择音频质量
默认
语音
音乐
标准直播推流
腾讯云自研实时音视频协议，相比传统的直播协议，具备如下优势：超低延迟、弱网超强抗性、支持全球范围，更多细节详见：
https://cloud.tencent.com/document/product/454/56592
RTC推流(推荐)

微信小程序
<live-pusher> 标签

最近更新时间：2023-08-29 11:12:12

<live-pusher> 是小程序内部用于支持音视频上行能力的功能标签，本文主要介绍该标签的使用方法。

版本支持

- 微信 App iOS 最低版本要求：6.5.21。
- 微信 App Android 最低版本要求：6.5.19。
- 小程序基础库最低版本要求：1.7.0。

> **说明**
> 通过 wx.getSystemInfo 可以获取当前基础库版本信息。

使用限制

出于政策和合规的考虑，微信暂时没有放开所有小程序对 <live-pusher> 和 <live-player> 标签的支持：

- 注册非个人主体类型的微信小程序。**微信小程序的主体必须为非个人主体类型，否则无法使用直播功能。**
- 企业账号的小程序暂时只开放如下表格中的类目：

主类目	子类目	小程序内容场景
社交	直播	涉及娱乐性质，如明星直播、生活趣事直播和宠物直播等。选择该类目后首次提交代码审核，需经当地互联网主管机关审核确认，预计审核时长7天左右
教育	在线视频课程	网课、在线培训、讲座等教育类直播
医疗	互联网医院，公立医院	问诊、大型健康讲座等直播
金融	银行、信托、基金、证券/期货、证券、期货投资咨询、保险、征信业务、新三板信息服务平台、股票信息服务平台（港股/美股）、消费金融	金融产品视频客服理赔、金融产品推广直播等
汽车	汽车预售服务	汽车预售、推广直播
政府主体账号	-	政府相关工作推广直播、领导讲话直播等

工具	视频客服	不涉及以上几类内容的一对一视频客服服务，如企业售后一对一视频服务等
IT 科技	多方通信；音视频设备	为多方提供电话会议/视频会议等服务；智能家居场景下控制摄像头

- 符合类目要求的小程序，需要在小程序管理后台的 **开发 > 接口设置** 中自助开通推拉流标签的使用权限，如下图所示：

> **注意**
> 如果以上设置都正确，但小程序依然不能正常工作，可能是微信内部的缓存没更新，请删除小程序并重启微信后，再进行尝试。

属性定义

属性名	类型	默认值	说明

url	String	-	用于音视频上行的推流 URL
mode	String	RTC	SD，HD，FHD，RTC
autopush	Boolean	false	是否自动启动推流
muted	Boolean	false	是否静音
enable-camera	Boolean	true	开启\关闭摄像头
auto-focus	Boolean	true	手动\自动对焦
orientation	String	vertical	vertical，horizontal
beauty	Number	0	美颜指数，取值 0 - 9，数值越大效果越明显
whiteness	Number	0	美白指数，取值 0 - 9，数值越大效果越明显
aspect	String	9：16	3：4，9：16
zoom	Boolean	false	是否正常焦距，true 表示将摄像头放大
device-position	String	front	front 前置摄像头，back 后置摄像头
min-bitrate	Number	200	最小码率，该数值决定了画面最差的清晰度表现
max-bitrate	Number	1000	最大码率，该数值决定了画面最好的清晰度表现
audio-quality	String	low	low 适合语音通话，high 代表高音质
waiting-image	String	-	当微信切到后台时的垫片图片
waiting-image-hash	String	-	当微信切到后台时的垫片图片的校验值
background-mute	Boolean	false	当微信切到后台时是否禁用声音采集
bindstatechange	String	-	用于指定一个 javascript 函数来接收音视频事件

debug	Boolean	false	是否开启调试模式

示例代码

```
<view id='video-box'>
 <live-pusher
     id="pusher"
     mode="RTC"
     url="{{pusher.push_url}}"
     autopush='true'
     bindstatechange="onPush">
 </live-pusher>
</view>
```

属性详解

- **url**
  用于音视频上行的推流 URL，以 `rtmp://` 协议前缀打头，腾讯云推流 URL 的获取方法见 快速获取 URL 文档。

  > **说明**
  > 小程序内部使用的 RTMP 协议是支持 UDP 加速的版本，在同样网络条件下，UDP 版本的 RTMP 会比开源版本的有更好的上行速度和抗抖动能力。

- **mode**
  SD、HD 和 FHD 主要用于直播类场景，例如赛事直播、在线教育、远程培训等等。SD、HD 和 FHD 分别对应三种默认的清晰度。该模式下，小程序会更加注重清晰度和观看的流畅性，不会过分强调低延迟，也不会为了延迟牺牲画质和流畅性。

  RTC 则主要用于双向视频通话或多人视频通话场景，例如金融开会、在线客服、车险定损、培训会议等。该模式下，小程序会更加注重降低点到点的时延，也会优先保证声音的质量，在必要的时候会对画面清晰度和画面的流畅性进行一定的缩水。
- **orientation 和 aspect**
  横屏（horizontal）模式还是竖屏（vertical）模式，默认是竖屏模式，即 home 键朝下。这时，小程序推出的画面的宽高比是3：4或者9：16这两种竖屏宽高比的画面，也就是宽 < 高。如果改成横屏模式，小程序推出的画面宽高比即变为4：3或者16：9这种横屏宽高比的画面，也就是宽 > 高。

  具体的宽高比是由 aspect 决定的，默认是9：16，也可以支持3：4。这是在 orientation 的属性值为 vertical 的情况下。如果 orientation 的属性值为 horizontal，那么3：4的效果等价于4：3，9：16的效果等价于16：9。
- **min-bitrate 和 max-bitrate**

这里首先要科普一个概念 —— 视频码率，指视频编码器每秒钟输出的视频数据的多少。在视频分辨率确定的情况下，视频码率越高，即每秒钟输出的数据越多，相应的画质也就越好。

所以 min-bitrate 和 max-bitrate 这两个属性，分别用于决定输出画面的最低清晰度和最高清晰度。这两个数值并非越大越好，因为用户的网络上行不是无限好的。但也不是越小越好，因为实际应用场景中，清晰与否是用户衡量产品体验的一个重要指标。具体的数值设定我们会在**“参数设置”**部分详细介绍。

小程序内部会自动处理好分辨率和码率的关系，例如2Mbps的码率，小程序会选择720p的分辨率进行匹配，而300kbps的码率下，小程序则会选择较低的分辨率来提高编码效率。所以您只需要关注 min-bitrate 和 max-bitrate 这一对参数就可以掌控画质了。

- **waiting-image 和 waiting-image-hash**

  出于用户隐私的考虑，在微信切到后台以后，小程序希望停止摄像头的画面采集。但是对于另一端的用户而言，画面会变成黑屏或者冻屏（停留在最后一帧），这种体验是非常差的。为了解决这个问题，我们引入了 waiting-image 属性，您可以设置一张有 “稍候” 含义的图片（waiting-image 是该图片的 URL，waiting-image-hash 则是该图片对应的 md5 校验值）。当微信切到后台以后，小程序会使用该图片作为摄像头画面的替代，以极低的流量占用维持视频流3分钟时间。

- **debug**

  调试音视频相关功能，如果没有很好的工具会是一个噩梦，所以小程序为 live-pusher 标签支持了 debug 模式，开始 debug 模式之后，原本用于渲染视频画面的窗口上，会显示一个半透明的 log 窗口，用于展示各项音视频指标和事件，降低您调试相关功能的难度，具体使用方法我们在 FAQ 中有详细说明。

参数设置

这么多参数，具体要怎样设置才比较合适呢？我们给出如下建议值：

场景	mode	min-bitrate	max-bitrate	audio-quality	说明
标清直播	SD	300kbps	800kbps	high	窄带场景，例如户外或者网络不稳定的情况下适用
高清直播	HD	600kbps	1200kbps	high	目前主流的 App 所采用的参数设定，普通直播场景推荐使用这一档
超清直播	FHD	600kbps	1800kbps	high	对清晰度要求比较苛刻的场景，普通手机观看使用 HD 即可
视频客服（用户）	RTC	200kbps	500kbps	high	这是一种声音为主，画面为辅的场景，所以画质不要设置的太高
车险定损（车主）	RTC	200kbps	1200kbps	high	由于可能要看车况详情，画质上限会设置的高一些
多人会议	RT	200k	1000	high	主讲人画质可以适当高一些，参与的质量可以

（主讲）	C	bps	kbps		设置的低一些
多人会议（参与）	RTC	150kbps	300kbps	low	作为会议参与者，不需要太高的画质和音质

> **说明**
> 如果不是对带宽特别没有信心的应用场景，audio-quality 选项请不要选择 low，其音质和延迟感都要会比 high 模式差很多。

对象操作

对象	说明
wx.createLivePusherContext()	通过 wx.createLivePusherContext() 可以将 <live-pusher> 标签和 javascript 对象关联起来，之后即可操作该对象
start	开始推流，如果 <live-pusher> 的 autopush 属性设置为 false（默认值），那么就可以使用 start 来手动开始推流
stop	停止推流
pause	暂停推流
resume	恢复推流，请与 pause 操作配对使用
switchCamera	切换前后摄像头
snapshot	推流截图，截图大小跟组件的大小一致。截图成功图片的临时路径为 ret.tempImagePath

```
var pusher = wx.createLivePusherContext('pusher');
pusher.start({
    success: function(ret){
        console.log('start push success!')
    }
    fail: function(){
        console.log('start push failed!')
    }
    complete: function(){
        console.log('start push complete!')
    }
});
```

内部事件

通过 <live-pusher> 标签的 bindstatechange 属性可以绑定一个事件处理函数，该函数可以监听推流模块的内部事件和异常通知。

1. 常规事件

code	事件定义	含义说明
1001	PUSH_EVT_CONNECT_SUCC	已经成功连接到云端服务器
1002	PUSH_EVT_PUSH_BEGIN	与服务器握手完毕，一切正常，准备开始上行推流
1003	PUSH_EVT_OPEN_CAMERA_SUCC	已成功启动摄像头，摄像头被占用或者被限制权限的情况下无法打开

2. 严重错误

code	事件定义	含义说明
-1301	PUSH_ERR_OPEN_CAMERA_FAIL	打开摄像头失败
-1302	PUSH_ERR_OPEN_MIC_FAIL	打开麦克风失败
-1303	PUSH_ERR_VIDEO_ENCODE_FAIL	视频编码失败
-1304	PUSH_ERR_AUDIO_ENCODE_FAIL	音频编码失败
-1305	PUSH_ERR_UNSUPPORTED_RESOLUTION	不支持的视频分辨率
-1306	PUSH_ERR_UNSUPPORTED_SAMPLERATE	不支持的音频采样率
-1	PUSH_ERR_NET_DISCON	网络断连，且经三次重连无效，可以放弃，更多重试请自行

3007	NECT	重启推流

3. 警告事件

内部警告并非不可恢复的错误，小程序内部的音视频 SDK 会启动相应的恢复措施，警告的目的主要用于提示开发者或者最终用户，例如：

- **PUSH_WARNING_NET_BUSY**
  上行网速不给力，建议提示用户改善当前的网络环境，例如让用户离家里的路由器近一点，或者切到 Wi-Fi 环境下再使用。
- **PUSH_WARNING_SERVER_DISCONNECT**
  请求被后台拒绝了，出现这个问题一般是由于 URL 里的 txSecret 计算错了，或者是 URL 被其他人占用了（跟播放不同，一个推流 URL 同时只能有一个用户使用）。
- **PUSH_WARNING_HANDUP_STOP**
  当用户单击小程序右上角的圆圈或者返回按钮时，微信会将小程序挂起，此时 <live-pusher> 会抛出5000这个事件。

code	事件定义	含义说明
1101	PUSH_WARNING_NET_BUSY	上行网速不够用，建议提示用户改善当前的网络环境
1102	PUSH_WARNING_RECONNECT	网络断连，已启动重连流程（重试失败超过三次会放弃）
1103	PUSH_WARNING_HW_ACCELERATION_FAIL	硬编码启动失败，自动切换到软编码
1107	PUSH_WARNING_SWITCH_SWENC	由于机器性能问题，自动切换到硬件编码
3001	PUSH_WARNING_DNS_FAIL	DNS 解析失败，启动重试流程
3002	PUSH_WARNING_SEVER_CONN_FAIL	服务器连接失败，启动重试流程
3003	PUSH_WARNING_SHAKE_FAIL	服务器握手失败，启动重试流程
3004	PUSH_WARNING_SERVER_DISCONNECT	RTMP 服务器主动断开，请检查推流地址的合法性或防盗链有效期
30	PUSH_WARNING_READ_WRITE_F	RTMP 读/写失败，将会断开连接

05	AIL	
5000	PUSH_WARNING_HANDUP_STOP	小程序被用户挂起，停止推流

4. 示例代码

```
Page({
   onPush: function(ret) {
      if(ret.detail.code == 1002) {
        console.log('推流成功了',ret);
      }
   },

  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
  //...
  }
})
```

<live-player> 标签

最近更新时间：2023-08-10 11:50:02

<live-player> 是小程序内部用于支持音视频下行（播放）能力的功能标签，本文主要介绍该标签的使用方法。

版本支持

- 微信 App iOS 最低版本要求：6.5.21 。
- 微信 App Android 最低版本要求：6.5.19。
- 小程序基础库最低版本要求：1.7.0。

> **说明**
> 通过 wx.getSystemInfo 可以获取当前基础库版本信息。

使用限制

出于政策和合规的考虑，微信暂时没有放开所有小程序对 <live-pusher> 和 <live-player> 标签的支持：

- 注册非个人主体类型的微信小程序。**微信小程序的主体必须为非个人主体类型，否则无法使用直播功能。**
- 企业账号的小程序暂时只开放如下表格中的类目：

主类目	子类目	小程序内容场景
社交	直播	涉及娱乐性质，如明星直播、生活趣事直播和宠物直播等。选择该类目后首次提交代码审核，需经当地互联网主管机关审核确认，预计审核时长7天左右
教育	在线视频课程	网课、在线培训、讲座等教育类直播
医疗	互联网医院，公立医院	问诊、大型健康讲座等直播
金融	银行、信托、基金、证券/期货、证券、期货投资咨询、保险、征信业务、新三板信息服务平台、股票信息服务平台（港股/美股）、消费金融	金融产品视频客服理赔、金融产品推广直播等
汽车	汽车预售服务	汽车预售、推广直播
政府主体账号	-	政府相关工作推广直播、领导讲话直播等
工具	视频客服	不涉及以上几类内容的一对一视频客服服务，如企业售后一对一视频服务等

IT 科技	多方通信；音视频设备	为多方提供电话会议/视频会议等服务；智能家居场景下控制摄像头

- 符合类目要求的小程序，需要在小程序管理后台的 **设置** > **接口设置** 中自助开通该组件权限，如下图所示：

> **注意**
> 如果以上设置都正确，但小程序依然不能正常工作，可能是微信内部的缓存没更新，请删除小程序并重启微信后，再进行尝试。

属性定义

属性名	类型	默认值	说明
src	String	-	用于音视频下行的播放 URL，支持 RTMP、FLV 等协议
mode	String	live	live，RTC
autoplay	Boolean	false	是否自动播放
muted	Boolean	false	是否静音
orientation	String	vertical	vertical，horizontal

object-fit	String	contain	contain，fillCrop
background-mute	Boolean	false	当微信切到后台时，是否关闭播放声音
min-cache	Number	1	最小缓冲延迟， 单位：秒
max-cache	Number	3	最大缓冲延迟， 单位：秒
bindstatechange	EventHandler	-	用于指定一个 javascript 函数来接受播放器事件
bindfullscreenchange	EventHandler	-	用于指定一个 javascript 函数来接受全屏事件
debug	Boolean	false	是否开启调试模式

示例代码

```
<view id='video-box'>
   <live-player
      wx:for="{{player}}"
      id="player_{{index}}"
      mode="RTC"
      object-fit="fillCrop"
      src="{{item.playUrl}}"
      autoplay='true'
      bindstatechange="onPlay">
   </live-player>
 </view>
```

超低时延

<live-player> 的 RTC 模式支持500ms以内的超低时延链路，可以应用在视频通话和远程遥控等场景中，要使用超低时延播放，需要注意如下几点：

- 推流端如果是微信小程序，请使用 <live-pusher> 的 RTC 模式。
- 推流端如果是 iOS 或者 Android SDK，请使用 setVideoQuality 的 MAIN_PUBLISHER 模式。
- 推流端如果是 Windows，请不要使用 OBS，延时太高，可以使用我们的 Windows SDK 。
- <live-player> 的 min-cache 和 max-cache 请不要自行设置，使用默认值。
- 播放地址请使用超低延时播放地址，也就是带了防盗链签名的 `rtmp://` 地址，如下：

对比项目	示例	时延

普通直播 URL	rtmp://3891.liveplay.myqcloud.com/live/3891_test_clock_for_rtmpacc	>2s
超低延时 URL	rtmp://3891.liveplay.myqcloud.com/live/3891_test_clock_for_rtmpacc?bizid=bizid&txTime=5FD4431C&txSecret=20e6d865f462dff61ada209d53c71cf9	< 500ms

属性详解

- **src**
  用于音视频下行的播放 URL，支持 RTMP 协议（URL 以 `rtmp://` 打头）和 FLV 协议（URL 以 `http://` 打头且以 `.flv` 结尾），腾讯云推流 URL 的获取方法见 如何生成推流 URL 。

  > **说明**
  > <live-player> 标签是不支持 HLS（m3u8）协议的，因为 <video> 已经支持 HLS（m3u8）播放协议了。但直播观看不推荐使用 HLS（m3u8）协议，延迟要比 RTMP 和 FLV 协议高一个数量级。

- **mode**
  live 模式主要用于直播类场景，例如赛事直播、在线教育、远程培训等等。该模式下，小程序内部的模块会优先保证观看体验的流畅，通过调整 min-cache 和 max-cache 属性，您可以调节观众（播放）端所感受到的时间延迟的大小，文档下面会详细介绍这两个参数。

  RTC 则主要用于双向视频通话或多人视频通话场景，例如金融开会、在线客服、车险定损、培训会议等等。在此模式下，对 min-cache 和 max-cache 的设置不会起作用，因为小程序内部会自动将延迟控制在一个很低的水平（500ms左右）。
- **min-cache 和 max-cache**
  这两个参数分别用于指定观看端的最小缓冲时间和最大缓冲时间。所谓**缓冲时间**，是指播放器为了缓解网络波动对观看流畅度的影响而引入的一个“蓄水池”，当来自网络的数据包出现卡顿甚至停滞的时候，“蓄水池”里的紧急用水可以让播放器还能坚持一小段时间，只要在这个短暂的时间内网速恢复正常，播放器就可以源源不断地渲染出流畅而平滑的视频画面。

  “蓄水池”里的水越多，抗网络波动的能力就越强，但代价就是观众端的延迟就越大，所以要在不同的场景下，使用不同的配置来达到体验上的平衡：
  - 码率比较低（1Mbps左右，画面以人物为主）的直播流，min-cache = 1，max-cache = 3 较合适。
  - 码率比较高（2Mbps - 3Mbps的高清游戏画面为主）的直播流，min-cache = 3，max-cache = 5 较合适。

  RTC 模式下这两个参数是无效的。
- **orientation**
  画面渲染角度，horizontal 代表是原始画面方向，vertical 代表向右旋转 90度。
- **object-fit**

画面填充模式，contain 代表把画面显示完成，但如果视频画面的宽高比和 <live-player> 标签的宽高比不一致，那么您将看到黑边。fillCrop 代表把屏幕全部撑满，但如果视频画面的宽高比和 <live-player> 标签的宽高比不一致，那么画面中多余的部分会被裁剪掉。

- **background-mute**
  微信切到后台以后是否继续播放声音，用于避免锁屏对于当前小程序正在播放的视频内容的影响。
- **sound-mode**设置播放模式，可设值为：ear 与 speaker，ear 代表使用听筒播放， speaker 代表使用扬声器，默认为扬声器。
- **debug**
  调试音视频相关功能，如果没有很好的工具会是一个噩梦，所以小程序为 live-pusher 标签支持了 debug 模式，开始 debug 模式之后，原本用于渲染视频画面的窗口上，会显示一个半透明的 log 窗口，用于展示各项音视频指标和事件，降低您调试相关功能的难度，具体使用方法我们在 FAQ 中有详细说明。

对象操作

- **wx.createLivePlayerContext()**
  通过 wx.createLivePlayerContext() 可以将 <live-player> 标签和 javascript 对象关联起来，之后即可操作该对象。
- **play**
  开始播放，如果 <live-player> 的 autoplay 属性设置为 false（默认值），那么就可以使用 play 来手动启动播放。
- **stop**
  停止播放。
- **pause**暂停播放，停留在最后画面。
- **resume**继续播放，与 pause 成对使用。
- **mute**
  设置静音。
- **requestFullScreen**
  进入全屏幕。
- **exitFullScreen**
  退出全屏幕。

```
var player = wx.createLivePlayerContext('pusher');
player.requestFullScreen({
   success: function(){
         console.log('enter full screen mode success!')
      }
      fail: function(){
         console.log('enter full screen mode failed!')
      }
      complete: function(){
```

```
        console.log('enter full screen mode complete!')
    }
});
```

内部事件

通过 <live-player> 标签的 bindstatechange 属性可以绑定一个事件处理函数，该函数可以监听推流模块的内部事件和异常通知。

1. 关键事件

code	事件定义	含义说明
2001	PLAY_EVT_CONNECT_SUCC	已经连接到云端服务器
2002	PLAY_EVT_RTMP_STREAM_BEGIN	服务器开始传输音视频数据
2003	PLAY_EVT_RCV_FIRST_I_FRAME	网络接收到首段音视频数据
2004	PLAY_EVT_PLAY_BEGIN	视频播放开始，可以在收到此事件之前先用默认图片代表等待状态
2006	PLAY_EVT_PLAY_END	视频播放结束
2007	PLAY_EVT_PLAY_LOADING	进入缓冲中状态，此时播放器在等待或积攒来自服务器的数据
-2301	PLAY_ERR_NET_DISCONNECT	网络连接断开，且重新连接亦不能恢复，播放器已停止播放

> **说明**
> 播放 HTTP:// 打头的 FLV 协议地址时，如果观众遇到播放中直播流断开的情况，小程序是不会抛出 PLAY_EVT_PLAY_END 事件的，这是因为 FLV 协议中没有定义停止事件，所以只能通过监听 PLAY_ERR_NET_DISCONNECT 来替代之。

2. 警告事件

内部警告并非不可恢复的错误，小程序内部的音视频 SDK 会启动相应的恢复措施，警告的目的主要用于提示开发者或者最终用户，例如：

code	事件定义	含义说明
2101	PLAY_WARNING_VIDEO_DECODE_FAIL	当前视频帧解码失败
2102	PLAY_WARNING_AUDIO_DECODE_FAIL	当前音频帧解码失败
2103	PLAY_WARNING_RECONNECT	网络断连，已启动自动重连恢复（重连超过三次就直接抛送 PLAY_ERR_NET_DISCONNECT 了）
2104	PLAY_WARNING_RECV_DATA_LAG	视频流不太稳定，可能是观看者当前网速不充裕
2105	PLAY_WARNING_VIDEO_PLAY_LAG	当前视频播放出现卡顿
2106	PLAY_WARNING_HW_ACCELERATION_FAIL	硬解启动失败，采用软解
2107	PLAY_WARNING_VIDEO_DISCONTINUITY	当前视频帧不连续，视频源可能有丢帧，可能会导致画面花屏
3001	PLAY_WARNING_DNS_FAIL	DNS解析失败（仅播放 RTMP:// 地址时会抛送）
3002	PLAY_WARNING_SEVER_CONN_FAIL	服务器连接失败（仅播放 RTMP:// 地址时会抛送）
3003	PLAY_WARNING_SHAKE_FAIL	服务器握手失败（仅播放 RTMP:// 地址时会抛送）

3. 示例代码

```
Page({
  onPlay: function(ret) {
    if(ret.detail.code == 2004) {
          console.log('视频播放开始',ret);
    }
  },

  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
```

```
    //...
    }
})
```

特别说明

1. <live-player> 组件是由客户端创建的原生组件，它的层级是最高的，可以使用 <cover-view> 和 <cover-image> 覆盖在上面。
2. 请勿在 <scroll-view> 中使用 <live-player> 组件。

<mlvb-live-room> 组件

最近更新时间：2023-05-26 10:51:00

属性定义

<mlvb-live-room> 组件包含如下属性：

属性	类型	值	说明
template	String	'float'	必要，标识组件使用的界面模板。
roomID	String	自定义	可选，房间号（role = audience 时，roomID 不能为空）。
roomName	String	自定义	可选，房间名。
role	String	‘anchor’，'audience'	必要，anchor 代表主播，audience 代表观众。
pureaudio	Boolean	true，false	可选，默认 false，是否开启纯音频推流。
beauty	Number	0 – 5	可选，默认5，美颜级别0 – 5。
muted	Boolean	true，false	可选，默认 false，是否静音。
debug	Boolean	true，false	可选，默认 false，是否打印推流 debug 信息。
bindRoomEvent	function	–	必要，监听 mlvb-live-room 组件返回的事件。

操作接口

<mlvb-live-room> 组件包含如下操作接口，您需要先通过 selectComponent 获取 <mlvb-live-room> 标签的引用，之后就可以进行相应的操作。

> **注意**
> 后台接口限制并发为每秒100次请求，若您有高并发请求请提前 联系我们 处理，避免影响服务调用。

函数名	说明

start()	启动。
pause()	暂停。
resume()	恢复。
stop()	停止。
requestJoinAnchor()	请求连麦，适用于 audience。
respondJoinAnchor(agree:Boolean, audience:Object)	同意连麦，适用于 anchor。
switchCamera()	切换摄像头。
sendTextMsg(text:String)	发送文本消息。

```
var liveroom = this.selectComponent("#id_liveroom");
liveroom.pause();
```

事件通知

<mlvb-live-room> 标签通过 onRoomEvent 返回内部事件，事件参数格式如下：

```
"detail": {
  "tag": "事件tag标识，具有唯一性",
  "code": "事件代码",
  "detail": "对应事件的详细参数"
}
```

示例代码

```
// Page.json 文件
"usingComponents": {
    "mlvb-live-room": "/pages/components/mlvb-live-room/mlvbliveroomview"
}

// Page.wxml 文件
<mlvb-live-room id="id_liveroom"
        roomID="{{roomID}}"
        roomName="{{roomName}}"
        template="float"
        beauty="{{beauty}}"
        muted="{{muted}}"
```

```
        role="{{role}}"
        debug="{{debug}}"
        bindRoomEvent="onRoomEvent">
</mlvb-live-room>


// Page.js 文件
Page({
    data: {
        //...
        role: 'anchor', // 角色视您的业务场景而定
        roomID: '',
        roomName: '',
        beauty: 3,
        muted: false,
        debug: false,
    },
    //...
    onRoomEvent: function(e){
        switch(e.detail.tag){
            case 'roomClosed': {
                //房间已经关闭
                break;
            }
            case 'error': {
                //发生错误
                var code = e.detail.code;
                var detail = e.detail.detail;
                break;
            }
            case 'recvTextMsg': {
                //收到文本消息
                var text = e.detail.detail;
                break;
            }
            case 'requestJoinAnchor': {
                //主播收到来自观众的连麦请求
                var audience = e.detail;
                var name = audience.userName;
                var id = audience.userID;
                // 允许请求
                liveroom.respondJoinReq(true, audience)
                break;
            }
            case 'linkOn': {
                //连麦观众在连麦成功时会收到此通知
```

```
            break;
          }
          case 'linkOut': {
            //连麦观众退出连麦时会收到此通知
            break;
          }
        }
    },

  onShow: function () {
  },

  onHide: function () {
  },

  onRead: function() {
     var liveroom = this.selectComponent("#id_liveroom");
     this.setData({
        roomName: '测试',
     });
    liveroom.start();
  },

})
```

Web

概览

最近更新时间：2023-09-19 18:43:54

TXLivePusher

直播推流

腾讯云直播推流器，主要用于桌面端浏览器快直播推流。通过浏览器采集用户的画面和声音，通过 WebRTC 将视频流和音频流传输推送到腾讯云服务端。

功能介绍

API	描述
checkSupport	静态函数，检查浏览器支持性
setRenderView	设置本地视频画面的预览容器
setVideoQuality	设置推流视频质量
setAudioQuality	设置推流音频质量
setProperty	调用 TXLivePusher 的高级 API 接口
startCamera	打开本地摄像头
stopCamera	关闭本地摄像头
startMicrophone	打开麦克风
stopMicrophone	关闭麦克风
startScreenCapture	开启屏幕采集
stopScreenCapture	关闭屏幕采集
startMediaFile	开始采集本地媒体文件流
stopMediaFile	停止采集本地媒体文件流
startPush	开始音视频数据推流
stopPush	停止推送音视频数据

isPushing	当前推流器是否正在推流中
getDeviceManager	获取设备管理对象
setVideoMute	设置是否禁用视频流
setAudioMute	设置是否禁用音频流
setObserver	设置推流器回调
destroy	离开页面或者退出时，清理 SDK 实例

TXLivePusherObserver

直播推流回调通知

腾讯云直播推流的回调通知，回调包括推流器状态，统计信息，警告以及错误信息。

功能介绍

API	描述
onError	直播推流器错误通知
onWarning	直播推流器警告通知
onCaptureFirstAudioFrame	首帧音频采集完成的回调通知
onCaptureFirstVideoFrame	首帧视频采集完成的回调通知
onPushStatusUpdate	推流器连接状态回调通知
onStatisticsUpdate	直播推流器统计数据回调

TXDeviceManager

设备管理

腾讯云设备管理接口，主要用于管理摄像头和麦克风设备。

功能介绍

API	描述
getDevicesList	获取设备列表
setCurrentDevice	设置要使用的设备
getCurrentDevice	获取当前的设备信息
switchDevice	切换设备
switchCamera	切换摄像头设备
switchMicrophone	切换麦克风设备

TXLivePusher

最近更新时间：2023-09-19 18:43:54

直播推流接口主要是通过浏览器采集用户的画面和声音，并通过 WebRTC 进行推送。

checkSupport

静态函数，检查浏览器支持性。

```
static checkSupport(): Promise<SupportResult>;
```

返回

返回 Promise 对象，其中检查结果 `SupportResult` 结构如下：

字段	类型	描述
isWebRTCSupported	boolean	是否支持 WebRTC
isH264EncodeSupported	boolean	是否支持 H264 编码
isH264DecodeSupported	boolean	是否支持 H264 解码
isMediaDevicesSupported	boolean	是否支持获取媒体设备及媒体流
isScreenCaptureSupported	boolean	是否支持屏幕采集
isMediaFileSupported	boolean	是否支持获取本地媒体文件流

setRenderView

设置本地视频画面的预览容器，提供一个 div 节点，本地采集的视频会在容器里渲染。

```
setRenderView(container: string | HTMLDivElement): void;
```

参数

`container`：容器的 ID 或者 dom 节点。

setVideoQuality

设置推流视频质量，SDK 已经内置了视频质量模板，直接通过预定义的模板来设置推流视频质量。

```
setVideoQuality(quality: string): void;
```

参数

`quality` 预定义的视频质量模板名称。内置的视频质量模板如下所示：

模板名	分辨率（宽 × 高）	帧率（fps）	码率（kbps）
120p	160 × 120	15	200
180p	320 × 180	15	350
240p	320 × 240	15	400
360p	640 × 360	15	800
480p	640 × 480	15	900
720p	1280 × 720	15	1500
1080p	1920 × 1080	15	2000
2K	2560 × 1440	30	4860
4K	3840 × 2160	30	9000

说明

- 由于设备和浏览器的限制，视频分辨率不一定能够完全匹配，在这种情况下，浏览器会自动调整分辨率使其接近对应的分辨率。
- 如果以上视频质量参数（分辨率、帧率和码率）不符合您的要求，您可以通过 `setProperty` 单独设置自定义的值。
- 默认使用 `720p` ，即 `setVideoQuality('720p')` 。

setAudioQuality

设置推流音频质量，SDK 已经内置了音频质量模板，直接通过预定义的模板来设置推流音频质量。

```
setAudioQuality(quality: string): void;
```

参数

`quality` 预定义的音频质量模板名称。内置的音频质量模板如下所示：

模板名	采样率	码率（kbps）
standard	48000	40
high	48000	128

说明

默认使用 `standard` ，即 `setAudioQuality('standard')` 。

setProperty

主要用于调用一些高级功能例如设置视频的分辨率、帧率和码率。

```
setProperty(key: string, value: any): void;
```

参数

- `key` 高级 API 对应的 key。
- `value` 调用 key 所对应的高级 API 时，需要的参数。

说明

目前支持以下高级功能：

Key	Value	描述	示例
setVideoResolution	{width: number; height: number;}	设置视频的分辨率	setProperty('setVideoResolution', { width: 1920, height: 1080 })
setVideoFPS	number	设置视频的帧率	setProperty('setVideoFPS', 25)
setVideoBitrate	number	设置视频的码率	setProperty('setVideoBitrate', 2000)
setAudioSampleRate	number	设置音频的采样率	setProperty('setAudioSampleRate', 44100)
setAudioBitrate	number	设置音频的码率	setProperty('setAudioBitrate', 200)
setConnectRetryCount	number	设置连接重试次数 默认值：3，取值范围[0,10] 当 SDK 与服务器异常断开连接时，SDK 会尝试与服务器重连	setProperty('setConnectRetryCount', 5)
setConnectRetryDelay	number	设置连接重试延迟	setProperty('setConnectRetryDelay', 2)

		默认值：1，单位为秒；取值范围 [0,10] 当 SDK 与服务器异常断开连接时，SDK 会尝试与服务器重连	

> **注意**
> 请在采集流和推流之前进行设置。

startCamera

打开本地摄像头设备。需要用户授权允许浏览器访问摄像头，授权失败或者访问设备失败，会通过回调接口抛出警告通知。

```
startCamera(deviceId?: string): void;
```

参数

`deviceId`：摄像头设备 ID，可选参数，指定打开的摄像头设备。设备 ID 可通过 TXDeviceManager 中的方法 getDevicesList 获取。

stopCamera

关闭本地摄像头设备。

```
stopCamera(): void;
```

startMicrophone

打开麦克风设备。需要用户授权允许浏览器访问麦克风，授权失败或者访问设备失败，会通过回调接口抛出警告通知。

```
startMicrophone(deviceId?: string): void;
```

参数

`deviceId`：麦克风设备 ID，可选参数，指定打开的麦克风设备。设备 ID 可通过 TXDeviceManager 中的方法 getDevicesList 获取。

stopMicrophone

关闭麦克风设备。

```
stopMicrophone(): void;
```

startScreenCapture

开启屏幕采集。需要用户授权允许浏览器访问屏幕，授权失败或者访问设备失败，会通过回调接口抛出警告通知。

```
startScreenCapture(): void;
```

stopScreenCapture

关闭屏幕采集。

```
stopScreenCapture(): void;
```

startMediaFile

开始采集本地媒体文件流。将用户本地的 MP4 视频文件转换成音视频流。

```
startMediaFile(file?: File): void;
```

参数

`file`：可选，本地媒体文件，不传则 javascript 模拟触发文件上传操作。

说明

- 需要先调用 setRenderView 方法设置承载本地视频画面的 div 容器。
- 本地视频文件当前仅支持 MP4 文件。
- safari 浏览器限制，javascript 模拟触发文件上传操作可能不生效，建议手动传入 file 对象。
- 浏览器限制，页面处于未激活状态时（最小化、激活其它标签页），采集文件流会暂停。建议提示用户推流过程中不要切换浏览器标签页。

stopMediaFile

停止采集本地媒体文件流，结束本地文件播放。

```
stopMediaFile(): void;
```

startPush

开始音视频数据推流，建立 WebRTC 连接，往腾讯云服务器推送数据。

```
startPush(pushUrl: string): void;
```

参数

`pushUrl`：WebRTC 推流地址，请填写腾讯云的推流地址。推流地址的格式参见 腾讯云标准直播 URL ，只需要将 RTMP 推流地址前面的 `rtmp://` 替换成 `webrtc://` 即可。

stopPush

停止推送音视频数据，关闭 WebRTC 连接。

```
stopPush(): void;
```

isPushing

查询当前推流器是否正在推流中。

```
isPushing(): boolean;
```

返回

true：正在推流；false：未推流。

getDeviceManager

获取设备管理对象。通过设备管理，可以进行查询设备列表，切换设备等操作。

```
getDeviceManager(): TXDeviceManager;
```

返回

`TXDeviceManager` 实例对象。具体参见 TXDeviceManager 。

setVideoMute

设置是否禁用视频流。

```
setVideoMute(mute: boolean): void;
```

参数

`mute`：true – 禁用，false – 启用。

说明

- 只有当前有视频流时，设置才会生效。
- 禁用之后每一帧都会用黑色像素填充，实际上仍在采集视频流。

setAudioMute

设置是否禁用音频流。

```
setAudioMute(mute: boolean): void;
```

参数

`mute`：true – 禁用，false – 启用。

说明

- 只有当前有音频流时，设置才会生效。
- 禁用之后每一帧都是没有声音的，实际上仍在采集音频流。

setObserver

设置推流器回调。通过设置回调，可以监听推流器的一些回调事件，包括推流器状态、统计数据、警告和错误信息等。

```
setObserver(observer: TXLivePusherObserver): void;
```

参数

`observer`：推流器的回调目标对象。具体请参见 TXLivePusherObserver 。

destroy

离开页面或者退出时，清理 SDK 实例，避免可能会产生的内存泄露，调用前先执行 stop 方法。

```
destroy(): void;
```

TXLivePusherObserver

最近更新时间：2023-07-12 15:40:22

腾讯云直播推流的回调通知，回调包括推流器状态，统计信息，警告以及错误信息。

onError

直播推流器错误通知，推流器出现错误时，会回调该通知。

```
onError(code: number, msg: string, extraInfo: Object): void;
```

参数

- code：错误码，请参见 推流错误码 。
- msg：错误信息。
- extraInfo：扩展信息，目前未用到。

onWarning

直播推流器警告通知。

```
onWarning(code: number, msg: string, extraInfo: Object): void;
```

参数

- code：错误码，请参见 推流错误码 。
- msg：错误信息。
- extraInfo：扩展信息，目前未用到。

说明

打开摄像头、麦克风、屏幕录制失败时返回的错误信息，可参见 getUserMedia 异常 和 getDisplayMedia 异常 。

onCaptureFirstAudioFrame

首帧音频采集完成的回调通知。

```
onCaptureFirstAudioFrame(): void;
```

onCaptureFirstVideoFrame

首帧视频采集完成的回调通知。

```
onCaptureFirstVideoFrame(): void;
```

onPushStatusUpdate

推流器连接状态回调通知。

```
onPushStatusUpdate(status: number, msg: string, extraInfo: Object): void;
```

参数

- status：连接状态码，请参见 推流状态码 。
- msg：连接状态信息。
- extraInfo：扩展信息，目前未用到。

onStatisticsUpdate

直播推流器统计数据回调，主要是 WebRTC 相关的统计数据，不同浏览器返回的数据可能不一致。

```
onStatisticsUpdate(statistics: Object): void;
```

参数

statistics：推流统计数据，请参见 推流统计数据 。

推流错误码

onError 和 onWarning 中用到的错误码如下：

枚举值	数值	描述
TXLIVE_ERROR_WEBRTC_FAILED	-1	WebRTC 接口调用失败
TXLIVE_ERROR_REQUEST_FAILED	-2	请求服务器推流接口返回报错
TXLIVE_WARNING_CAMERA_START_FAILED	-10001	打开摄像头失败
TXLIVE_WARNING_MICROPHONE_START_FAILED	-10002	打开麦克风失败
TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	-100	打开屏幕录制失败

	3	
TXLIVE_WARNING_MEDIA_FILE_START_FAILED	-1004	打开本地媒体文件失败
TXLIVE_WARNING_CAMERA_INTERRUPTED	-1005	摄像头被中断（设备被拔出或者权限被用户取消）
TXLIVE_WARNING_MICROPHONE_INTERRUPTED	-1006	麦克风被中断（设备被拔出或者权限被用户取消）
TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	-1007	屏幕录制被中断（Chrome 浏览器单击自带的停止共享按钮）

推流状态码

onPushStatusUpdate 用到的状态码如下：

枚举值	数值	描述
TXLIVE_PUSH_STATUS_DISCONNECTED	0	与服务器断开连接
TXLIVE_PUSH_STATUS_CONNECTING	1	正在连接服务器
TXLIVE_PUSH_STATUS_CONNECT_SUCCESS	2	连接服务器成功
TXLIVE_PUSH_STATUS_RECONNECTING	3	重连服务器中

推流统计数据

通过 onStatisticsUpdate 回调接口可以监听直播推流过程中 WebRTC 相关的统计数据。SDK 会以一秒一次的频率统计数据，返回的对象基本结构如下：

字段	说明
timestamp	数据采样的时间，自1970年1月1日（UTC）起经过的毫秒数
video	视频流相关的数据
audio	音频流相关的数据

video

字段	说明
bitrate	视频码率，单位：bit/s
framesPerSecond	视频帧率
frameWidth	视频宽度，Firefox下不存在
frameHeight	视频高度，Firefox下不存在
framesEncoded	编码帧数
framesSent	发送帧数，Firefox下不存在
packetsSent	发送包数
nackCount	NACK（Negative ACKnowledgement）数
firCount	FIR（Full Intra Request），关键帧重传请求数
pliCount	PLI（Picture Loss Indication），视频帧丢失重传数
frameEncodeAvgTime	平均编码时间，单位：ms，Safari / Firefox 下不存在
packetSendDelay	数据包发送之前本地缓存的平均时间，单位：ms，Firefox 下不存在

audio

字段	说明
bitrate	音频码率，单位：bit/s
packetsSent	发送包数

TXDeviceManager

最近更新时间：2023-09-19 18:43:54

腾讯云设备管理接口，主要用于管理摄像头和麦克风设备。

getDevicesList

获取设备列表。

```
getDevicesList(type?: string): Promise<TXMediaDeviceInfo[]>;
```

参数

`type`：设备类型，可选参数，不传返回所有设备列表。传 video 返回摄像头设备列表，传 audio 返回麦克风设备列表。

返回

返回 Promise 对象，其中设备信息 `TXMediaDeviceInfo` 结构如下：

字段	类型	描述
type	string, 'video' |'audio'	设备类型，video - 摄像头，audio - 麦克风
deviceId	string	设备 Id
deviceName	string	设备名称

说明

- 该接口不支持在 HTTP 协议下使用，请使用 HTTPS 协议部署您的网站。
- 浏览器出于安全的考虑，在用户未授权摄像头或麦克风访问权限前，deviceId 及 deviceName 字段可能都是空的。因此建议在用户授权访问后，再调用该接口获取设备详情。

setCurrentDevice

设置要使用的设备。

```
setCurrentDevice(type: string, deviceId: string): void;
```

参数

- `type`：设备类型，video - 摄像头设备，audio - 麦克风设备。
- `deviceId` 从 getDevicesList 中得到的设备 ID。

getCurrentDevice

获取当前的设备信息。

```
getCurrentDevice(type: string): Promise<TXMediaDeviceInfo | null>;
```

参数

`type`：设备类型，video – 摄像头设备，audio – 麦克风设备。

返回

返回 Promise 对象，其中设备信息 `TXMediaDeviceInfo` 结构请参见 getDevicesList 返回说明。

switchDevice

切换当前正在使用的设备。

```
switchDevice(type: string, deviceId: string): void;
```

参数

- `type`：设备类型，video – 摄像头设备，audio – 麦克风设备。
- `deviceId` 从 getDevicesList 中得到的设备 ID。

说明

- 该方法仅适用于从摄像头和麦克风采集音视频时调用，屏幕录制和本地媒体文件采集流时不支持调用该接口。
- 如果还没开始推流，则只更新本地流；如果已经开始推流，同步更新推到服务器的音视频流。

switchCamera

切换摄像头设备。等同于 `switchDevice('video', deviceId)` 。

```
switchCamera(deviceId: string): void;
```

参数

`deviceId`：从 `getDevicesList('video')` 中得到的设备 ID。

switchMicrophone

切换麦克风设备。等同于 `switchDevice('audio', deviceId)` 。

```
switchMicrophone(deviceId: string): void;
```

参数

`deviceId`：从 `getDevicesList('audio')` 中得到的设备 ID。

TCPlayer

最近更新时间：2023-10-11 16:20:22

本文档是介绍适用于直播和点播播放的 Web 播放器（ TCPlayer ）的相关参数以及 API。本文档适合有一定 Javascript 语言基础的开发人员阅读。

初始化参数

播放器初始化需要传入两个参数，第一个为播放器容器 ID，第二个为功能参数对象。

```
var player = TCPlayer('player-container-id', options);
```

options 参数列表

options 对象可配置的参数：

名称	类型	默认值	说明
appID	String	无	通过 fileID 播放点播媒体文件时必选，为对应腾讯云账号的 appID
fileID	String	无	通过 fileID 播放点播媒体文件时必选，为点播媒体文件的 ID
sources	Array	无	播放器播放地址，格式：`[{ src: '//path/to/video.mp4', type: 'video/mp4' }]`
width	String/Number	无	播放器区域宽度，单位像素，按需设置，可通过 CSS 控制播放器尺寸。
height	String/Number	无	播放器区域高度，单位像素，按需设置，可通过 CSS 控制播放器尺寸。
controls	Boolean	true	是否显示播放器的控制栏。
poster	String	无	设置封面图片完整地址（如果上传的视频已生成封面图，优先使用生成的封面图，详细请参见 云点播 - 管理视频 ）。
autoplay	Boolean	false	是否自动播放。
playbackRates	Array	[0.5，1，1.25，1.5，2]	设置变速播放倍率选项，仅 HTML5 播放模式有效。

loop	Boolean	false	是否循环播放。
muted	Boolean	false	是否静音播放。
preload	String	auto	是否需要预加载，有3个属性"auto"，"meta"和"none"，移动端由于系统限制，设置 auto 无效。
swf	String	无	Flash 播放器 swf 文件的 URL。
posterImage	Boolean	true	是否显示封面。
bigPlayButton	Boolean	true	是否显示居中的播放按钮（浏览器劫持嵌入的播放按钮无法去除）。
language	String	"zh-CN"	设置语言，可选值为 "zh-CN"/"en"
languages	Object	无	设置多语言词典。
controlBar	Object	无	设置控制栏属性的参数组合，后面有详细介绍。
reportable	Boolean	true	设置是否开启数据上报。
plugins	Object	无	设置插件功能属性的参数组合，后面有详细介绍。
hlsConfig	Object	无	hls.js 的启动配置，详细内容请参见官方文档 hls.js。
webrtcConfig	Object	无	webrtc 的启动配置，后面有详细介绍。

注意：
controls、playbackRates、loop、preload、posterImage 这些参数在浏览器劫持播放的状态下将无效（什么是劫持播放？）。

controlBar 参数列表

controlBar 参数可以配置播放器控制栏的功能，支持的属性有：

名称	类型	默认值	说明
playToggle	Boole	true	是否显示播放、暂停切换按钮。

	an		
progressControl	Boolean	true	是否显示播放进度条。
volumePanel	Boolean	true	是否显示音量控制。
currentTimeDisplay	Boolean	true	是否显示视频当前时间。
durationDisplay	Boolean	true	是否显示视频时长。
timeDivider	Boolean	true	是否显示时间分割符。
playbackRateMenuButton	Boolean	true	是否显示播放速率选择按钮。
fullscreenToggle	Boolean	true	是否显示全屏按钮。
QualitySwitcherMenuButton	Boolean	true	是否显示清晰度切换菜单。

注意：
controlBar 参数在浏览器劫持播放的状态下将无效（什么是劫持播放？）。

plugins 插件参数列表

plugins 参数可以配置播放器插件的功能，支持的属性有：

名称	类型	默认值	说明
ContinuePlay	Object	无	控制续播功能，支持的属性如下： • auto：Boolean 是否在播放时自动续播 • text：String 提示文案 • btnText：String 按钮文案
VttThumbnail	Object	无	控制缩略图显示，支持的属性如下： • vttUrl：String vtt，文件绝对地址，必传 • basePath：String 图片路径，非必须，不传时使用 vttUrl 的 path

			• imgUrl：String 图片绝对地址，非必须
ProgressMarker	Boolean	无	控制进度条显示
DynamicWatermark	Object	无	控制动态水印显示，支持文字和图片，支持的属性为： • type：String 水印类型为文字或图片，取值为 text | image，默认 text，非必传。 • content：String 文字水印内容，必传。 • speed：Number 水印移动速度，取值范围 0-1, 默认值 0.2，非必传。 • opacity：Number 文字水印透明度，取值范围 0-1，非必传。 • fontSize：String 文字字体大小，默认12px，非必传。 • color： String 文字颜色，非必传。 • left：String, 文字位置，支持单位为百分比和 px，该字段设置时，speed 字段无效，非必传。 • top、right、bottom：说明同 left。 • width：String 图片水印宽度，非必传。 • height：String 图片水印高度，非必传 。

ContextMenu	Object	无	可选值如下：
			• mirror：Boolean 控制是否支持镜像显示
			• statistic：Boolean 控制是否支持显示数据面板
			• levelSwitch：Object 控制切换清晰度时的文案提示
			  ○ open：Boolean 是否开启提示
			  ○ switchingText：String，开始切换清晰度时的提示文案
			  ○ switchedText：String，切换成功时的提示文案
			  ○ switchErrorText：String，切换失败时的提示文案

webrtcConfig 参数列表

webrtcConfig 参数来控制播放 WebRTC 过程中的行为表现，支持的属性有：

名称	类型	默认值	说明
connectRetryCount	Number	3	SDK 与服务器重连次数
connectRetryDelay	Number	1	SDK 与服务器重连延时
receiveVideo	Boolean	true	是否拉取视频流

receiveAudio	Boolean	true	是否拉取音频流
showLog	Boolean	false	是否在控制台打印日志

对象方法

初始化播放器返回对象的方法列表：

名称	参数及类型	返回值及类型	说明
src()	(String)	无	设置播放地址。
ready(function)	(Function)	无	设置播放器初始化完成后的回调。
play()	无	无	播放以及恢复播放。
pause()	无	无	暂停播放。
currentTime(seconds)	(Number)	(Number)	获取当前播放时间点，或者设置播放时间点，该时间点不能超过视频时长。
duration()	无	(Number)	获取视频时长。
volume(percent)	(Number) [0，1][可选]	(Number)/设置时无返回	获取或设置播放器音量。
poster(src)	(String)	(String)/设置时无返回	获取或设置播放器封面。
requestFullscreen()	无	无	进入全屏模式。
exitFullscreen()	无	无	退出全屏模式。
isFullscreen()	无	Boolean	返回是否进入了全屏模式。
on(type，listener)	(String, Function)	无	监听事件。
one(type，listener)	(String, Function)	无	监听事件，事件处理函数最多只执行1次。
off(type，listener)	(String, Function)	无	解绑事件监听。

buffered()	无	TimeRanges	返回视频缓冲区间。
bufferedPercent()	无	值范围[0，1]	返回缓冲长度占视频时长的百分比。
width()	(Number) [可选]	(Number)/设置时无返回	获取或设置播放器区域宽度，如果通过 CSS 设置播放器尺寸，该方法将无效。
height()	(Number) [可选]	(Number)/设置时无返回	获取或设置播放器区域高度，如果通过 CSS 设置播放器尺寸，该方法将无效。
videoWidth()	无	(Number)	获取视频分辨率的宽度。
videoHeight()	无	(Number)	获取视频分辨率的高度。
dispose()	无	无	销毁播放器。

> ⚠ **注意：**
> 对象方法不能同步调用，需要在相应的事件（如 loadedmetadata）触发后才可以调用，除了 ready、on、one 以及 off。

事件

播放器可以通过初始化返回的对象进行事件监听，示例：

```
var player = TCPlayer('player-container-id', options);
// player.on(type, function);
player.on('error', function(error) {
  // 做一些处理
});
```

其中 type 为事件类型，支持的事件有：

名称	介绍
play	已经开始播放，调用 play() 方法或者设置了 autoplay 为 true 且生效时触发，这时 paused 属性为 false。
playing	因缓冲而暂停或停止后恢复播放时触发，paused 属性为 false 。通常用这个事件来标记视频真正播放，play 事件只是开始播放，画面并没有开始渲染。
loadstart	开始加载数据时触发。
durationc	视频的时长数据发生变化时触发。

hange	
loadedmetadata	已加载视频的 metadata。
loadeddata	当前帧的数据已加载，但没有足够的数据来播放视频的下一帧时，触发该事件。
progress	在获取到媒体数据时触发。
canplay	当播放器能够开始播放视频时触发。
canplaythrough	当播放器预计能够在不停下来进行缓冲的情况下持续播放指定的视频时触发。
error	视频播放出现错误时触发。
pause	暂停时触发。
ratechange	播放速率变更时触发。
seeked	搜寻指定播放位置结束时触发。
seeking	搜寻指定播放位置开始时触发。
timeupdate	当前播放位置有变更，可以理解为 currentTime 有变更。
volumechange	设置音量或者 muted 属性值变更时触发。
waiting	播放停止，下一帧内容不可用时触发。
ended	视频播放已结束时触发。此时 currentTime 值等于媒体资源最大值。
resolutionswitching	清晰度切换进行中。
resolutionswitched	清晰度切换完毕。
fullscreenchange	全屏状态切换时触发。
webrtcevent	播放 webrtc 时的事件集合。
webrtcsta	播放 webrtc 时的统计数据。

ts	

WebrtcEvent 列表

播放器可以通过 webrtcevent 获取播放 webrtc 过程中的所有事件，示例：

```
var player = TCPlayer('player-container-id', options);
player.on('webrtcevent', function(event) {
  // 从回调参数 event 中获取事件状态码及相关数据
});
```

webrtcevent 状态码如下：

状态码	回调参数	介绍
1001	无	开始拉流
1002	无	已经连接服务器
1003	无	视频播放开始
1004	无	停止拉流，结束视频播放
1005	无	连接服务器失败，已启动自动重连恢复
1006	无	获取流数据为空
1007	localSdp	开始请求信令服务器
1008	remoteSdp	请求信令服务器成功
1009	无	拉流卡顿等待缓冲中
1010	无	拉流卡顿结束恢复播放

错误码

当播放器触发 error 事件时，监听函数会返回错误码，其中3位数以上的错误码为媒体数据接口错误码。错误码列表：

名称	描述
-1	播放器没有检测到可用的视频地址。
-2	获取视频数据超时。
1	视频数据加载过程中被中断。

	可能原因： • 网络中断。 • 浏览器异常中断。解决方案： • 查看浏览器控制台网络请求信息，确认网络请求是否正常。 • 重新进行播放流程。

2	由于网络问题造成加载视频失败。
	• 可能原因：网络中断。
	• 解决方案：
	  ○ 查看浏览器控制台网络请求信息，确认网络请求是否正常。
	  ○ 重新进行播放流程。

3	视频解码时发生错误。
	• 可能原因：视频数据异常，解码器解码失败。
	• 解决方案：
	  ○ 尝试重新转码再进行播放，排除由于转码流程引入的问题。
	  ○ 确认原始视频是否正常。
	  ○ 请联系技术客服并提供播放参数进行定位排查。

4	视频因格式不支持或者服务器或网络的问题无法加载。
	• 可能原因：
	  ○ 获取不到视频数据，CDN 资源不存在或者没有返回视频数据。
	  ○ 当前播放环境不支持播放该视频格式。
	• 解决方案：
	  ○ 查看浏览器控制台网络请求信息，确认视频数据请求是否正常。
	  ○ 确认是否按照使用文档加载了对应视频格式的播放脚本。
	  ○ 确认当前浏览器和页面环境是否支持将要播放的视频格式。
	  ○ 请联系技术客服并提供播放参数进行定位排查。

5	视频解密时发生错误。
	• 可能原因：
	  ○ 解密用的密钥不正确。
	  ○ 请求密钥接口返回异常。
	  ○ 当前播放环境不支持视频解密功能。
	• 解决方案：
	  ○ 确认密钥是否正确，以及密钥接口是否返回正常。
	  ○ 请联系技术客服并提供播放参数进行定位排查。

10	点播媒体数据接口请求超时。在获取媒体数据时，播放器重试3次后仍没有任何响应，会抛出该错误。 • 可能原因：

	○ 当前网络环境无法连接到媒体数据接口，或者媒体数据接口被劫持。 ○ 媒体数据接口异常。 • 解决方案： ○ 尝试打开我们提供的 Demo 页面看是否可以正常播放。 ○ 请联系技术客服并提供播放参数进行定位排查。
11	点播媒体数据接口没有返回数据。在获取媒体数据时，播放器重试3次后仍没有数据返回，会抛出该错误。 • 可能原因： ○ 当前网络环境无法连接到媒体数据接口，或者媒体数据接口被劫持。 ○ 媒体数据接口异常。 • 解决方案： ○ 尝试打开我们提供的 Demo 页面看是否可以正常播放。 ○ 请联系技术客服并提供播放参数进行定位排查。
12	点播媒体数据接口返回异常数据。在获取媒体数据时，播放器重试3次后仍返回无法解析的数据，会抛出该错误。 • 可能原因： ○ 当前网络环境无法连接到媒体数据接口，或者媒体数据接口被劫持。 ○ 播放参数有误，媒体数据接口无法处理。 ○ 媒体数据接口异常。 • 解决方案： ○ 尝试打开我们提供的 Demo 页面看是否可以正常播放。 ○ 请联系技术客服并提供播放参数进行定位排查。
13	播放器没有检测到可以在当前播放器播放的视频数据，请对该视频进行转码操作。
14	HTML5 + hls.js 模式下播放 hls 出现网络异常，异常详情可在 event.source 中查看，详细介绍请看 hls.js 的官方文档 Network Errors 。
15	HTML5 + hls.js 模式下播放 hls 出现多媒体异常，异常详情可在 event.source 中查看，详细介绍请看 hls.js 的官方文档 Media Errors 。
16	HTML5 + hls.js 模式下播放 hls 出现多路复用异常，异常详情可在 event.source 中查看，详细介绍请看 hls.js 的官方文档 Mux Errors 。
17	HTML5 + hls.js 模式下播放 hls 出现其他异常，异常详情可在 event.source 中查看，详细介绍请看 hls.js 的官方文档 Other Errors 。
10008	媒体数据服务没有找到对应播放参数的媒体数据，请确认请求参数 appID fileID 是否正确，以及对应的媒体数据是否已经被删除。

Flutter
API 概览

最近更新时间：2024-05-08 17:33:43

V2TXLivePlayer

腾讯云直播播放器。请参见 V2TXLivePlayer 。

功能

主要负责从指定的直播流地址拉取音视频数据，并进行解码和本地渲染播放。

介绍

播放器包含如下能力：

- 支持 RTMP、HTTP-FLV、TRTC、WebRTC 协议。
- 屏幕截图，可以截取当前直播流的视频画面。
- 延时调节，可以设置播放器缓存自动调整的最小和最大时间。
- 自定义的视频数据处理，您可以根据项目需要处理直播流中的视频数据后，再进行渲染以及播放。

SDK 基础函数

API	描述
create	创建实例
destroy	销毁实例
addListener	添加播放器回调
removeListener	移除播放器回调

播放基础接口

API	描述
setRenderViewID	设置渲染视图的ID
startLivePlay	开始播放音视频流
stopPlay	停止播放音视频流
isPlaying	播放器是否正在播放中

视频相关接口

API	描述
setRenderRotation	设置本地渲染画面旋转角度
setRenderFillMode	设置画面的填充模式
pauseVideo	暂停播放器的视频流
resumeVideo	恢复播放器的视频流
snapshot	截取播放过程中的视频画面
enableObserveVideoFrame	开启/关闭对视频帧的监听回调

音频相关接口

API	描述
pauseAudio	暂停播放器的音频流
resumeAudio	恢复播放器的音频流
setPlayoutVolume	设置播放器音量
enableVolumeEvaluation	启用播放音量大小提示

更多实用接口

API	描述
setCacheParams	设置播放器缓存自动调整的最小和最大时间（单位：秒）
showDebugView	显示仪表盘
enableReceiveSeiMessage	开启接收 SEI 消息
setProperty	调用 V2TXLivePlayer 的高级 API 接口

V2TXLivePlayerObserver

功能

腾讯云直播的播放器回调通知。

介绍

可以接收 V2TXLivePlayer 播放器的一些回调通知，包括播放器状态、播放音量回调、音视频首帧回调、统计数据、警告和错误信息等。

SDK 基础回调

API	描述
onError	错误回调，表示 SDK 不可恢复的错误，一定要监听并分情况给用户适当的界面提示
onWarning	警告回调，用于告知您一些非严重性问题，例如出现卡顿或者可恢复的解码失败
onConnected	已经成功连接到服务器

视频相关回调

API	描述
onVideoPlaying	视频播放事件
onVideoLoading	视频加载事件
onSnapshotComplete	截图回调
onVideoResolutionChanged	直播播放器分辨率变化通知
onRenderVideoFrame	自定义视频渲染回调

音频相关回调

API	描述
onAudioPlaying	音频播放事件
onAudioLoading	音频加载事件
onPlayoutVolumeUpdate	播放器音量大小

统计回调

API	描述
onStatisticsUpdate	直播播放器统计数据回调

SEI 回调

API	描述
onReceiveSeiMessage	收到 SEI 消息的回调

V2TXLivePusher

腾讯云直播推流器。请参见 V2TXLivePusher 。

功能

主要负责将本地的音频和视频画面进行编码，并推送到指定的推流地址，支持任意的推流服务端。

介绍

推流器包含如下能力：

- 自定义的视频采集，让您可以根据项目需要定制自己的音视频数据源。
- 美颜、滤镜、贴纸，包含多套美颜磨皮算法（自然&光滑）和多款色彩空间滤镜（支持自定义滤镜）。
- Qos 流量控制技术，具备上行网络自适应能力，可以根据主播端网络的具体情况实时调节音视频数据量。
- 脸形调整、动效挂件，支持基于优图 AI 人脸识别技术的大眼、瘦脸、隆鼻等脸形微调以及动效挂件效果，只需要购买优图 License 就可以轻松实现丰富的直播效果。

SDK 基础函数

API	描述
create	创建实例
destroy	销毁实例
addListener	添加推流器回调
removeListener	移除推流器回调

推流基础接口

API	描述
setRenderViewID	设置本地摄像头预览视图的ID
startPush	开始音视频数据推流
stopPush	停止推送音视频数据
isPushing	当前推流器是否正在推流中

视频相关接口

API	描述
enableCustomVideoCapture	开启/关闭自定义视频采集
enableCustomVideoProcess	开启/关闭自定义视频处理
pauseVideo	暂停推流器的视频流
resumeVideo	恢复推流器的视频流
sendCustomVideoFrame	在自定义视频采集模式下，将采集的视频数据发送到SDK
setVideoQuality	设置推流视频编码参数
setRenderRotation	设置本地摄像头预览画面的旋转角度
setRenderMirror	设置摄像头镜像类型
setEncoderMirror	设置视频编码镜像
setWatermark	设置推流器水印。默认情况下，水印不开启
snapshot	截取推流过程中的视频画面
startCamera	打开本地摄像头
stopCamera	关闭本地摄像头
startVirtualCamera	开启图片推流
stopVirtualCamera	关闭图片推流
startScreenCapture	开启屏幕采集

stopScreenCapture	关闭屏幕采集

美颜相关接口

API	描述
getBeautyManager	获取美颜管理对象

音频相关接口

API	描述
startMicrophone	打开麦克风
stopMicrophone	关闭麦克风
setAudioQuality	设置推流音频质量
enableVolumeEvaluation	启用采集音量大小提示
enableCustomAudioCapture	开启/关闭自定义音频采集
sendCustomAudioFrame	在自定义音频采集模式下，将采集的音频数据发送到SDK
pauseAudio	暂停推流器的音频流
resumeAudio	恢复推流器的音频流

音效相关接口

API	描述
getAudioEffectManager	获取音效管理对象

设备管理相关接口

API	描述
getDeviceManager	获取设备管理对象

更多实用接口

API	描述
setProperty	调用 V2TXLivePusher 的高级 API 接口
setMixTranscodingConfig	设置云端的混流转码参数
showDebugView	显示仪表盘
sendSeiMessage	发送 SEI 消息

V2TXLivePusherObserver

功能

腾讯云直播的推流回调通知。

介绍

可以接收 V2TXLivePusher 推流器的一些推流通知，包括推流器连接状态、音视频首帧回调、统计数据、警告和错误信息等。

SDK 基础回调

API	描述
onError	错误回调，表示 SDK 不可恢复的错误，一定要监听并分情况给用户适当的界面提示
onWarning	警告回调，用于告知您一些非严重性问题，例如出现卡顿或者可恢复的解码失败
onPushStatusUpdate	推流器连接状态回调通知

视频相关回调

API	描述
onCaptureFirstVideoFrame	首帧视频采集完成的回调通知
onSnapshotComplete	截图回调

onProcessVideoFrame	自定义视频处理回调
onGLContextDestroyed	SDK 内部的 OpenGL 环境的销毁通知
onScreenCaptureStarted	当屏幕分享开始时，SDK 会通过此回调通知
onScreenCaptureStopped	当屏幕分享停止时，SDK 会通过此回调通知

音频相关回调

API	描述
onCaptureFirstAudioFrame	首帧音频采集完成的回调通知
onMicrophoneVolumeUpdate	麦克风采集音量值回调
onMusicObserverStart	音频开始播放
onMusicObserverPlayProgress	音频播放中
onMusicObserverComplete	音频播放结束

统计回调

API	描述
onStatisticsUpdate	直播推流器统计数据回调

混流回调

API	描述
onSetMixTranscodingConfig	设置云端的混流转码参数的回调

示例

最近更新时间：2023-09-19 18:43:55

针对开发者的接入反馈的高频问题，腾讯云提供有直观易懂的 API-Example 工程，方便开发者可以快速的了解相关 API 的使用，欢迎使用。

工程地址

所属平台	GitHub 地址
Flutter	Github

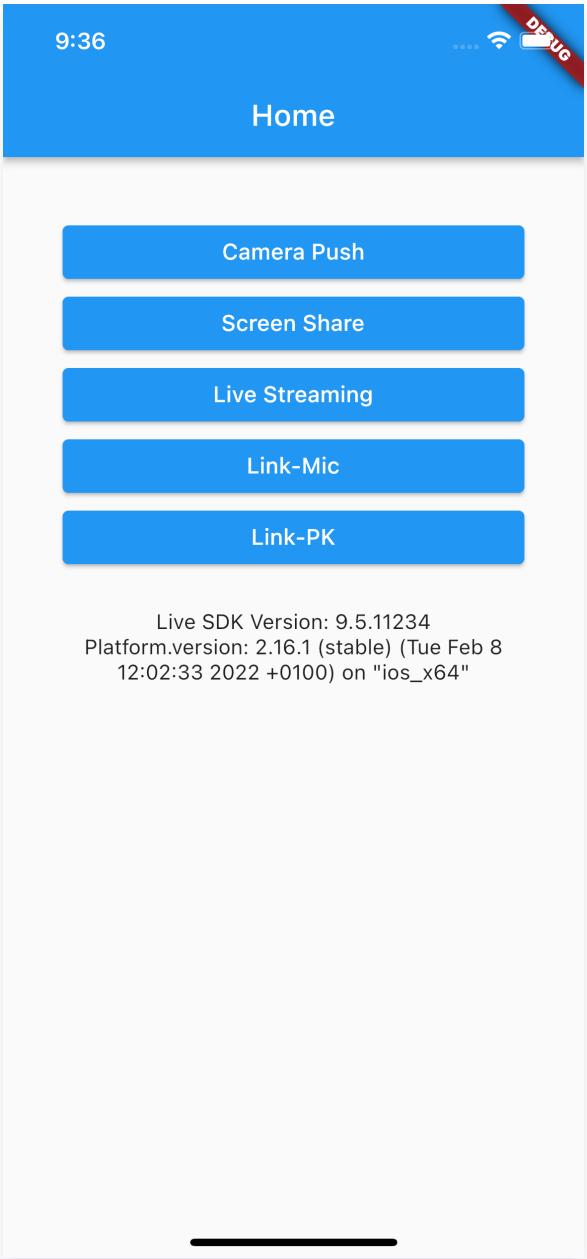
目录说明

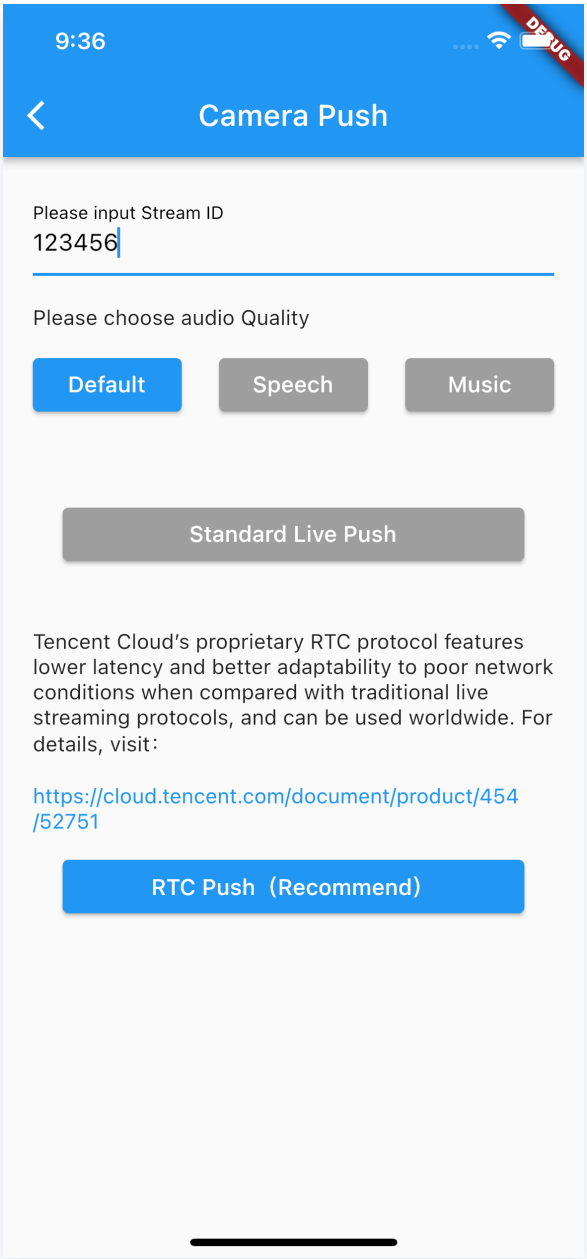
在这个示例项目中包含了以下基础功能场景：

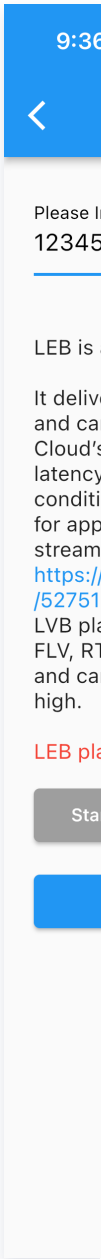
- 摄像头推流
- 录屏推流
- 直播拉流
- 连麦互动
- 连麦 PK

应用图示

示例主页	推流示例	拉流示例


9:36
DEBUG
Home
Camera Push
Screen Share
Live Streaming
Link-Mic
Link-PK
Live SDK Version: 9.5.11234
Platform.version: 2.16.1 (stable) (Tue Feb 8 12:02:33 2022 +0100) on "ios_x64"


9:36
DEBUG
Camera Push
Please input Stream ID
123456
Please choose audio Quality
Default
Speech
Music
Standard Live Push
Tencent Cloud's proprietary RTC protocol features lower latency and better adaptability to poor network conditions when compared with traditional live streaming protocols, and can be used worldwide. For details, visit:
https://cloud.tencent.com/document/product/454/52751
RTC Push（Recommend）



错误码表

最近更新时间：2023-08-18 09:49:42

onError 严重错误

SDK 发现了一些严重问题，推流无法继续了，例如 License 不合法，调用失败。

> **注意**
> 视频编码失败并不会直接影响推流，SDK 会做自行处理以保证后面的视频编码成功。

code	事件定义	含义说明
0	V2TXLIVE_OK	没有错误
-1	V2TXLIVE_ERROR_FAILED	暂未归类的通用错误
-2	V2TXLIVE_ERROR_INVALID_PARAMETER	调用 API 时，传入的参数不合法
-3	V2TXLIVE_ERROR_REFUSED	API 调用被拒绝
-4	V2TXLIVE_ERROR_NOT_SUPPORTED	当前 API 不支持调用
-5	V2TXLIVE_ERROR_INVALID_LICENSE	Licence 不合法，调用失败。注：在 `startLivePlay` 之前，需要通过 `V2TXLivePremier#setLicence` 设置 License 后方可成功播放，否则将播放失败（黑屏），全局仅设置一次即可。直播 License、短视频 License 和播放器 License 均可使用，若您暂未获取上述 License，可 快速免费申请测试版 License 以正常播放，正式版 License 需 购买
-6	V2TXLIVE_ERROR_REQUEST_TIMEOUT	请求服务器超时
-7	V2TXLIVE_ERROR_SERVER_PROCESS_FAILED	服务器拒绝请求
-8	V2TXLIVE_ERROR_DISCONNECTED	连接断开
-2	V2TXLIVE_ERROR_NO_AVAILABL	找不到可用的 HEVC 解码器，当抛出此错误码，

304	E_HEVC_DECODERS	说明当前设备不支持 H265 解码，请切换至 H264 数据流进行播放

onWarning 警告事件

大部分的警告事件会触发一些重试性的保护逻辑或者恢复逻辑，很大概率能够由 SDK 自行恢复。部分警告事件需要由开发者处理。警告的目的主要用于提示开发者或者最终用户。

code	事件定义	含义说明
11001	V2TXLIVE_WARNING_NETWORK_BUSY	网络状况不佳：上行带宽太小，上传数据受阻
2105	V2TXLIVE_WARNING_VIDEO_BLOCK	当前视频播放出现卡顿
-1301	V2TXLIVE_WARNING_CAMERA_START_FAILED	摄像头打开失败
-1316	V2TXLIVE_WARNING_CAMERA_OCCUPIED	摄像头正在被占用中，可尝试打开其他摄像头
-1314	V2TXLIVE_WARNING_CAMERA_NO_PERMISSION	摄像头设备未授权，通常在移动设备出现，可能是权限被用户拒绝了
-1302	V2TXLIVE_WARNING_MICROPHONE_START_FAILED	麦克风打开失败
-1319	V2TXLIVE_WARNING_MICROPHONE_OCCUPIED	麦克风正在被占用中，例如移动设备正在通话时，打开麦克风会失败
-1317	V2TXLIVE_WARNING_MICROPHONE_NO_PERMISSION	麦克风设备未授权，通常在移动设备出现，可能是权限被用户拒绝了
-1309	V2TXLIVE_WARNING_SCREEN_CAPTURE_NOT_SUPPORTED	当前系统不支持屏幕分享
-130	V2TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	开始录屏失败，如果在移动设备出现，可能是权限被用户拒绝了

8		
-7001	V2TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	录屏被系统中断