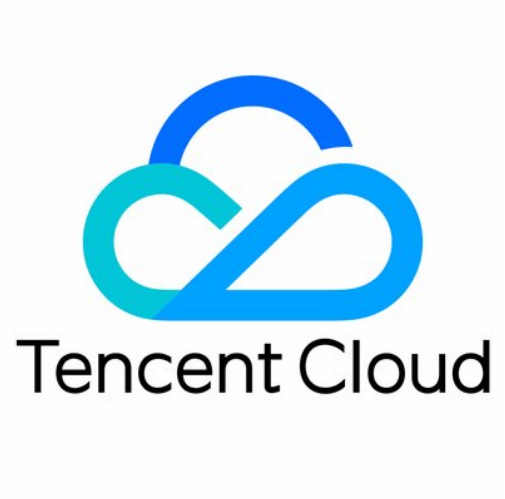
Mobile Live Video Broadcasting

Client API

Product Documentation



Contents

Client API
iOS&Mac
API Overview

Last updated：2024-06-06 15:47:57

API OVERVIEW

LivePusher Interface

FuncList	DESC
setObserver:	Sets the pusher callback
setRenderView:	Sets the local camera preview
setRenderMirror:	Sets the view mirror of the local camera
setEncoderMirror:	Sets the video encoder mirror
setRenderRotation:	Sets the rotation angle of the view
setRenderFillMode:	Sets the fill mode of the local video image
startCamera:	Enables the local camera
stopCamera	Disables the local camera
startMicrophone	Enables the local microphone
stopMicrophone	Disables the microphone
startVirtualCamera:	Enables the image streaming
stopVirtualCamera	Disables the image streaming
startScreenCapture:	Enables video capturing
stopScreenCapture	Disables video capture
pauseAudio	Mute local audio
resumeAudio	Resume the audio stream of the pusher

pauseVideo	Pause the video stream of the pusher
resumeVideo	Resume the video stream of the pusher
startPush:	Starts pushing the audio and video data
stopPush	Stops pushing the audio and video data
isPushing	Indicates whether the pusher is currently pushing streams
setAudioQuality:	Sets the audio quality for pushing
setVideoQuality:	Set the video encoding parameters for pushing
getBeautyManager	Obtains the beauty manager
getAudioEffectManager	Obtains the audio effect manager
getDeviceManager	Obtains the video device manager
snapshot	Captures the local view in the pushing process
setWatermark:x:y:scale:	Sets the pusher watermark image. By default, the watermark is disabled
enableVolumeEvaluation:	Enables volume update
enableCustomVideoProcess:pixelFormat:bufferType:	Enables or disables custom video processing
enableCustomVideoCapture:	Enables or disables custom video capture
enableCustomAudioCapture:	Turn on/off custom audio capture
sendCustomVideoFrame:	Sends the collected video data to the SDK in the custom video capture mode
sendCustomAudioFrame:	In the custom audio collection mode, send the collected audio data to the SDK
enableAudioProcessObserver:format:	Enables/Disables audio process callback
sendSeiMessage:data:	Use SEI channel to send custom message
showDebugView:	Indicates whether the debug view of the pusher video status information is displayed
setProperty:value:	Calls the advanced API of V2TXLivePusher

setMixTranscodingConfig:	Sets On-Cloud MixTranscoding parameters
startLocalRecording:	Start recording audio and video stream
stopLocalRecording	Stop recording audio and video stream
enableVoiceActivityDetection:	Enable voice activity detection

Live pusher Event Callback

FuncList	DESC
onError:message:extraInfo:	Live pusher error notification, which is called back when the pusher encounters an error
onWarning:message:extraInfo:	Live pusher warning notification
onCaptureFirstAudioFrame	Callback notification indicating that collection of the first audio frame is complete
onCaptureFirstVideoFrame	Callback notification indicating that collection of the first video frame is complete
onMicrophoneVolumeUpdate:	Microphone-collected volume callback
onPushStatusUpdate:message:extraInfo:	Callback notification of the pusher connection status
onStatisticsUpdate:	Live pusher statistics callback
onSnapshotComplete:	Screenshot callback
onProcessAudioFrame:	Audio data captured by the local mic, pre-processed by the audio module, effect-processed and BGM-mixed
onProcessVideoFrame:dstFrame:	Custom video processing callback
onGLContextDestroyed	Callback of destroying the OpenGL context in the SDK
onSetMixTranscodingConfig:message:	Callback of setting On-Cloud MixTranscoding parameters, which corresponds to the setMixTranscodingConfig API
onScreenCaptureStarted	The SDK returns this callback when you call startScreenCapture and other APIs to start screen sharing.
onScreenCaptureStopped:	The SDK returns this callback when you call stopScreenCapture to stop screen sharing

onLocalRecordBegin:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete:storagePath:	The SDK returns this callback when you call stopLocalRecording to start local recording.
onVoiceActivityDetectionUpdate:	After calling enableVoiceActivityDetection to turn on voice activity detection, you will receive this callback notification when the anchor starts or stops speaking.

V2TXLivePlayer Interface

FuncList	DESC
setObserver:	Sets the player callback
setRenderView:	Sets the rendering view of the player. This control is responsible for presenting the video content
setRenderRotation:	Sets the rotation angle of the player view
setRenderFillMode:	Sets the fill mode of the view
startLivePlay:	Starts playing the audio and video streams
stopPlay	Stops playing the audio and video streams
isPlaying	Indicates whether the player is playing the audio and video streams
pauseAudio	Pauses the audio stream of the player
resumeAudio	Resumes the audio stream of the player
pauseVideo	Pauses the video stream of the player
resumeVideo	Resumes the video stream of the player
setPlayoutVolume:	Sets the volume
setCacheParams:maxTime:	Set the minimum time and maximum time (unit:

	s) for auto adjustment of the player cache
switchStream:	Seamlessly switch live stream urls, supporting FLV and LEB protocols
getStreamList	Get Stream Info List
enableVolumeEvaluation:	Enables playback volume update
snapshot	Captures the video view in the playback process
enableObserveVideoFrame:pixelFormat:bufferType:	Turn on/off the monitoring callback of the video frame
enableObserveAudioFrame:	Turn on/off the monitoring callback of the audio frame
enableReceiveSeiMessage:payloadType:	Enables the receiving of SEI messages
enablePictureInPicture:	Enables Picture-in-Picture mode
showDebugView:	Indicates whether the debug view of the player video status information is displayed
setProperty:value:	Calls the advanced API of V2TXLivePlayer
startLocalRecording:	Start recording audio and video stream
stopLocalRecording	Stop recording audio and video stream

Live Player Event Callback

FuncList	DESC
onError:code:message:extraInfo:	live player error notification, which is called back when the player encounters an error
onWarning:code:message:extraInfo:	live player warning notification
onVideoResolutionChanged:width:height:	live player resolution change notification
onConnected:extraInfo:	live player has successfully connected to the server notification
onVideoPlaying:firstPlay:extraInfo:	Video playback event

onAudioPlaying:firstPlay:extraInfo:	Audio playback event
onVideoLoading:extraInfo:	Video loading event
onAudioLoading:extraInfo:	Audio loading event
onPlayoutVolumeUpdate:volume:	Player playback volume callback
onStatisticsUpdate:statistics:	Live player statistics callback
onSnapshotComplete:image:	Screenshot callback
onRenderVideoFrame:frame:	Custom video rendering callback
onPlayoutAudioFrame:frame:	Audio Data callback
onReceiveSeiMessage:payloadType:data:	Callback of receiving an SEI message. The sender calls `sendSeiMessage` in V2TXLivePusher to send an SEI
onStreamSwitched:url:code:	Resolution stream switch callback
onPictureInPictureStateUpdate:state:message:extraInfo:	Picture-in-Picture state change callback
onLocalRecordBegin:errCode:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording:durationMs:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete:errCode:storagePath:	The SDK returns this callback when you call stopLocalRecording to start local recording.

V2TXLive High-level interface

FuncList	DESC
getSDKVersionStr	Get the SDK version number
setObserver:	Set V2TXLivePremier callback interface
setLogConfig:	Set Log configuration information
setEnvironment:	Set up SDK access environment

setLicence:key:	Set SDK authorization license
setSocks5Proxy:port:username:password:config:	Set SDK socks5 proxy config
enableAudioCaptureObserver:format:	Enables/Disables audio capture callback
enableAudioPlayoutObserver:format:	Enables/Disables audio playout callback
enableVoiceEarMonitorObserver:	Enables/Disables in-ear monitoring callback
setUserId:	Set user id
callExperimentalAPI:	Call experimental APIs

V2TXLive Advanced callback interface

FuncList	DESC
onLog:log:	Custom Log output callback interface
onLicenceLoaded:Reason:	setLicence result callback interface
onCaptureAudioFrame:	Raw audio data captured locally
onPlayoutAudioFrame:	Data mixed from each channel before being submitted to the system for playback
onVoiceEarMonitorAudioFrame:	In-ear monitoring data

Voice effect APIs

FuncList	DESC
enableVoiceEarMonitor:	Enabling in-ear monitoring
setVoiceEarMonitorVolume:	Setting in-ear monitoring volume
setVoiceReverbType:	Setting voice reverb effects
setVoiceChangerType:	Setting voice changing effects
setVoiceVolume:	Setting speech volume
setVoicePitch:	Setting speech pitch

Background music APIs

FuncList	DESC
startPlayMusic:onStart:onProgress:onComplete:	Starting background music
stopPlayMusic:	Stopping background music
pausePlayMusic:	Pausing background music
resumePlayMusic:	Resuming background music
setAllMusicVolume:	Setting the local and remote playback volume of background music
setMusicPublishVolume:volume:	Setting the remote playback volume of a specific music track
setMusicPlayoutVolume:volume:	Setting the local playback volume of a specific music track
setMusicPitch:pitch:	Adjusting the pitch of background music
setMusicSpeedRate:speedRate:	Changing the speed of background music
getMusicCurrentPosInMS:	Getting the playback progress (ms) of background music
getMusicDurationInMS:	Getting the total length (ms) of background music
seekMusicToPosInMS:pts:	Setting the playback progress (ms) of background music
setMusicScratchSpeedRate:speedRate:	Adjust the speed change effect of the scratch disc
preloadMusic:onProgress:onError:	Preload background music
getMusicTrackCount:	Get the number of tracks of background music
setMusicTrack:track:	Specify the playback track of background music

beauty interface

FuncList	DESC
setBeautyStyle:	Sets the beauty (skin smoothing) filter algorithm.
setBeautyLevel:	Sets the strength of the beauty filter.
setWhitenessLevel:	Sets the strength of the brightening filter.
enableSharpnessEnhancement:	Enables clarity enhancement.
setRuddyLevel:	Sets the strength of the rosy skin filter.
setFilter:	Sets color filter.
setFilterStrength:	Sets the strength of color filter.
setGreenScreenFile:	Sets green screen video
setEyeScaleLevel:	Sets the strength of the eye enlarging filter.
setFaceSlimLevel:	Sets the strength of the face slimming filter.
setFaceVLevel:	Sets the strength of the chin slimming filter.
setChinLevel:	Sets the strength of the chin lengthening/shortening filter.
setFaceShortLevel:	Sets the strength of the face shortening filter.
setFaceNarrowLevel:	Sets the strength of the face narrowing filter.
setNoseSlimLevel:	Sets the strength of the nose slimming filter.
setEyeLightenLevel:	Sets the strength of the eye brightening filter.
setToothWhitenLevel:	Sets the strength of the teeth whitening filter.
setWrinkleRemoveLevel:	Sets the strength of the wrinkle removal filter.
setPounchRemoveLevel:	Sets the strength of the eye bag removal filter.
setSmileLinesRemoveLevel:	Sets the strength of the smile line removal filter.
setForeheadLevel:	Sets the strength of the hairline adjustment filter.
setEyeDistanceLevel:	Sets the strength of the eye distance adjustment filter.
setEyeAngleLevel:	Sets the strength of the eye corner adjustment filter.
setMouthShapeLevel:	Sets the strength of the mouth shape adjustment filter.

setNoseWingLevel:	Sets the strength of the nose wing narrowing filter.
setNosePositionLevel:	Sets the strength of the nose position adjustment filter.
setLipsThicknessLevel:	Sets the strength of the lip thickness adjustment filter.
setFaceBeautyLevel:	Sets the strength of the face shape adjustment filter.
setMotionTmpl:inDir:	Selects the AI animated effect pendant.
setMotionMute:	Sets whether to mute during animated effect playback.

Type definitions of audio/video devices

FuncList	DESC
onDeviceChanged:type:state:	The status of a local device changed (for desktop OS only)

Device APIs

FuncList	DESC
isFrontCamera	Querying whether the front camera is being used
switchCamera:	Switching to the front/rear camera (for mobile OS)
isCameraZoomSupported	Querying whether the current camera supports zooming (for mobile OS)
getCameraZoomMaxRatio	Getting the maximum zoom ratio of the camera (for mobile OS)
setCameraZoomRatio:	Setting the camera zoom ratio (for mobile OS)
isAutoFocusEnabled	Querying whether automatic face detection is supported (for mobile OS)
enableCameraAutoFocus:	Enabling auto focus (for mobile OS)
setCameraFocusPosition:	Adjusting the focus (for mobile OS)
isCameraTorchSupported	Querying whether flash is supported (for mobile OS)
enableCameraTorch:	Enabling/Disabling flash, i.e., the torch mode (for mobile

	OS)
setAudioRoute:	Setting the audio route (for mobile OS)
setExposureCompensation:	Set the exposure parameters of the camera, ranging from - 1 to 1
getDevicesList:	Getting the device list (for desktop OS)
setCurrentDevice:deviceId:	Setting the device to use (for desktop OS)
getCurrentDevice:	Getting the device currently in use (for desktop OS)
setCurrentDeviceVolume:deviceType:	Setting the volume of the current device (for desktop OS)
getCurrentDeviceVolume:	Getting the volume of the current device (for desktop OS)
setCurrentDeviceMute:deviceType:	Muting the current device (for desktop OS)
getCurrentDeviceMute:	Querying whether the current device is muted (for desktop OS)
enableFollowingDefaultAudioDevice:enable:	Set the audio device used by SDK to follow the system default device (for desktop OS)
startCameraDeviceTest:	Starting camera testing (for desktop OS)
stopCameraDeviceTest	Ending camera testing (for desktop OS)
startMicDeviceTest:	Starting mic testing (for desktop OS)
startMicDeviceTest:playback:	Starting mic testing (for desktop OS)
stopMicDeviceTest	Ending mic testing (for desktop OS)
startSpeakerDeviceTest:	Starting speaker testing (for desktop OS)
stopSpeakerDeviceTest	Ending speaker testing (for desktop OS)
setObserver:	set onDeviceChanged callback (for Mac)
setCameraCapturerParam:	Set camera acquisition preferences

Disused APIs

FuncList	DESC
setSystemVolumeType:	Setting the system volume type (for mobile OS)

V2TXLivePusher

Last updated：2024-06-06 15:47:57

Module: V2TXLivePusher @ TXLiteAVSDK

Function: Tencent Cloud live pusher

Function

Tencent Cloud Live Pusher

Introduce

It is mainly responsible for encoding the local audio and video images and pushing them to the specified streaming address, supporting any streaming server.
Flowmakers include the following capabilities:
Customized video capture, allowing you to customize your own audio and video data sources according to project needs.
Beautification, filters, stickers, including multiple sets of beautification and microdermabrasion algorithms (natural & smooth) and a variety of color space filters (support custom filters).
Qos flow control technology, with uplink network adaptive capability, can adjust the amount of audio and video data in real time according to the specific conditions of the host network.
Face shape adjustment, animation pendants, support face shape fine-tuning and animation pendant effects based on Youtu AI face recognition technology such as big eyes, thin face, nose augmentation, etc. You only need to purchase Youtu License to easily achieve rich live broadcast effects.

V2TXLivePusher

V2TXLivePusher

FuncList	DESC

setObserver:	Sets the pusher callback
setRenderView:	Sets the local camera preview
setRenderMirror:	Sets the view mirror of the local camera
setEncoderMirror:	Sets the video encoder mirror
setRenderRotation:	Sets the rotation angle of the view
setRenderFillMode:	Sets the fill mode of the local video image
startCamera:	Enables the local camera
stopCamera	Disables the local camera
startMicrophone	Enables the local microphone
stopMicrophone	Disables the microphone
startVirtualCamera:	Enables the image streaming
stopVirtualCamera	Disables the image streaming
startScreenCapture:	Enables video capturing
stopScreenCapture	Disables video capture
pauseAudio	Mute local audio
resumeAudio	Resume the audio stream of the pusher
pauseVideo	Pause the video stream of the pusher
resumeVideo	Resume the video stream of the pusher
startPush:	Starts pushing the audio and video data
stopPush	Stops pushing the audio and video data
isPushing	Indicates whether the pusher is currently pushing streams
setAudioQuality:	Sets the audio quality for pushing
setVideoQuality:	Set the video encoding parameters for pushing
getBeautyManager	Obtains the beauty manager
getAudioEffectManager	Obtains the audio effect manager

getDeviceManager	Obtains the video device manager
snapshot	Captures the local view in the pushing process
setWatermark:x:y:scale:	Sets the pusher watermark image. By default, the watermark is disabled
enableVolumeEvaluation:	Enables volume update
enableCustomVideoProcess:pixelFormat:bufferType:	Enables or disables custom video processing
enableCustomVideoCapture:	Enables or disables custom video capture
enableCustomAudioCapture:	Turn on/off custom audio capture
sendCustomVideoFrame:	Sends the collected video data to the SDK in the custom video capture mode
sendCustomAudioFrame:	In the custom audio collection mode, send the collected audio data to the SDK
enableAudioProcessObserver:format:	Enables/Disables audio process callback
sendSeiMessage:data:	Use SEI channel to send custom message
showDebugView:	Indicates whether the debug view of the pusher video status information is displayed
setProperty:value:	Calls the advanced API of V2TXLivePusher
setMixTranscodingConfig:	Sets On-Cloud MixTranscoding parameters
startLocalRecording:	Start recording audio and video stream
stopLocalRecording	Stop recording audio and video stream
enableVoiceActivityDetection:	Enable voice activity detection

setObserver:

setObserver:

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

Sets the pusher callback

By setting the callback, you can listen to some callback events of V2TXLivePusher, including the pusher status, volume callback, statistics, warnings, and error messages.

Param	DESC
observer	Callback target of the pusher. For more information, see V2TXLivePusherObserver.

setRenderView:

setRenderView:

- (V2TXLiveCode)setRenderView:	(TXView *)view

Sets the local camera preview

Images collected by the local camera will be eventually displayed on the view that is passed in after it is overlaid by multiple effects, such as beauty filters, facial feature adjustments, and filters.

Param	DESC
view	Local camera preview.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderMirror:

setRenderMirror:

- (V2TXLiveCode)setRenderMirror:	(V2TXLiveMirrorType)mirrorType

Sets the view mirror of the local camera

Local cameras are divided into the front camera and the rear camera. By default, images from the front camera are mirrored, and images from the rear camera are not mirrored. Here, you can modify the default mirror type of the front or rear camera.

Param	DESC
mirrorType	Mirror type of the camera V2TXLiveMirrorType.

	V2TXLiveMirrorTypeAuto `Default` : default mirror type. In this case, images from the front camera are mirrored, and images from the rear camera are not mirrored. V2TXLiveMirrorTypeEnable: both the front camera and rear camera are switched to mirror mode. V2TXLiveMirrorTypeDisable: both the front camera and rear camera are switched to non-mirror mode.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

setEncoderMirror:

setEncoderMirror:

- (V2TXLiveCode)setEncoderMirror:	(BOOL)mirror

Sets the video encoder mirror

Param	DESC
mirror	Specifies whether the mirrored images are viewed. NO `Default` : non-mirrored images are viewed on the player side. YES: mirrored images are viewed on the player side.

Note

The encoder mirror only influences video effects on the audience side.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

setRenderRotation:

setRenderRotation:

- (V2TXLiveCode)setRenderRotation:	(V2TXLiveRotation)rotation

Sets the rotation angle of the view

Param	DESC
rotation	Rotation angle of the view V2TXLiveRotation. V2TXLiveRotation0 `Default` : 0 degrees, which means the view is not rotated. V2TXLiveRotation90: rotate 90 degrees clockwise. V2TXLiveRotation180: rotate 180 degrees clockwise. V2TXLiveRotation270: rotate 270 degrees clockwise.

Note

Only the view is rotated, and images that are pushed are not affected.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

setRenderFillMode:

setRenderFillMode:

- (V2TXLiveCode)setRenderFillMode:	(V2TXLiveFillMode)mode

Sets the fill mode of the local video image

Param	DESC
mode	Fill mode of the view V2TXLiveFillMode. V2TXLiveFillModeFill: **Default**: fill the screen with the image without leaving any black edges. If the aspect ratio of the view is different from that of the screen, part of the view will be cropped. V2TXLiveFillModeFit make the view fit the screen without cropping. If the aspect ratio of the view is different from that of the screen, black edges will appear. V2TXLiveFillModeScaleFill fill the screen with the stretched image, thus the length and width may not change proportionally.

Return Desc:

Return code V2TXLiveCode

V2TXLIVE_OK: successful

startCamera:

startCamera:

- (V2TXLiveCode)startCamera:	(BOOL)frontCamera

Enables the local camera

Param	DESC
frontCamera	Specifies whether to switch to the front camera. YES `Default` : switch to the front camera. NO: switch to the rear camera.

Note

startVirtualCamera, startCamera, startScreenCapture, if use the same Pusher instance, only one can publish. To switch between different capture sources, first stop the previous capture source, and then start the next capture source to ensure that start and stop of the same capture source are called in pairs. eg: when the capture source is switched from Camera to VirtualCamera, the call sequence is startCamera -> stopCamera -> startVirtualCamera.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

stopCamera

stopCamera

Disables the local camera

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startMicrophone

startMicrophone

Enables the local microphone

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

stopMicrophone

stopMicrophone

Disables the microphone

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startVirtualCamera:

startVirtualCamera:

- (V2TXLiveCode)startVirtualCamera:	(TXImage *)image

Enables the image streaming

Param	DESC
image	image.

Note

startVirtualCamera, startCamera, startScreenCapture, if use the same Pusher instance, only one can publish. To switch between different capture sources, first stop the previous capture source, and then start the next capture source to ensure that start and stop of the same capture source are called in pairs. eg: when the capture source is switched from Camera to VirtualCamera, the call sequence is startCamera -> stopCamera -> startVirtualCamera.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

stopVirtualCamera

stopVirtualCamera

Disables the image streaming

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startScreenCapture:

startScreenCapture:

- (V2TXLiveCode)startScreenCapture:	(NSString *)appGroup

Enables video capturing

Param	DESC
appGroup	The Application Group Identifier shared by the main App and Broadcast can be specified as nil. It is worth noting that the function will be more reliable according to the document guidelines.

Note

startVirtualCamera, startCamera, startScreenCapture, if use the same Pusher instance, only one can publish. To switch between different capture sources, first stop the previous capture source, and then start the next capture source to ensure that start and stop of the same capture source are called in pairs. eg: when the capture source is switched from Camera to ScreenCapture, the call sequence is startCamera -> stopCamera -> startScreenCapture.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_ERROR_NOT_SUPPORTED: this feature is not supported.

stopScreenCapture

stopScreenCapture

Disables video capture

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

pauseAudio

pauseAudio

Mute local audio

After muting the local audio, the SDK will not continue to collect the microphone sound,
The difference from **stopMicrophone** is **pauseAudio** does not stop sending audio data, instead continue to send silent packets with a very low bit rate.
Due to video file formats such as MP4, the continuity of the audio is very demanding. Using **stopMicrophone** will cause the recorded MP4 to be difficult to play.
Therefore, in scenes that require high recording quality, it is recommended to choose **pauseAudio** to record MP4 files with better compatibility.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeAudio

resumeAudio

Resume the audio stream of the pusher

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

pauseVideo

pauseVideo

Pause the video stream of the pusher

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeVideo

resumeVideo

Resume the video stream of the pusher

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startPush:

startPush:

- (V2TXLiveCode)startPush:	(NSString *)url

Starts pushing the audio and video data

Param	DESC
url	Push URL, which can be any push server.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: operation succeeded. The pusher starts connecting to the target push URL.
V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The URL is invalid.
V2TXLIVE_ERROR_INVALID_LICENSE: operation failed. The license is invalid and authentication failed.
V2TXLIVE_ERROR_REFUSED: operation failed. Duplicate streamId, please ensure that no other player or pusher is using this streamId now.

stopPush

stopPush

Stops pushing the audio and video data

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

isPushing

isPushing

Indicates whether the pusher is currently pushing streams

Return Desc:

Indicates whether the pusher is pushing streams.

1: yes.

0: no.

setAudioQuality:

setAudioQuality:

- (V2TXLiveCode)setAudioQuality:	(V2TXLiveAudioQuality)quality

Sets the audio quality for pushing

Param	DESC
quality	Audio quality V2TXLiveAudioQuality. V2TXLiveAudioQualityDefault `Default` : universal. V2TXLiveAudioQualitySpeech: speech. V2TXLiveAudioQualityMusic: music.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_REFUSED: the audio quality cannot be adjusted in the pushing process.

setVideoQuality:

setVideoQuality:

- (V2TXLiveCode)setVideoQuality:	(V2TXLiveVideoEncoderParam *)param

Set the video encoding parameters for pushing

Param	DESC
param	video encoding parameters V2TXLiveVideoEncoderParam.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

getBeautyManager

getBeautyManager

Obtains the beauty manager

With the beauty manager, you can use the following features:
Set the following cosmetic effects: beauty style, whitening, ruddy, big eyes, slim face, V-shape face, chin, short face, small nose, bright eyes, white teeth, remove eye bags, remove wrinkles, remove laugh lines.
Adjust the hairline, eye spacing, eye corners, mouth shape, nose wings, nose position, lip thickness, and face shape.
Set animated effects such as face widgets (materials).
Add makeup effects.
Recognize gestures.

please see TXBeautyManager

getAudioEffectManager

getAudioEffectManager

Obtains the audio effect manager

With the audio effect manager, you can use the following features:
Adjust the volume of human voice collected by the microphone.
Set the reverb and voice changing effects.
Start the headphone monitor, and set the volume of the headphone monitor.
Add the BGM, and adjust the playback effect of BGM.

please see TXAudioEffectManager

getDeviceManager

getDeviceManager

Obtains the video device manager

With the device manager, you can use the following features:
Switch between the front and rear cameras.
Set the auto focus.
Adjust the camera magnification.
Turn the flash on or off.
Switch between the earphone and speaker.
Modify the volume type (media volume or conversation volume).

please see TXDeviceManager

snapshot

snapshot

Captures the local view in the pushing process

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.
V2TXLIVE_ERROR_REFUSED: pushing is stopped, and the snapshot operation cannot be called.

setWatermark:x:y:scale:

setWatermark:x:y:scale:

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

Sets the pusher watermark image. By default, the watermark is disabled

Param	DESC
image	Watermark image. If the value is nil, it is equivalent to disabling the watermark.
scale	Scaling ratio of the watermark. Valid range: 0 - 1.
x	Display position of the watermark. Valid range: 0 - 1.
y	Display position of the watermark. Valid range: 0 - 1.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

enableVolumeEvaluation:

enableVolumeEvaluation:

- (V2TXLiveCode)enableVolumeEvaluation:	(NSUInteger)intervalMs

Enables volume update

After this feature is enabled, you can obtain the volume evaluation through the onMicrophoneVolumeUpdate callback.

Param	DESC
intervalMs	Interval for triggering the volume callback. The unit is ms. The minimum interval is 100 ms. If the value is equal to or smaller than 0, the callback is disabled. We recommend that you set this parameter to 300 ms. `Default` : 0.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

enableCustomVideoProcess:pixelFormat:bufferType:

enableCustomVideoProcess:pixelFormat:bufferType:

- (V2TXLiveCode)enableCustomVideoProcess:	(BOOL)enable

pixelFormat:	(V2TXLivePixelFormat)pixelFormat
bufferType:	(V2TXLiveBufferType)bufferType

Enables or disables custom video processing

Param	DESC
bufferType	Data format of callbacks.
enable	`YES` : enable; `NO` : disable (**default**).
pixelFormat	Pixel format of callbacks.

Note

Supported format combinations:

V2TXLivePixelFormatTexture2D+V2TXLiveBufferTypeTexture

V2TXLivePixelFormatNV12+V2TXLiveBufferTypePixelBuffer

V2TXLivePixelFormatBGRA32+V2TXLiveBufferTypePixelBuffer

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_NOT_SUPPORTED` : unsupported format.

enableCustomVideoCapture:

enableCustomVideoCapture:

- (V2TXLiveCode)enableCustomVideoCapture:	(BOOL)enable

Enables or disables custom video capture

In the custom video capture mode, the SDK no longer captures images from cameras. Only the encoding and sending capabilities are retained.

Param	DESC
enable	`YES` : enable custom video capture; `NO` (**default**): disable custom video capture.

Note

This API takes effect only when it is called before startPush.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

enableCustomAudioCapture:

enableCustomAudioCapture:

- (V2TXLiveCode)enableCustomAudioCapture:	(BOOL)enable

Turn on/off custom audio capture

@brief Turn on/off custom audio capture.

In the custom audio capture mode, the SDK no longer collects sound from the microphone, and only retains the encoding and sending capabilities.

@note It needs to be called before startPush to take effect.

@param enable YES: Open custom capture; NO: Close custom capture. `Default value` : `NO` .

@return Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

sendCustomVideoFrame:

sendCustomVideoFrame:

- (V2TXLiveCode)sendCustomVideoFrame:	(V2TXLiveVideoFrame *)videoFrame

Sends the collected video data to the SDK in the custom video capture mode

In the custom video capture mode, the SDK no longer captures images from cameras. Only the encoding and sending capabilities are retained.

You can pack collected SampleBuffer packets into V2TXLiveVideoFrame and periodically send them through this API.

Param	DESC

videoFrame	Video frames sent to the SDK V2TXLiveVideoFrame.

Note

You must call enableCustomVideoCapture to enable custom video capture before startPush .

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_INVALID_PARAMETER: The video frames fail to be sent because they are invalid.

sendCustomAudioFrame:

sendCustomAudioFrame:

- (V2TXLiveCode)sendCustomAudioFrame:	(V2TXLiveAudioFrame *)audioFrame

In the custom audio collection mode, send the collected audio data to the SDK

Param	DESC
audioFrame	Audio frame data sent to SDK V2TXLiveAudioFrame.

Note

You need to call enableCustomAudioCapture(boolean) before startPush to enable custom capture.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_INVALID_PARAMETER` : The audio frames fail to be sent because they are invalid.

enableAudioProcessObserver:format:

enableAudioProcessObserver:format:

- (V2TXLiveCode)enableAudioProcessObserver:	(BOOL)enable
format:	(V2TXLiveAudioFrameObserverFormat *)format

Enables/Disables audio process callback

Param	DESC
enable	`YES` : enable; `NO` (**default**): disable.
format	audio frame format.

Note

This API works only if you call it before startPush.

sendSeiMessage:data:

sendSeiMessage:data:

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

Use SEI channel to send custom message

The player end V2TXLivePlayer can receive the message via `onReceiveSeiMessage` callback in V2TXLivePlayerObserver.

Param	DESC
data	Data to be sent.
payloadType	Payload type. Valid values: `5` , `242` , `242` recommended.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

showDebugView:

showDebugView:

- (void)showDebugView:	(BOOL)isShow

Indicates whether the debug view of the pusher video status information is displayed

Param	DESC
isShow	Specifies whether to display the debug view. `Default` : NO.

setProperty:value:

setProperty:value:

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

Calls the advanced API of V2TXLivePusher

Param	DESC
key	Key of the advanced API, please see V2TXLiveProperty.
value	Parameter needed to call the advanced API corresponding to the key.

Note

This API is used to call some advanced features.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The key cannot be nil.

setMixTranscodingConfig:

setMixTranscodingConfig:

- (V2TXLiveCode)setMixTranscodingConfig:	(V2TXLiveTranscodingConfig *)config

Sets On-Cloud MixTranscoding parameters

If you have enabled relayed push on the "Function Configuration" page of the TRTC console, then each stream in a room will have a default CDN address.

There may be multiple anchors in a room, each sending their own video and audio, but CDN audience needs only one live stream.
Therefore, you need to mix multiple audio/video streams into one standard live stream, which requires mixing and transcoding.

When you call the `setMixTranscodingConfig()` API, the SDK will send a command to the Tencent Cloud transcoding server to combine multiple audio/video streams in the room into one stream.
You can use the `mixUsers` parameter to set the position of each channel of image and specify whether to mix only audio. You can also set the encoding parameters of the mixed stream, including `videoWidth` , `videoHeight` , and `videoBitrate` .

Image 1 => decoding ====> \\
\\
Image 2=> decoding => image mixing => encoding => **mixed image**
/
Image 3 => decoding ====> /

Audio 1 => decoding ====> \\
\\
Audio 2 => decoding => audio mixing => encoding => **mixed audio**
/
Audio 3 => decoding ====> /

For more information, please see On-Cloud MixTranscoding.

Param	DESC
config	Please see the description of V2TXLiveTranscodingConfig in `V2TXLiveDef.h` . Passing in `nil` will cancel On-Cloud MixTranscoding.

Note

Notes:

On-Cloud MixTranscoding will increase the delay of CDN live streaming by about 1-2 seconds.

If you call this API, the streams of co-anchors will be mixed into your stream or the `streamId` specified in `config`.

If you are still in the room but do not need to mix streams anymore, make sure that you pass in `nil` to cancel On-Cloud MixTranscoding. The On-Cloud MixTranscoding module starts working the moment you enable On-Cloud MixTranscoding. You may incur additional costs if you do not cancel it in a timely manner.

When you leave the room, mixing will be canceled automatically.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_REFUSED` : failed to set On-Cloud MixTranscoding parameters as stream pushing has not started.

startLocalRecording:

startLocalRecording:

- (V2TXLiveCode)startLocalRecording:	(V2TXLiveLocalRecordingParams *)params

Start recording audio and video stream

Note

The recording can only be started after the push stream is started, and it is invalid to start the recording in the non-push state.

Do not dynamically switch the resolution and soft/hard editing during the recording process, as there is a high probability that the generated video will be abnormal.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_INVALID_PARAMETER` : The parameter is invalid, such as filePath is empty.

`V2TXLIVE_ERROR_REFUSED` : API refuse, you must first call startPush to start publishing streaming.

stopLocalRecording

stopLocalRecording

Stop recording audio and video stream

Note

When the push stream is stopped, if the video is still being recorded, the SDK will automatically end the recording.

enableVoiceActivityDetection:

enableVoiceActivityDetection:

- (void)enableVoiceActivityDetection:	(BOOL) enable

Enable voice activity detection

Note

After turning on, you can get the start and stop of voice activities in the OnVoiceActivityDetectionUpdate callback

V2TXLivePusherObserver

Last updated：2024-06-06 15:47:57

Module: V2TXLivePusherObserver @ TXLiteAVSDK

Function: Tencent Cloud live pusher callback notification

Function

Callback notification for push streaming of Tencent Cloud Live.

Introduce

You can receive some push notifications from the V2TXLivePusher pusher, including the connection status of the pusher, callback of the first frame of audio and video, statistical data, warning and error messages, etc.

V2TXLivePusherObserver

V2TXLivePusherObserver

FuncList	DESC
onError:message:extraInfo:	Live pusher error notification, which is called back when the pusher encounters an error
onWarning:message:extraInfo:	Live pusher warning notification
onCaptureFirstAudioFrame	Callback notification indicating that collection of the first audio frame is complete
onCaptureFirstVideoFrame	Callback notification indicating that collection of the first video frame is complete
onMicrophoneVolumeUpdate:	Microphone-collected volume callback

onPushStatusUpdate:message:extraInfo:	Callback notification of the pusher connection status
onStatisticsUpdate:	Live pusher statistics callback
onSnapshotComplete:	Screenshot callback
onProcessAudioFrame:	Audio data captured by the local mic, pre-processed by the audio module, effect-processed and BGM-mixed
onProcessVideoFrame:dstFrame:	Custom video processing callback
onGLContextDestroyed	Callback of destroying the OpenGL context in the SDK
onSetMixTranscodingConfig:message:	Callback of setting On-Cloud MixTranscoding parameters, which corresponds to the setMixTranscodingConfig API
onScreenCaptureStarted	The SDK returns this callback when you call startScreenCapture and other APIs to start screen sharing.
onScreenCaptureStopped:	The SDK returns this callback when you call stopScreenCapture to stop screen sharing
onLocalRecordBegin:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete:storagePath:	The SDK returns this callback when you call stopLocalRecording to start local recording.
onVoiceActivityDetectionUpdate:	After calling enableVoiceActivityDetection to turn on voice activity detection, you will receive this callback notification when the anchor starts or stops speaking.

onError:message:extraInfo:

onError:message:extraInfo:

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

Live pusher error notification, which is called back when the pusher encounters an error

Param	DESC
code	Error code V2TXLiveCode.
extraInfo	Extended information.
msg	Error message.

onWarning:message:extraInfo:

onWarning:message:extraInfo:

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

Live pusher warning notification

Param	DESC
code	Warning code V2TXLiveCode.
extraInfo	Extended information.
msg	Warning message.

onCaptureFirstAudioFrame

onCaptureFirstAudioFrame

Callback notification indicating that collection of the first audio frame is complete

onCaptureFirstVideoFrame

onCaptureFirstVideoFrame

Callback notification indicating that collection of the first video frame is complete

onMicrophoneVolumeUpdate:

onMicrophoneVolumeUpdate:

- (void)onMicrophoneVolumeUpdate:	(NSInteger)volume

Microphone-collected volume callback

Param	DESC
volume	Current volume value for collection.

Note

This callback notification is received after enableVolumeEvaluation is called.

onPushStatusUpdate:message:extraInfo:

onPushStatusUpdate:message:extraInfo:

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

Callback notification of the pusher connection status

Param	DESC
extraInfo	Extended information.
msg	Connection status message.
status	Pusher connection status V2TXLivePushStatus .

onStatisticsUpdate:

onStatisticsUpdate:

- (void)onStatisticsUpdate:	(V2TXLivePusherStatistics *)statistics

Live pusher statistics callback

Param	DESC
statistics	Pusher statistics V2TXLivePusherStatistics .

onSnapshotComplete:

onSnapshotComplete:

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

Screenshot callback

Param	DESC
image	Captured video image.

Note

This callback notification will be received after calling snapshot .

onProcessAudioFrame:

onProcessAudioFrame:

- (void) onProcessAudioFrame:	(V2TXLiveAudioFrame *)frame

Audio data captured by the local mic, pre-processed by the audio module, effect-processed and BGM-mixed

After you configure the callback of custom audio processing, the SDK will return via this callback the data captured, pre-processed (ANS, AEC, and AGC), effect-processed and BGM-mixed in PCM format, before it is submitted to the network module for encoding.

The audio data returned via this callback is in PCM format and has a fixed frame length (time) of 0.02s.

The formula to convert a frame length in seconds to one in bytes is **sample rate * frame length in seconds * number of sound channels * audio bit depth**.

Assume that the audio is recorded on a single channel with a sample rate of 48,000 Hz and audio bit depth of 16 bits, which are the default settings of TRTC. The frame length in bytes will be **48000 * 0.02s * 1 * 16 bits = 15360 bits = 1920 bytes**.

Param	DESC

frame	Audio frames in PCM format

Note

1. Please avoid time-consuming operations in this callback function. The SDK processes an audio frame every 20 ms, so if your operation takes more than 20 ms, it will cause audio exceptions.
2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.

onProcessVideoFrame:dstFrame:

onProcessVideoFrame:dstFrame:

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

Custom video processing callback

Param	DESC
dstFrame	For images after processing.
srcFrame	For images before processing.

Note

You will receive this callback only after you call enableCustomVideoProcess
to enable custom video processing.
Case 1: The beauty filter component generates new textures.
If the beauty filter component you use generates a new texture frame (for the processed image) during image processing, please set `dstFrame.textureId` to a new texture ID in the callback API.

```
(void) onProcessVideoFrame:(V2TXLiveVideoFrame * _Nonnull)srcFrame dstFrame:(V2TXLiveVideoFrame * _Nonnull)dstFrame
{
GLuint dstTextureId = renderItemWithTexture(srcFrame.textureId, srcFrame.width, srcFrame.height);
dstFrame.textureId = dstTextureId;
return 0;
}
```

Case 2: The third-party beauty filter component doesn't generate new textures.

If the third-party beauty filter component you use does not generate new textures and you need to manually set an input texture and an output texture for the component, please consider the following scheme:

```
(void) onProcessVideoFrame:(V2TXLiveVideoFrame * _Nonnull)srcFrame dstFrame:(V2TXLiveVideoFrame * _Nonnull)dstFrame
{
thirdparty_process(srcFrame.textureId, srcFrame.width, srcFrame.height, dstFrame.textureId);
return 0;
}
```

onGLContextDestroyed

onGLContextDestroyed

Callback of destroying the OpenGL context in the SDK

onSetMixTranscodingConfig:message:

onSetMixTranscodingConfig:message:

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

Callback of setting On-Cloud MixTranscoding parameters, which corresponds to the {@link setMixTranscodingConfig} API

Param	DESC
code	0: successful; other values: failed.
msg	Error message.

onScreenCaptureStarted

onScreenCaptureStarted

The SDK returns this callback when you call {@link startScreenCapture} and other APIs to start screen sharing.

onScreenCaptureStopped:

onScreenCaptureStopped:

- (void)onScreenCaptureStopped:	(int)reason

The SDK returns this callback when you call {@link stopScreenCapture} to stop screen sharing

Param	DESC
Reason	for stop. `0` : Screen capture stopped by user. `1` : On iOS platform means the screen recording is interrupted by the system; Mac, Windows means the screen sharing window is closed. `2` : On windows platform indicates that the display screen status of screen sharing is changed (such as the interface is pulled out, the projection mode is changed, etc.); other platforms do not throw.

onLocalRecordBegin:storagePath:

onLocalRecordBegin:storagePath:

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

The SDK returns this callback when you call {@link startLocalRecording} to start local recording.

Param	DESC
code	status. 0: successful. -1: failed. -2: unsupported format. -6: recording has been started. Stop recording first. -7: recording file already exists and needs to be deleted. -8: recording directory does not have the write permission. Please check the directory permission.
storagePath	recording filePath.

onLocalRecording:storagePath:

onLocalRecording:storagePath:

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

The SDK returns this callback when you call {@link startLocalRecording} to start local recording, which means recording task in progress.

Param	DESC
durationMs	recording duration.
storagePath	recording filePath.

onLocalRecordComplete:storagePath:

onLocalRecordComplete:storagePath:

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

The SDK returns this callback when you call {@link stopLocalRecording} to start local recording.

Param	DESC
code	status 0: successful. -1: failed. -2: Switching resolution or horizontal and vertical screen causes the recording to stop. -3: recording duration is too short or no video or audio data is received. Check the recording duration or whether audio or video capture is enabled.
storagePath	recording filePath.

onVoiceActivityDetectionUpdate:

onVoiceActivityDetectionUpdate:

- (void)onVoiceActivityDetectionUpdate:	(BOOL)active

After calling {@link enableVoiceActivityDetection} to turn on voice activity detection, you will receive this callback notification when the anchor starts or stops speaking.

Param	DESC
active	The voice starts or stops.

V2TXLivePlayer

Last updated：2024-06-06 15:47:57

Module: V2TXLivePlayer @ TXLiteAVSDK

Function: Tencent Cloud live player

Function

Tencent Cloud Live Player.
It is mainly responsible for pulling audio and video data from the specified live stream address, decoding and rendering locally.

Introduce

The player includes the following capabilities:
Support RTMP, HTTP-FLV, HLS, TRTC, WebRTC protocols.
Screen capture, you can capture the video screen of the current live stream.
Delay adjustment, you can set the minimum and maximum time for automatic adjustment of the player cache.
Customized video data processing, you can process the video data in the live stream according to the needs of the project, and then render and play it.

V2TXLivePlayer

V2TXLivePlayer

FuncList	DESC
setObserver:	Sets the player callback
setRenderView:	Sets the rendering view of the player. This control is responsible for presenting the video content

setRenderRotation:	Sets the rotation angle of the player view
setRenderFillMode:	Sets the fill mode of the view
startLivePlay:	Starts playing the audio and video streams
stopPlay	Stops playing the audio and video streams
isPlaying	Indicates whether the player is playing the audio and video streams
pauseAudio	Pauses the audio stream of the player
resumeAudio	Resumes the audio stream of the player
pauseVideo	Pauses the video stream of the player
resumeVideo	Resumes the video stream of the player
setPlayoutVolume:	Sets the volume
setCacheParams:maxTime:	Set the minimum time and maximum time (unit: s) for auto adjustment of the player cache
switchStream:	Seamlessly switch live stream urls, supporting FLV and LEB protocols
getStreamList	Get Stream Info List
enableVolumeEvaluation:	Enables playback volume update
snapshot	Captures the video view in the playback process
enableObserveVideoFrame:pixelFormat:bufferType:	Turn on/off the monitoring callback of the video frame
enableObserveAudioFrame:	Turn on/off the monitoring callback of the audio frame
enableReceiveSeiMessage:payloadType:	Enables the receiving of SEI messages
enablePictureInPicture:	Enables Picture-in-Picture mode
showDebugView:	Indicates whether the debug view of the player video status information is displayed
setProperty:value:	Calls the advanced API of V2TXLivePlayer

startLocalRecording:	Start recording audio and video stream
stopLocalRecording	Stop recording audio and video stream

setObserver:

setObserver:

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

Sets the player callback

By setting the callback, you can listen to some callback events of V2TXLivePlayer,
including the player status, playback volume callback, first frame audio/video callback, statistics, warnings, and error messages.

Param	DESC
observer	Callback target of the player. For more information, see V2TXLivePlayerObserver.

setRenderView:

setRenderView:

- (V2TXLiveCode)setRenderView:	(TXView *)view

Sets the rendering view of the player. This control is responsible for presenting the video content

Param	DESC
view	Player rendering view.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderRotation:

setRenderRotation:

- (V2TXLiveCode)setRenderRotation:	(V2TXLiveRotation)rotation

Sets the rotation angle of the player view

Param	DESC
rotation	Rotation angle of the view V2TXLiveRotation. V2TXLiveRotation0 **Default**: 0 degrees, which means the view is not rotated. V2TXLiveRotation90: rotate 90 degrees clockwise. V2TXLiveRotation180: rotate 180 degrees clockwise. V2TXLiveRotation270: rotate 270 degrees clockwise.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderFillMode:

setRenderFillMode:

- (V2TXLiveCode)setRenderFillMode:	(V2TXLiveFillMode)mode

Sets the fill mode of the view

Param	DESC
mode	Fill mode of the view V2TXLiveFillMode. V2TXLiveFillModeFill: **Default**: fill the screen with the image without leaving any black edges. If the aspect ratio of the view is different from that of the screen, part of the view will be cropped. V2TXLiveFillModeFit make the view fit the screen without cropping. If the aspect ratio of the view is different from that of the screen, black edges will appear. V2TXLiveFillModeScaleFill fill the screen with the stretched image, thus the length and width may not change proportionally.

Return Desc:

Return code V2TXLiveCode
V2TXLIVE_OK: successful

startLivePlay:

startLivePlay:

- (V2TXLiveCode)startLivePlay:	(NSString *)url

Starts playing the audio and video streams

Param	DESC
url	URL of the audio and video streams to be played. The RTMP, HTTP-FLV and TRTC streaming protocols are supported.

Note

Starting from version 10.7, the Licence needs to be set through setLicence or setLicence before it can be played successfully, otherwise the playback will fail (black screen), and it can only be set once globally. Live Licence, UGC Licence, and Player Licence can all be used. If you have not obtained the above Licence, you can `quickly apply for a beta Licence for free` To play, the official licence needs to be `purchased` .

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: operation succeeded. The player starts connecting to the URL and playing the audio and video streams.
V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The URL is invalid.
V2TXLIVE_ERROR_REFUSED: operation failed. Duplicate streamId, please ensure that no other player or pusher is using this streamId now.
V2TXLIVE_ERROR_INVALID_LICENSE: The licence is invalid and the playback fails.

stopPlay

stopPlay

Stops playing the audio and video streams

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

isPlaying

isPlaying

Indicates whether the player is playing the audio and video streams

Return Desc:

Indicates whether the player is playing the audio and video streams.
1: yes.
0: no.

pauseAudio

pauseAudio

Pauses the audio stream of the player

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeAudio

resumeAudio

Resumes the audio stream of the player

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

pauseVideo

pauseVideo

Pauses the video stream of the player

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeVideo

resumeVideo

Resumes the video stream of the player

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setPlayoutVolume:

setPlayoutVolume:

- (V2TXLiveCode)setPlayoutVolume:	(NSUInteger)volume

Sets the volume

Param	DESC
volume	Volume. Valid range: 0 - 100. **Default**: 100.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setCacheParams:maxTime:

setCacheParams:maxTime:

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

Set the minimum time and maximum time (unit: s) for auto adjustment of the player cache

Param	DESC
maxTime	Maximum time for auto cache adjustment. The value must be greater than 0. **Default**: 5.

minTime	Minimum time for auto cache adjustment. The value must be greater than 0. **Default**: 1.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. MinTime and maxTime must be greater than 0.

V2TXLIVE_ERROR_REFUSED: operation failed. Change of cache is not suppoted when playing.

switchStream:

switchStream:

- (V2TXLiveCode)switchStream:	(NSString*)newUrl

Seamlessly switch live stream urls, supporting FLV and LEB protocols

Param	DESC
newUrl	New pull address.

getStreamList

getStreamList

Get Stream Info List

enableVolumeEvaluation:

enableVolumeEvaluation:

- (V2TXLiveCode)enableVolumeEvaluation:	(NSUInteger)intervalMs

Enables playback volume update

After this feature is enabled, you can obtain the SDK's volume evaluation through the onPlayoutVolumeUpdate callback.

Param	DESC
intervalMs	Interval for triggering the volume callback. The unit is ms. The minimum interval is 100 ms.

	If the value is equal to or smaller than 0, the callback is disabled. We recommend that you set this parameter to 300 ms. **Default**: 0.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

snapshot

snapshot

Captures the video view in the playback process

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.
V2TXLIVE_ERROR_REFUSED: playback is stopped, the snapshot operation cannot be called.

enableObserveVideoFrame:pixelFormat:bufferType:

enableObserveVideoFrame:pixelFormat:bufferType:

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

Turn on/off the monitoring callback of the video frame

The SDK will no longer render the video after you turn on this switch. You can get the video frame through V2TXLivePlayerObserver and execute custom rendering logic.

Param	DESC
bufferType	Video data format for custom rendering callback V2TXLiveBufferType。
enable	Whether to enable custom rendering. **Default**: NO.
pixelFormat	Video pixel format for custom rendering callback V2TXLivePixelFormat。

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_NOT_SUPPORTED: the pixel format or data format is not supported.

enableObserveAudioFrame:

enableObserveAudioFrame:

- (V2TXLiveCode)enableObserveAudioFrame:	(BOOL)enable

Turn on/off the monitoring callback of the audio frame

if you turn on this switch, You can get the audio frame through V2TXLivePlayerObserver and execute custom logic.

Param	DESC
enable	Whether to enable the callback of the audio frame. **Default**: NO.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

enableReceiveSeiMessage:payloadType:

enableReceiveSeiMessage:payloadType:

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

Enables the receiving of SEI messages

Param	DESC
enable	`YES` : enable; `NO` (**default**): disable.
payloadType	The payload type of SEI messages. Valid values: `5` , `242` , `243` , please be consistent with the payload type of the sender.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

enablePictureInPicture:

enablePictureInPicture:

- (V2TXLiveCode)enablePictureInPicture:	(BOOL)enable

Enables Picture-in-Picture mode

Param	DESC
enable	`YES` : enable; `NO` (**default**): disable.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

showDebugView:

showDebugView:

- (void)showDebugView:	(BOOL)isShow

Indicates whether the debug view of the player video status information is displayed

Param	DESC
isShow	Specifies whether to display the debug view. **Default**: NO.

setProperty:value:

setProperty:value:

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

Calls the advanced API of V2TXLivePlayer

Param	DESC
key	Key of the advanced API, please see V2TXLiveProperty.
value	Parameter needed to call the advanced API corresponding to the key.

Note

This API is used to call some advanced features.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The key cannot be nil.

startLocalRecording:

startLocalRecording:

- (V2TXLiveCode)startLocalRecording:	(V2TXLiveLocalRecordingParams *)params

Start recording audio and video stream

Note

The recording can only be started after the play stream is started, and it is invalid to start the recording in the non-play state.

Do not dynamically switch soft/hard decoding during the recording process, as there is a high probability that the generated video will be abnormal.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_INVALID_PARAMETER` : The parameter is invalid, such as filePath is empty.

`V2TXLIVE_ERROR_REFUSED` : API refuse, you must first call startLivePlay to start playing streaming.

stopLocalRecording

stopLocalRecording

Stop recording audio and video stream

Note

When the play stream is stopped, if the video is still being recorded, the SDK will automatically end the recording.

V2TXLivePlayerObserver

Last updated：2024-06-06 15:47:57

Module: V2TXLivePlayerObserver @ TXLiteAVSDK

Function: Tencent Cloud live player callback notification

Function

Player callback notification for Tencent Cloud Live.

Introduce

You can receive some callback notifications from the V2TXLivePlayer player, including player status, playback volume callback, audio and video first frame callback, statistical data, warning and error messages, etc.

V2TXLivePlayerObserver

V2TXLivePlayerObserver

FuncList	DESC
onError:code:message:extraInfo:	live player error notification, which is called back when the player encounters an error
onWarning:code:message:extraInfo:	live player warning notification
onVideoResolutionChanged:width:height:	live player resolution change notification
onConnected:extraInfo:	live player has successfully connected to the server notification
onVideoPlaying:firstPlay:extraInfo:	Video playback event
onAudioPlaying:firstPlay:extraInfo:	Audio playback event

onVideoLoading:extraInfo:	Video loading event
onAudioLoading:extraInfo:	Audio loading event
onPlayoutVolumeUpdate:volume:	Player playback volume callback
onStatisticsUpdate:statistics:	Live player statistics callback
onSnapshotComplete:image:	Screenshot callback
onRenderVideoFrame:frame:	Custom video rendering callback
onPlayoutAudioFrame:frame:	Audio Data callback
onReceiveSeiMessage:payloadType:data:	Callback of receiving an SEI message. The sender calls `sendSeiMessage` in V2TXLivePusher to send an SEI
onStreamSwitched:url:code:	Resolution stream switch callback
onPictureInPictureStateUpdate:state:message:extraInfo:	Picture-in-Picture state change callback
onLocalRecordBegin:errCode:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording:durationMs:storagePath:	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete:errCode:storagePath:	The SDK returns this callback when you call stopLocalRecording to start local recording.

onError:code:message:extraInfo:

onError:code:message:extraInfo:

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

live player error notification, which is called back when the player encounters an error

Param	DESC
code	Error code V2TXLiveCode.
extraInfo	Extended information.
msg	Error message.
player	Player object that calls back this notification.

onWarning:code:message:extraInfo:

onWarning:code:message:extraInfo:

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

live player warning notification

Param	DESC
code	Warning code V2TXLiveCode.
extraInfo	Extended information.
msg	Warning message.
player	Player object that calls back this notification.

onVideoResolutionChanged:width:height:

onVideoResolutionChanged:width:height:

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

live player resolution change notification

Param	DESC
height	Video height.
player	Player object that calls back this notification.
width	Video width.

onConnected:extraInfo:

onConnected:extraInfo:

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

live player has successfully connected to the server notification

Param	DESC
extraInfo	Extended information.
player	Player object that calls back this notification.

onVideoPlaying:firstPlay:extraInfo:

onVideoPlaying:firstPlay:extraInfo:

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

Video playback event

Param	DESC
extraInfo	Extended information.
firstPlay	Play for the first time.

player	Player object that calls back this notification.

onAudioPlaying:firstPlay:extraInfo:

onAudioPlaying:firstPlay:extraInfo:

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

Audio playback event

Param	DESC
extraInfo	Extended information.
firstPlay	Play for the first time.
player	Player object that calls back this notification.

onVideoLoading:extraInfo:

onVideoLoading:extraInfo:

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

Video loading event

Param	DESC
extraInfo	Extended information.
player	Player object that calls back this notification.

onAudioLoading:extraInfo:

onAudioLoading:extraInfo:

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

Audio loading event

Param	DESC
extraInfo	Extended information.
player	Player object that calls back this notification.

onPlayoutVolumeUpdate:volume:

onPlayoutVolumeUpdate:volume:

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

Player playback volume callback

Param	DESC
player	Player object that calls back this notification.
volume	Current playback volume.

Note

This callback notification is received after enableVolumeEvaluation is called to enable playback volume display.

onStatisticsUpdate:statistics:

onStatisticsUpdate:statistics:

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

Live player statistics callback

Param	DESC

player	Player object that calls back this notification.
statistics	Player statistics V2TXLivePlayerStatistics.

onSnapshotComplete:image:

onSnapshotComplete:image:

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

Screenshot callback

Param	DESC
image	Captured video image.
player	Player object that calls back this notification.

Note

This callback notification is received after snapshot is called to snapshot.

onRenderVideoFrame:frame:

onRenderVideoFrame:frame:

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

Custom video rendering callback

Param	DESC
player	Player object that calls back this notification.
videoFrame	Video frame data V2TXLiveVideoFrame.

Note

Need you call enableObserveVideoFrame to turn on the callback switch.

onPlayoutAudioFrame:frame:

onPlayoutAudioFrame:frame:

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

Audio Data callback

Param	DESC
aduioFrame	Audio frame data V2TXLiveAudioFrame.
player	Player object that calls back this notification.

Note

Need you call enableObserveAudioFrame to turn on the callback switch. Please use the data of audioFrame in the current callback.

onReceiveSeiMessage:payloadType:data:

onReceiveSeiMessage:payloadType:data:

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

Callback of receiving an SEI message. The sender calls `sendSeiMessage` in {@link V2TXLivePusher} to send an SEI

Param	DESC
data	sei message data.
payloadType	The payload type of the received SEI message.
player	Player object that calls back this notification.

Note

You will receive this callback after calling `enableReceiveSeiMessage` in V2TXLivePlayer to enable the receiving of SEI.

onStreamSwitched:url:code:

onStreamSwitched:url:code:

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

Resolution stream switch callback

Param	DESC
code	Status code, 0:success, -1:timeout, -2:failed, server error, -3:failed, client error.
player	Player object that calls back this notification.
url	Switched playback address.

Note

This callback notification is received after switchStream is called to switch stream.

onPictureInPictureStateUpdate:state:message:extraInfo:

onPictureInPictureStateUpdate:state:message:extraInfo:

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

Picture-in-Picture state change callback

Param	DESC
extraInfo	Extended information.

player	Player object that calls back this notification.
state	Picture-in-Picture state。

Note

This callback notification is received after enablePictureInPicture is called to enable Picture-in-Picture.

onLocalRecordBegin:errCode:storagePath:

onLocalRecordBegin:errCode:storagePath:

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

The SDK returns this callback when you call {@link startLocalRecording} to start local recording.

Param	DESC
code	status. 0: successful. -1: failed. -2: unsupported format. -6: recording has been started. Stop recording first. -7: recording file already exists and needs to be deleted. -8: recording directory does not have the write permission. Please check the directory permission.
player	Player object that calls back this notification.
storagePath	recording filePath.

onLocalRecording:durationMs:storagePath:

onLocalRecording:durationMs:storagePath:

- (void)onLocalRecording:	(id<V2TXLivePlayer>)player
durationMs:	(NSInteger)durationMs

storagePath:	(NSString *)storagePath

The SDK returns this callback when you call {@link startLocalRecording} to start local recording, which means recording task in progress.

Param	DESC
durationMs	recording duration.
player	Player object that calls back this notification.
storagePath	recording filePath.

onLocalRecordComplete:errCode:storagePath:

onLocalRecordComplete:errCode:storagePath:

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

The SDK returns this callback when you call {@link stopLocalRecording} to start local recording.

Param	DESC
code	status 0: successful. -1: failed. -2: Switching resolution or horizontal and vertical screen causes the recording to stop. -3: recording duration is too short or no video or audio data is received. Check the recording duration or whether audio or video capture is enabled.
player	Player object that calls back this notification.
storagePath	recording filePath.

V2TXLivePremier

Last updated：2024-06-06 15:47:57

Module: V2TXLivePremier @ TXLiteAVSDK

Function: V2TXLive High-level interface

V2TXLivePremier

V2TXLivePremier

FuncList	DESC
getSDKVersionStr	Get the SDK version number
setObserver:	Set V2TXLivePremier callback interface
setLogConfig:	Set Log configuration information
setEnvironment:	Set up SDK access environment
setLicence:key:	Set SDK authorization license
setSocks5Proxy:port:username:password:config:	Set SDK socks5 proxy config
enableAudioCaptureObserver:format:	Enables/Disables audio capture callback
enableAudioPlayoutObserver:format:	Enables/Disables audio playout callback
enableVoiceEarMonitorObserver:	Enables/Disables in-ear monitoring callback
setUserId:	Set user id
callExperimentalAPI:	Call experimental APIs

V2TXLivePremierObserver

FuncList	DESC

onLog:log:	Custom Log output callback interface
onLicenceLoaded:Reason:	setLicence result callback interface
onCaptureAudioFrame:	Raw audio data captured locally
onPlayoutAudioFrame:	Data mixed from each channel before being submitted to the system for playback
onVoiceEarMonitorAudioFrame:	In-ear monitoring data

getSDKVersionStr

getSDKVersionStr

Get the SDK version number

setObserver:

setObserver:

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

Set V2TXLivePremier callback interface

setLogConfig:

setLogConfig:

+ (V2TXLiveCode)setLogConfig:	(V2TXLiveLogConfig *)config

Set Log configuration information

setEnvironment:

setEnvironment:

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

Set up SDK access environment

Param	DESC
env	currently supports two parameters "default" and "GDPR". default: In the default environment, the SDK will find the best access point in the world for access. GDPR: All audio and video data and quality statistics will not pass through servers in mainland China.

Note

If your application has no special requirements, please do not call this interface for setting.

setLicence:key:

setLicence:key:

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

Set SDK authorization license

Try and Purchase a License: https://www.tencentcloud.com/document/product/1071/38546.

Param	DESC
key	the key of licence.
url	the url of licence.

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

Set SDK socks5 proxy config

Param	DESC
config	protocol configured with socks5 proxy.
host	socks5 proxy host.
password	socks5 proxy password.
port	socks5 proxy port.
username	socks5 proxy username.

enableAudioCaptureObserver:format:

enableAudioCaptureObserver:format:

+ (V2TXLiveCode)enableAudioCaptureObserver:	(BOOL)enable
format:	(V2TXLiveAudioFrameObserverFormat *)format

Enables/Disables audio capture callback

Param	DESC
enable	`YES` : enable; `NO` (**default**): disable.
format	audio frame format.

Note

This API works only if you call it before startPush.

enableAudioPlayoutObserver:format:

enableAudioPlayoutObserver:format:

+ (V2TXLiveCode)enableAudioPlayoutObserver:	(BOOL)enable
format:	(V2TXLiveAudioFrameObserverFormat *)format

Enables/Disables audio playout callback

Param	DESC
enable	`YES` : enable; `NO` (**default**): disable.
format	audio frame format.

enableVoiceEarMonitorObserver:

enableVoiceEarMonitorObserver:

+ (V2TXLiveCode)enableVoiceEarMonitorObserver:	(BOOL)enable

Enables/Disables in-ear monitoring callback

Param	DESC
enable	`YES` : enable; `NO` (**default**): disable.

setUserId:

setUserId:

+ (void)setUserId:	(NSString *)userId

Set user id

Param	DESC
userId	User/device id maintained by the service side itself.

callExperimentalAPI:

callExperimentalAPI:

+ (V2TXLiveCode)callExperimentalAPI:	(NSString *)jsonStr

Call experimental APIs

Param	DESC
jsonStr	JSON string describing interface and parameters.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.
V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed because of illegal parameter.

onLog:log:

onLog:log:

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

Custom Log output callback interface

onLicenceLoaded:Reason:

onLicenceLoaded:Reason:

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

setLicence result callback interface

Param	DESC
reason	the reason for failure.
result	the result of setLicence interface, 0 succeeds, negative number fails.

onCaptureAudioFrame:

onCaptureAudioFrame:

- (void) onCaptureAudioFrame:	(V2TXLiveAudioFrame *)frame

Raw audio data captured locally

Param	DESC
frame	Audio frames in PCM format.

Note

1. Please avoid time-consuming operations in this callback function. The SDK processes an audio frame every 20 ms, so if your operation takes more than 20 ms, it will cause audio exceptions.
2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.
3. The audio data returned via this callback **does not include** pre-processing effects like background music, audio effects, or reverb, and therefore has a very short delay.

onPlayoutAudioFrame:

onPlayoutAudioFrame:

- (void) onPlayoutAudioFrame:	(V2TXLiveAudioFrame *)frame

Data mixed from each channel before being submitted to the system for playback

After you configure the callback of custom audio processing, the SDK will return to you via this callback the data (PCM format) mixed from each channel before it is submitted to the system for playback.
The audio data returned via this callback is in PCM format and has a fixed frame length (time) of 0.02s.
The formula to convert a frame length in seconds to one in bytes is **sample rate * frame length in seconds * number of sound channels * audio bit depth**.
Assume that the audio is recorded on a single channel with a sample rate of 48,000 Hz and audio bit depth of 16 bits, which are the default settings of SDK. The frame length in bytes will be **48000 * 0.02s * 1 * 16 bits = 15360 bits = 1920 bytes**.

Param	DESC
frame	Audio frames in PCM format.

Note

1. Please avoid time-consuming operations in this callback function. The SDK processes an audio frame every 20 ms, so if your operation takes more than 20 ms, it will cause audio exceptions.
2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.

3. The audio data returned via this callback is the audio data mixed from each channel before it is played. It does not include the in-ear monitoring data.

onVoiceEarMonitorAudioFrame:

onVoiceEarMonitorAudioFrame:

- (void) onVoiceEarMonitorAudioFrame:	(V2TXLiveAudioFrame *)frame

In-ear monitoring data

After you configure the callback of custom audio processing, the SDK will return to you via this callback the in-ear monitoring data (PCM format) before it is submitted to the system for playback.

The audio returned is in PCM format and has a not-fixed frame length (time).

The formula to convert a frame length in seconds to one in bytes is **sample rate * frame length in seconds * number of sound channels * audio bit depth**.

Assume that the audio is recorded on a single channel with a sample rate of 48,000 Hz and audio bit depth of 16 bits, which are the default settings of TRTC. The length of 0.02s frame in bytes will be **48000 * 0.02s * 1 * 16 bits = 15360 bits = 1920 bytes**.

Param	DESC
frame	Audio frames in PCM format

Note

1. Please avoid time-consuming operations in this callback function, or it will cause audio exceptions.
2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.

TXAudioEffectManager

Last updated：2024-06-06 15:47:57

Module: management class for background music, short audio effects, and voice effects

Description: sets background music, short audio effects, and voice effects

TXAudioEffectManager

TXAudioEffectManager

FuncList	DESC
enableVoiceEarMonitor:	Enabling in-ear monitoring
setVoiceEarMonitorVolume:	Setting in-ear monitoring volume
setVoiceReverbType:	Setting voice reverb effects
setVoiceChangerType:	Setting voice changing effects
setVoiceVolume:	Setting speech volume
setVoicePitch:	Setting speech pitch
startPlayMusic:onStart:onProgress:onComplete:	Starting background music
stopPlayMusic:	Stopping background music
pausePlayMusic:	Pausing background music
resumePlayMusic:	Resuming background music
setAllMusicVolume:	Setting the local and remote playback volume of background music
setMusicPublishVolume:volume:	Setting the remote playback volume of a specific music track
setMusicPlayoutVolume:volume:	Setting the local playback volume of a specific music track

setMusicPitch:pitch:	Adjusting the pitch of background music
setMusicSpeedRate:speedRate:	Changing the speed of background music
getMusicCurrentPosInMS:	Getting the playback progress (ms) of background music
getMusicDurationInMS:	Getting the total length (ms) of background music
seekMusicToPosInMS:pts:	Setting the playback progress (ms) of background music
setMusicScratchSpeedRate:speedRate:	Adjust the speed change effect of the scratch disc
preloadMusic:onProgress:onError:	Preload background music
getMusicTrackCount:	Get the number of tracks of background music
setMusicTrack:track:	Specify the playback track of background music

StructType

FuncList	DESC
TXAudioMusicParam	Background music playback information

EnumType

EnumType	DESC
TXVoiceReverbType	Reverb effects
TXVoiceChangeType	Voice changing effects

enableVoiceEarMonitor:

enableVoiceEarMonitor:

- (void)enableVoiceEarMonitor:	(BOOL)enable

Enabling in-ear monitoring

After enabling in-ear monitoring, anchors can hear in earphones their own voice captured by the mic. This is designed for singing scenarios.

In-ear monitoring cannot be enabled for Bluetooth earphones. This is because Bluetooth earphones have high latency. Please ask anchors to use wired earphones via a UI reminder.
Given that not all phones deliver excellent in-ear monitoring effects, we have blocked this feature on some phones.

Param	DESC
enable	`YES:` enable; `NO` : disable

Note

In-ear monitoring can be enabled only when earphones are used. Please remind anchors to use wired earphones.

setVoiceEarMonitorVolume:

setVoiceEarMonitorVolume:

- (void)setVoiceEarMonitorVolume:	(NSInteger)volume

Setting in-ear monitoring volume

This API is used to set the volume of in-ear monitoring.

Param	DESC
volume	Volume. Value range: 0-100; default: 100

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setVoiceReverbType:

setVoiceReverbType:

- (void)setVoiceReverbType:	(TXVoiceReverbType)reverbType

Setting voice reverb effects

This API is used to set reverb effects for human voice. For the effects supported, please see TXVoiceReverbType.

Note

Effects become invalid after room exit. If you want to use the same effect after you enter the room again, you need to set the effect again using this API.

setVoiceChangerType:

setVoiceChangerType:

- (void)setVoiceChangerType:	(TXVoiceChangeType)changerType

Setting voice changing effects

This API is used to set voice changing effects. For the effects supported, please see TXVoiceChangeType.

Note

Effects become invalid after room exit. If you want to use the same effect after you enter the room again, you need to set the effect again using this API.

setVoiceVolume:

setVoiceVolume:

- (void)setVoiceVolume:	(NSInteger)volume

Setting speech volume

This API is used to set the volume of speech. It is often used together with the music volume setting API setAllMusicVolume to balance between the volume of music and speech.

Param	DESC
volume	Volume. Value range: 0-100; default: 100

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setVoicePitch:

setVoicePitch:

-(void)setVoicePitch:	(double)pitch

Setting speech pitch

This API is used to set the pitch of speech.

Param	DESC
pitch	Ptich，Value range: -1.0f~1.0f; default: 0.0f。

startPlayMusic:onStart:onProgress:onComplete:

startPlayMusic:onStart:onProgress:onComplete:

- (void)startPlayMusic:	(TXAudioMusicParam *)musicParam
onStart:	(TXAudioMusicStartBlock _Nullable)startBlock
onProgress:	(TXAudioMusicProgressBlock _Nullable)progressBlock
onComplete:	(TXAudioMusicCompleteBlock _Nullable)completeBlock

Starting background music

You must assign an ID to each music track so that you can start, stop, or set the volume of music tracks by ID.

Param	DESC
completeBlock	Callback of ending music
musicParam	Music parameter
progressBlock	Callback of playback progress
startBlock	Callback of starting music

Note

1. If you play the same music track multiple times, please use the same ID instead of a separate ID for each playback.
2. If you want to play different music tracks at the same time, use different IDs for them.
3. If you use the same ID to play a music track different from the current one, the SDK will stop the current one before playing the new one.

stopPlayMusic:

stopPlayMusic:

- (void)stopPlayMusic:	(int32_t)id

Stopping background music

Param	DESC
id	Music ID

pausePlayMusic:

pausePlayMusic:

- (void)pausePlayMusic:	(int32_t)id

Pausing background music

Param	DESC
id	Music ID

resumePlayMusic:

resumePlayMusic:

- (void)resumePlayMusic:	(int32_t)id

Resuming background music

Param	DESC
id	Music ID

setAllMusicVolume:

setAllMusicVolume:

- (void)setAllMusicVolume:	(NSInteger)volume

Setting the local and remote playback volume of background music

This API is used to set the local and remote playback volume of background music.

Local volume: the volume of music heard by anchors

Remote volume: the volume of music heard by audience

Param	DESC
volume	Volume. Value range: 0-100; default: 60

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setMusicPublishVolume:volume:

setMusicPublishVolume:volume:

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

Setting the remote playback volume of a specific music track

This API is used to control the remote playback volume (the volume heard by audience) of a specific music track.

Param	DESC
id	Music ID
volume	Volume. Value range: 0-100; default: 60

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setMusicPlayoutVolume:volume:

setMusicPlayoutVolume:volume:

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

Setting the local playback volume of a specific music track

This API is used to control the local playback volume (the volume heard by anchors) of a specific music track.

Param	DESC
id	Music ID
volume	Volume. Value range: 0-100. default: 60

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setMusicPitch:pitch:

setMusicPitch:pitch:

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

Adjusting the pitch of background music

Param	DESC
id	Music ID
pitch	Pitch. Value range: floating point numbers in the range of [-1, 1]; default: 0.0f

setMusicSpeedRate:speedRate:

setMusicSpeedRate:speedRate:

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

Changing the speed of background music

Param	DESC
id	Music ID
speedRate	Music speed. Value range: floating point numbers in the range of [0.5, 2]; default: 1.0f

getMusicCurrentPosInMS:

getMusicCurrentPosInMS:

- (NSInteger)getMusicCurrentPosInMS:	(int32_t)id

Getting the playback progress (ms) of background music

Param	DESC
id	Music ID

Return Desc:

The milliseconds that have passed since playback started. -1 indicates failure to get the the playback progress.

getMusicDurationInMS:

getMusicDurationInMS:

- (NSInteger)getMusicDurationInMS:	(NSString *)path

Getting the total length (ms) of background music

Param	DESC
path	Path of the music file.

Return Desc:

The length of the specified music file is returned. -1 indicates failure to get the length.

seekMusicToPosInMS:pts:

seekMusicToPosInMS:pts:

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

Setting the playback progress (ms) of background music

Param	DESC
id	Music ID
pts	Unit: millisecond

Note

Do not call this API frequently as the music file may be read and written to each time the API is called, which can be time-consuming.

Wait till users finish dragging the progress bar before you call this API.

The progress bar controller on the UI tends to update the progress at a high frequency as users drag the progress bar. This will result in poor user experience unless you limit the frequency.

setMusicScratchSpeedRate:speedRate:

setMusicScratchSpeedRate:speedRate:

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

Adjust the speed change effect of the scratch disc

Param	DESC
id	Music ID
scratchSpeedRate	Scratch disc speed, the default value is 1.0f, the range is: a floating point number between [-12.0 ~ 12.0], the positive/negative speed value indicates the direction is positive/negative, and the absolute value indicates the speed.

Note

Precondition preloadMusic succeeds.

preloadMusic:onProgress:onError:

preloadMusic:onProgress:onError:

- (void)preloadMusic:	(TXAudioMusicParam *)preloadParam
onProgress:	(TXMusicPreloadProgressBlock _Nullable)progressBlock

onError:	(TXMusicPreloadErrorBlock _Nullable)errorBlock

Preload background music

You must assign an ID to each music track so that you can start, stop, or set the volume of music tracks by ID.

Param	DESC
musicParam	Music parameter

Note

1. Preload supports up to 2 preloads with different IDs at the same time, and the preload time does not exceed 10 minutes,you need to stopPlayMusic after use, otherwise the memory will not be released.
2. If the music corresponding to the ID is being played, the preloading fails, and stopPlayMusic must be called first.
3. When the musicParam passed to startPlayMusic is exactly the same, preloading works.

getMusicTrackCount:

getMusicTrackCount:

- (NSInteger)getMusicTrackCount:	(int32_t)id

Get the number of tracks of background music

Param	DESC
id	Music ID

setMusicTrack:track:

setMusicTrack:track:

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

Specify the playback track of background music

Param	DESC
id	Music ID

index	Specify which track to play (the first track is played by default). Value range [0, total number of tracks).

Note

The total number of tracks can be obtained through the getMusicTrackCount interface.

TXVoiceReverbType

TXVoiceReverbType

Reverb effects

Reverb effects can be applied to human voice. Based on acoustic algorithms, they can mimic voice in different environments. The following effects are supported currently:
0: original; 1: karaoke; 2: room; 3: hall; 4: low and deep; 5: resonant; 6: metal; 7: husky; 8: ethereal; 9: studio; 10: melodious; 11: studio2;

Enum	Value	DESC
TXVoiceReverbType_0	0	disable
TXVoiceReverbType_1	1	KTV
TXVoiceReverbType_2	2	small room
TXVoiceReverbType_3	3	great hall
TXVoiceReverbType_4	4	deep voice
TXVoiceReverbType_5	5	loud voice
TXVoiceReverbType_6	6	metallic sound
TXVoiceReverbType_7	7	magnetic sound
TXVoiceReverbType_8	8	ethereal
TXVoiceReverbType_9	9	studio
TXVoiceReverbType_10	10	melodious
TXVoiceReverbType_11	11	studio2

TXVoiceChangeType

TXVoiceChangeType

Voice changing effects

Voice changing effects can be applied to human voice. Based on acoustic algorithms, they change the tone of voice. The following effects are supported currently:
0: original; 1: child; 2: little girl; 3: middle-aged man; 4: metal; 5: nasal; 6: foreign accent; 7: trapped beast; 8: otaku; 9: electric; 10: robot; 11: ethereal

Enum	Value	DESC
TXVoiceChangeType_0	0	disable
TXVoiceChangeType_1	1	naughty kid
TXVoiceChangeType_2	2	Lolita
TXVoiceChangeType_3	3	uncle
TXVoiceChangeType_4	4	heavy metal
TXVoiceChangeType_5	5	catch cold
TXVoiceChangeType_6	6	foreign accent
TXVoiceChangeType_7	7	caged animal trapped beast
TXVoiceChangeType_8	8	indoorsman
TXVoiceChangeType_9	9	strong current
TXVoiceChangeType_10	10	heavy machinery
TXVoiceChangeType_11	11	intangible

TXAudioMusicParam

TXAudioMusicParam

Background music playback information

The information, including playback ID, file path, and loop times, is passed in the startPlayMusic API.
1. If you play the same music track multiple times, please use the same ID instead of a separate ID for each playback.
2. If you want to play different music tracks at the same time, use different IDs for them.
3. If you use the same ID to play a music track different from the current one, the SDK will stop the current one before playing the new one.

EnumType	DESC
ID	`Field description:` music ID **Note** the SDK supports playing multiple music tracks. IDs are used to distinguish different music tracks and control their start, end, volume, etc.
endTimeMS	`Field description:` the point in time in milliseconds for ending music playback. 0 indicates that playback continues till the end of the music track.
isShortFile	`Field description:` whether the music played is a short music track `Valid values:` `YES` : short music track that needs to be looped; `NO` (default): normal-length music track
loopCount	`Field description:` number of times the music track is looped `Valid values:` 0 or any positive integer. 0 (default) indicates that the music is played once, 1 twice, and so on.
path	`Field description:` absolute path of the music file or url.the mp3,aac,m4a,wav supported.
publish	`Field description:` whether to send the music to remote users `Valid values:` `YES` : remote users can hear the music played locally; `NO` (default): only the local user can hear the music.
startTimeMS	`Field description:` the point in time in milliseconds for starting music playback

TXBeautyManager

Last updated：2024-06-06 15:47:57

Module: beauty filter and image processing parameter configurations

Function: you can modify parameters such as beautification, filter, and green screen

TXBeautyManager

TXBeautyManager

FuncList	DESC
setBeautyStyle:	Sets the beauty (skin smoothing) filter algorithm.
setBeautyLevel:	Sets the strength of the beauty filter.
setWhitenessLevel:	Sets the strength of the brightening filter.
enableSharpnessEnhancement:	Enables clarity enhancement.
setRuddyLevel:	Sets the strength of the rosy skin filter.
setFilter:	Sets color filter.
setFilterStrength:	Sets the strength of color filter.
setGreenScreenFile:	Sets green screen video
setEyeScaleLevel:	Sets the strength of the eye enlarging filter.
setFaceSlimLevel:	Sets the strength of the face slimming filter.
setFaceVLevel:	Sets the strength of the chin slimming filter.
setChinLevel:	Sets the strength of the chin lengthening/shortening filter.
setFaceShortLevel:	Sets the strength of the face shortening filter.
setFaceNarrowLevel:	Sets the strength of the face narrowing filter.

setNoseSlimLevel:	Sets the strength of the nose slimming filter.
setEyeLightenLevel:	Sets the strength of the eye brightening filter.
setToothWhitenLevel:	Sets the strength of the teeth whitening filter.
setWrinkleRemoveLevel:	Sets the strength of the wrinkle removal filter.
setPounchRemoveLevel:	Sets the strength of the eye bag removal filter.
setSmileLinesRemoveLevel:	Sets the strength of the smile line removal filter.
setForeheadLevel:	Sets the strength of the hairline adjustment filter.
setEyeDistanceLevel:	Sets the strength of the eye distance adjustment filter.
setEyeAngleLevel:	Sets the strength of the eye corner adjustment filter.
setMouthShapeLevel:	Sets the strength of the mouth shape adjustment filter.
setNoseWingLevel:	Sets the strength of the nose wing narrowing filter.
setNosePositionLevel:	Sets the strength of the nose position adjustment filter.
setLipsThicknessLevel:	Sets the strength of the lip thickness adjustment filter.
setFaceBeautyLevel:	Sets the strength of the face shape adjustment filter.
setMotionTmpl:inDir:	Selects the AI animated effect pendant.
setMotionMute:	Sets whether to mute during animated effect playback.

EnumType

EnumType	DESC
TXBeautyStyle	Beauty (skin smoothing) filter algorithm

setBeautyStyle:

setBeautyStyle:

- (void)setBeautyStyle:	(TXBeautyStyle)beautyStyle

Sets the beauty (skin smoothing) filter algorithm.

TRTC has multiple built-in skin smoothing algorithms. You can select the one most suitable for your product needs:

Param	DESC
beautyStyle	Beauty filter style. `TXBeautyStyleSmooth` : smooth; `TXBeautyStyleNature` : natural; `TXBeautyStylePitu` : Pitu

setBeautyLevel:

setBeautyLevel:

- (void)setBeautyLevel:	(float)beautyLevel

Sets the strength of the beauty filter.

Param	DESC
beautyLevel	Strength of the beauty filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

setWhitenessLevel:

setWhitenessLevel:

- (void)setWhitenessLevel:	(float)whitenessLevel

Sets the strength of the brightening filter.

Param	DESC
whitenessLevel	Strength of the brightening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

enableSharpnessEnhancement:

enableSharpnessEnhancement:

- (void)enableSharpnessEnhancement:	(BOOL)enable

Enables clarity enhancement.

setRuddyLevel:

setRuddyLevel:

- (void)setRuddyLevel:	(float)ruddyLevel

Sets the strength of the rosy skin filter.

Param	DESC
ruddyLevel	Strength of the rosy skin filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

setFilter:

setFilter:

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

Sets color filter.

The color filter is a color lookup table image containing color mapping relationships. You can find several predefined filter images in the official demo we provide.
The SDK performs secondary processing on the original video image captured by the camera according to the mapping relationships in the lookup table to achieve the expected filter effect.

Param	DESC
image	Color lookup table containing color mapping relationships. The image must be in PNG format.

setFilterStrength:

setFilterStrength:

- (void)setFilterStrength:	(float)strength

Sets the strength of color filter.

The larger this value, the more obvious the effect of the color filter, and the greater the color difference between the video image processed by the filter and the original video image.
The default strength is 0.5, and if it is not sufficient, it can be adjusted to a value above 0.5. The maximum value is 1.

Param	DESC
strength	Value range: 0–1. The greater the value, the more obvious the effect. Default value: 0.5

setGreenScreenFile:

setGreenScreenFile:

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

Sets green screen video

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.
The green screen feature enabled by this API is not capable of intelligent keying. It requires that there be a green screen behind the videoed person or object for further chroma keying.

Param	DESC
path	Path of the video file in MP4 format. An empty value indicates to disable the effect.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeScaleLevel:

setEyeScaleLevel:

- (int)setEyeScaleLevel:	(float)eyeScaleLevel

Sets the strength of the eye enlarging filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
eyeScaleLevel	Strength of the eye enlarging filter. Value range: 0–9. `0` indicates to disable the

	filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceSlimLevel:

setFaceSlimLevel:

- (int)setFaceSlimLevel:	(float)faceSlimLevel

Sets the strength of the face slimming filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceSlimLevel	Strength of the face slimming filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceVLevel:

setFaceVLevel:

- (int)setFaceVLevel:	(float)faceVLevel

Sets the strength of the chin slimming filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceVLevel	Strength of the chin slimming filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setChinLevel:

setChinLevel:

- (int)setChinLevel:	(float)chinLevel

Sets the strength of the chin lengthening/shortening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
chinLevel	Strength of the chin lengthening/shortening filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates that the chin is shortened, and a value greater than 0 indicates that the chin is lengthened.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceShortLevel:

setFaceShortLevel:

- (int)setFaceShortLevel:	(float)faceShortLevel

Sets the strength of the face shortening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceShortLevel	Strength of the face shortening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceNarrowLevel:

setFaceNarrowLevel:

- (int)setFaceNarrowLevel:	(float)faceNarrowLevel

Sets the strength of the face narrowing filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
level	Strength of the face narrowing filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setNoseSlimLevel:

setNoseSlimLevel:

- (int)setNoseSlimLevel:	(float)noseSlimLevel

Sets the strength of the nose slimming filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
noseSlimLevel	Strength of the nose slimming filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeLightenLevel:

setEyeLightenLevel:

- (int)setEyeLightenLevel:	(float)eyeLightenLevel

Sets the strength of the eye brightening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
eyeLightenLevel	Strength of the eye brightening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setToothWhitenLevel:

setToothWhitenLevel:

- (int)setToothWhitenLevel:	(float)toothWhitenLevel

Sets the strength of the teeth whitening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
toothWhitenLevel	Strength of the teeth whitening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setWrinkleRemoveLevel:

setWrinkleRemoveLevel:

- (int)setWrinkleRemoveLevel:	(float)wrinkleRemoveLevel

Sets the strength of the wrinkle removal filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
wrinkleRemoveLevel	Strength of the wrinkle removal filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setPounchRemoveLevel:

setPounchRemoveLevel:

- (int)setPounchRemoveLevel:	(float)pounchRemoveLevel

Sets the strength of the eye bag removal filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
pounchRemoveLevel	Strength of the eye bag removal filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setSmileLinesRemoveLevel:

setSmileLinesRemoveLevel:

- (int)setSmileLinesRemoveLevel:	(float)smileLinesRemoveLevel

Sets the strength of the smile line removal filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
smileLinesRemoveLevel	Strength of the smile line removal filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setForeheadLevel:

setForeheadLevel:

- (int)setForeheadLevel:	(float)foreheadLevel

Sets the strength of the hairline adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
foreheadLevel	Strength of the hairline adjustment filter. Value range: -9–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeDistanceLevel:

setEyeDistanceLevel:

- (int)setEyeDistanceLevel:	(float)eyeDistanceLevel

Sets the strength of the eye distance adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC

eyeDistanceLevel	Strength of the eye distance adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to widen, and a value greater than 0 indicates to narrow.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeAngleLevel:

setEyeAngleLevel:

- (int)setEyeAngleLevel:	(float)eyeAngleLevel

Sets the strength of the eye corner adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
eyeAngleLevel	Strength of the eye corner adjustment filter. Value range: -9–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setMouthShapeLevel:

setMouthShapeLevel:

- (int)setMouthShapeLevel:	(float)mouthShapeLevel

Sets the strength of the mouth shape adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
mouthShapeLevel	Strength of the mouth shape adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to widen, and a value greater

	than 0 indicates to narrow.

Return Desc:

0: Success; -5: feature of license not supported.

setNoseWingLevel:

setNoseWingLevel:

- (int)setNoseWingLevel:	(float)noseWingLevel

Sets the strength of the nose wing narrowing filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
noseWingLevel	Strength of the nose wing adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to widen, and a value greater than 0 indicates to narrow.

Return Desc:

0: Success; -5: feature of license not supported.

setNosePositionLevel:

setNosePositionLevel:

- (int)setNosePositionLevel:	(float)nosePositionLevel

Sets the strength of the nose position adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
nosePositionLevel	Strength of the nose position adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to lift, and a value greater than 0 indicates to lower.

Return Desc:

0: Success; -5: feature of license not supported.

setLipsThicknessLevel:

setLipsThicknessLevel:

- (int)setLipsThicknessLevel:	(float)lipsThicknessLevel

Sets the strength of the lip thickness adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
lipsThicknessLevel	Strength of the lip thickness adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to thicken, and a value greater than 0 indicates to thin.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceBeautyLevel:

setFaceBeautyLevel:

- (int)setFaceBeautyLevel:	(float)faceBeautyLevel

Sets the strength of the face shape adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceBeautyLevel	Strength of the face shape adjustment filter. Value range: 0–9. `0` indicates to disable the filter, and the greater the value, the more obvious the effect.

Return Desc:

0: Success; -5: feature of license not supported.

setMotionTmpl:inDir:

setMotionTmpl:inDir:

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

Selects the AI animated effect pendant.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
tmplDir	Directory of the animated effect material file
tmplName	Animated effect pendant name

setMotionMute:

setMotionMute:

- (void)setMotionMute:	(BOOL)motionMute

Sets whether to mute during animated effect playback.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect. Some animated effects have audio effects, which can be disabled through this API when they are played back.

Param	DESC
motionMute	`YES` : mute; `NO` : unmute

TXBeautyStyle

TXBeautyStyle

Beauty (skin smoothing) filter algorithm

TRTC has multiple built-in skin smoothing algorithms. You can select the one most suitable for your product needs.

Enum	Value	DESC

TXBeautyStyleSmooth	0	Smooth style, which uses a more radical algorithm for more obvious effect and is suitable for show live streaming.
TXBeautyStyleNature	1	Natural style, which retains more facial details for more natural effect and is suitable for most live streaming use cases.
TXBeautyStylePitu	2	Pitu style, which is provided by YouTu Lab. Its skin smoothing effect is between the smooth style and the natural style, that is, it retains more skin details than the smooth style and has a higher skin smoothing degree than the natural style.

TXDeviceManager

Last updated：2024-06-06 15:47:57

Module: audio/video device management module

Description: manages audio/video devices such as camera, mic, and speaker.

TXDeviceManager

TXDeviceObserver

FuncList	DESC
onDeviceChanged:type:state:	The status of a local device changed (for desktop OS only)

TXDeviceManager

FuncList	DESC
isFrontCamera	Querying whether the front camera is being used
switchCamera:	Switching to the front/rear camera (for mobile OS)
isCameraZoomSupported	Querying whether the current camera supports zooming (for mobile OS)
getCameraZoomMaxRatio	Getting the maximum zoom ratio of the camera (for mobile OS)
setCameraZoomRatio:	Setting the camera zoom ratio (for mobile OS)
isAutoFocusEnabled	Querying whether automatic face detection is supported (for mobile OS)
enableCameraAutoFocus:	Enabling auto focus (for mobile OS)
setCameraFocusPosition:	Adjusting the focus (for mobile OS)

isCameraTorchSupported	Querying whether flash is supported (for mobile OS)
enableCameraTorch:	Enabling/Disabling flash, i.e., the torch mode (for mobile OS)
setAudioRoute:	Setting the audio route (for mobile OS)
setExposureCompensation:	Set the exposure parameters of the camera, ranging from - 1 to 1
getDevicesList:	Getting the device list (for desktop OS)
setCurrentDevice:deviceId:	Setting the device to use (for desktop OS)
getCurrentDevice:	Getting the device currently in use (for desktop OS)
setCurrentDeviceVolume:deviceType:	Setting the volume of the current device (for desktop OS)
getCurrentDeviceVolume:	Getting the volume of the current device (for desktop OS)
setCurrentDeviceMute:deviceType:	Muting the current device (for desktop OS)
getCurrentDeviceMute:	Querying whether the current device is muted (for desktop OS)
enableFollowingDefaultAudioDevice:enable:	Set the audio device used by SDK to follow the system default device (for desktop OS)
startCameraDeviceTest:	Starting camera testing (for desktop OS)
stopCameraDeviceTest	Ending camera testing (for desktop OS)
startMicDeviceTest:	Starting mic testing (for desktop OS)
startMicDeviceTest:playback:	Starting mic testing (for desktop OS)
stopMicDeviceTest	Ending mic testing (for desktop OS)
startSpeakerDeviceTest:	Starting speaker testing (for desktop OS)
stopSpeakerDeviceTest	Ending speaker testing (for desktop OS)
setObserver:	set onDeviceChanged callback (for Mac)
setCameraCapturerParam:	Set camera acquisition preferences
setSystemVolumeType:	Setting the system volume type (for mobile OS)

StructType

FuncList	DESC
TXCameraCaptureParam	Camera acquisition parameters
TXMediaDeviceInfo	Audio/Video device information (for desktop OS)

EnumType

EnumType	DESC
TXSystemVolumeType	System volume type
TXAudioRoute	Audio route (the route via which audio is played)
TXMediaDeviceType	Device type (for desktop OS)
TXMediaDeviceState	Device operation
TXCameraCaptureMode	Camera acquisition preferences

onDeviceChanged:type:state:

onDeviceChanged:type:state:

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

The status of a local device changed (for desktop OS only)

The SDK returns this callback when a local device (camera, mic, or speaker) is connected or disconnected.

Param	DESC
deviceId	Device ID
state	Device status. `0` : connected; `1` : disconnected; `2` : started
type	Device type

isFrontCamera

isFrontCamera

Querying whether the front camera is being used

switchCamera:

switchCamera:

- (NSInteger)switchCamera:	(BOOL)frontCamera

Switching to the front/rear camera (for mobile OS)

isCameraZoomSupported

isCameraZoomSupported

Querying whether the current camera supports zooming (for mobile OS)

getCameraZoomMaxRatio

getCameraZoomMaxRatio

Getting the maximum zoom ratio of the camera (for mobile OS)

setCameraZoomRatio:

setCameraZoomRatio:

- (NSInteger)setCameraZoomRatio:	(CGFloat)zoomRatio

Setting the camera zoom ratio (for mobile OS)

Param	DESC
zoomRatio	Value range: 1-5. 1 indicates the widest angle of view (original), and 5 the narrowest angle of view (zoomed in).The maximum value is recommended to be 5. If the value exceeds 5, the video will become blurred.

isAutoFocusEnabled

isAutoFocusEnabled

Querying whether automatic face detection is supported (for mobile OS)

enableCameraAutoFocus:

enableCameraAutoFocus:

- (NSInteger)enableCameraAutoFocus:	(BOOL)enabled

Enabling auto focus (for mobile OS)

After auto focus is enabled, the camera will automatically detect and always focus on faces.

setCameraFocusPosition:

setCameraFocusPosition:

- (NSInteger)setCameraFocusPosition:	(CGPoint)position

Adjusting the focus (for mobile OS)

This API can be used to achieve the following:

1. A user can tap on the camera preview.

2. A rectangle will appear where the user taps, indicating the spot the camera will focus on.

3. The user passes the coordinates of the spot to the SDK using this API, and the SDK will instruct the camera to focus as required.

Param	DESC
position	The spot to focus on. Pass in the coordinates of the spot you want to focus on.

Note

Before using this API, you must first disable auto focus using enableCameraAutoFocus.

Return Desc:

0: operation successful; negative number: operation failed.

isCameraTorchSupported

isCameraTorchSupported

Querying whether flash is supported (for mobile OS)

enableCameraTorch:

enableCameraTorch:

- (NSInteger)enableCameraTorch:	(BOOL)enabled

Enabling/Disabling flash, i.e., the torch mode (for mobile OS)

setAudioRoute:

setAudioRoute:

- (NSInteger)setAudioRoute:	(TXAudioRoute)route

Setting the audio route (for mobile OS)

A mobile phone has two audio playback devices: the receiver at the top and the speaker at the bottom.
If the audio route is set to the receiver, the volume is relatively low, and audio can be heard only when the phone is put near the ear. This mode has a high level of privacy and is suitable for answering calls.
If the audio route is set to the speaker, the volume is relatively high, and there is no need to put the phone near the ear. This mode enables the "hands-free" feature.

setExposureCompensation:

setExposureCompensation:

- (NSInteger)setExposureCompensation:	(CGFloat)value

Set the exposure parameters of the camera, ranging from - 1 to 1

getDevicesList:

getDevicesList:

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

Getting the device list (for desktop OS)

Param	DESC
type	Device type. Set it to the type of device you want to get. For details, please see the definition of `TXMediaDeviceType` .

Note

To ensure that the SDK can manage the lifecycle of the `ITXDeviceCollection` object, after using this API, please call the `release` method to release the resources.

Do not use `delete` to release the Collection object returned as deleting the ITXDeviceCollection* pointer will cause crash.

The valid values of `type` are `TXMediaDeviceTypeMic` , `TXMediaDeviceTypeSpeaker` , and `TXMediaDeviceTypeCamera` .

This API can be used only on macOS and Windows.

setCurrentDevice:deviceId:

setCurrentDevice:deviceId:

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

Setting the device to use (for desktop OS)

Param	DESC
deviceId	Device ID. You can get the ID of a device using the getDevicesList API.
type	Device type. For details, please see the definition of `TXMediaDeviceType` .

Return Desc:

0: operation successful; negative number: operation failed.

getCurrentDevice:

getCurrentDevice:

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

Getting the device currently in use (for desktop OS)

setCurrentDeviceVolume:deviceType:

setCurrentDeviceVolume:deviceType:

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

Setting the volume of the current device (for desktop OS)

This API is used to set the capturing volume of the mic or playback volume of the speaker, but not the volume of the camera.

Param	DESC
volume	Volume. Value range: 0-100; default: 100

getCurrentDeviceVolume:

getCurrentDeviceVolume:

- (NSInteger)getCurrentDeviceVolume:	(TXMediaDeviceType)type

Getting the volume of the current device (for desktop OS)

This API is used to get the capturing volume of the mic or playback volume of the speaker, but not the volume of the camera.

setCurrentDeviceMute:deviceType:

setCurrentDeviceMute:deviceType:

- (NSInteger)setCurrentDeviceMute:	(BOOL)mute

deviceType:	(TXMediaDeviceType)type

Muting the current device (for desktop OS)

This API is used to mute the mic or speaker, but not the camera.

getCurrentDeviceMute:

getCurrentDeviceMute:

- (BOOL)getCurrentDeviceMute:	(TXMediaDeviceType)type

Querying whether the current device is muted (for desktop OS)

This API is used to query whether the mic or speaker is muted. Camera muting is not supported.

enableFollowingDefaultAudioDevice:enable:

enableFollowingDefaultAudioDevice:enable:

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

Set the audio device used by SDK to follow the system default device (for desktop OS)

This API is used to set the microphone and speaker types. Camera following the system default device is not supported.

Param	DESC
enable	Whether to follow the system default audio device. true: following. When the default audio device of the system is changed or new audio device is plugged in, the SDK immediately switches the audio device. false：not following. When the default audio device of the system is changed or new audio device is plugged in, the SDK doesn't switch the audio device.
type	Device type. For details, please see the definition of `TXMediaDeviceType` .

startCameraDeviceTest:

startCameraDeviceTest:

- (NSInteger)startCameraDeviceTest:	(NSView *)view

Starting camera testing (for desktop OS)

Note

You can use the setCurrentDevice API to switch between cameras during testing.

stopCameraDeviceTest

stopCameraDeviceTest

Ending camera testing (for desktop OS)

startMicDeviceTest:

startMicDeviceTest:

- (NSInteger)startMicDeviceTest:	(NSInteger)interval

Starting mic testing (for desktop OS)

This API is used to test whether the mic functions properly. The mic volume detected (value range: 0-100) is returned via a callback.

Param	DESC
interval	Interval of volume callbacks

Note

When this interface is called, the sound recorded by the microphone will be played back to the speakers by default.

startMicDeviceTest:playback:

startMicDeviceTest:playback:

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

Starting mic testing (for desktop OS)

This API is used to test whether the mic functions properly. The mic volume detected (value range: 0-100) is returned via a callback.

Param	DESC
interval	Interval of volume callbacks
playback	Whether to play back the microphone sound. The user will hear his own sound when testing the microphone if `playback` is true.

stopMicDeviceTest

stopMicDeviceTest

Ending mic testing (for desktop OS)

startSpeakerDeviceTest:

startSpeakerDeviceTest:

- (NSInteger)startSpeakerDeviceTest:	(NSString *)audioFilePath

Starting speaker testing (for desktop OS)

This API is used to test whether the audio playback device functions properly by playing a specified audio file. If users can hear audio during testing, the device functions properly.

Param	DESC
filePath	Path of the audio file

stopSpeakerDeviceTest

stopSpeakerDeviceTest

Ending speaker testing (for desktop OS)

setObserver:

setObserver:

- (void)setObserver:	(nullable id<TXDeviceObserver>) observer

set onDeviceChanged callback (for Mac)

setCameraCapturerParam:

setCameraCapturerParam:

- (void)setCameraCapturerParam:	(TXCameraCaptureParam *)params

Set camera acquisition preferences

setSystemVolumeType:

setSystemVolumeType:

- (NSInteger)setSystemVolumeType:	(TXSystemVolumeType)type

Setting the system volume type (for mobile OS)

@deprecated This API is not recommended after v9.5. Please use the `startLocalAudio(quality)` API in `TRTCCloud` instead, which param `quality` is used to decide audio quality.

TXSystemVolumeType(Deprecated)

TXSystemVolumeType(Deprecated)

System volume type

Enum	Value	DESC
TXSystemVolumeTypeAuto	0	Auto
TXSystemVolumeTypeMedia	1	Media volume
TXSystemVolumeTypeVOIP	2	Call volume

TXAudioRoute

TXAudioRoute

Audio route (the route via which audio is played)

Audio route is the route (speaker or receiver) via which audio is played. It applies only to mobile devices such as mobile phones.
A mobile phone has two speakers: one at the top (receiver) and the other the bottom.
If the audio route is set to the receiver, the volume is relatively low, and audio can be heard only when the phone is put near the ear. This mode has a high level of privacy and is suitable for answering calls.
If the audio route is set to the speaker, the volume is relatively high, and there is no need to put the phone near the ear. This mode enables the "hands-free" feature.

Enum	Value	DESC
TXAudioRouteSpeakerphone	0	Speakerphone: the speaker at the bottom is used for playback (hands-free). With relatively high volume, it is used to play music out loud.
TXAudioRouteEarpiece	1	Earpiece: the receiver at the top is used for playback. With relatively low volume, it is suitable for call scenarios that require privacy.

TXMediaDeviceType

TXMediaDeviceType

Device type (for desktop OS)

This enumerated type defines three types of audio/video devices, namely camera, mic and speaker, so that you can use the same device management API to manage three types of devices.

Enum	Value	DESC
TXMediaDeviceTypeUnknown	-1	undefined device type
TXMediaDeviceTypeAudioInput	0	microphone
TXMediaDeviceTypeAudioOutput	1	speaker or earpiece
TXMediaDeviceTypeVideoCamera	2	camera

TXMediaDeviceState

TXMediaDeviceState

Device operation

This enumerated value is used to notify the status change of the local device onDeviceChanged.

Enum	Value	DESC
TXMediaDeviceStateAdd	0	The device has been plugged in
TXMediaDeviceStateRemove	1	The device has been removed
TXMediaDeviceStateActive	2	The device has been enabled
TXMediaDefaultDeviceChanged	3	system default device changed

TXCameraCaptureMode

TXCameraCaptureMode

Camera acquisition preferences

This enum is used to set camera acquisition parameters.

Enum	Value	DESC
TXCameraResolutionStrategyAuto	0	Auto adjustment of camera capture parameters. SDK selects the appropriate camera output parameters according to the actual acquisition device performance and network situation, and maintains a balance between device performance and video preview quality.
TXCameraResolutionStrategyPerformance	Not Defined	Give priority to equipment performance. SDK selects the closest camera output parameters according to the user's encoder resolution and frame rate, so as to ensure the performance of the device.
TXCameraResolutionStrategyHighQuality	Not Defined	Give priority to the quality of video preview. SDK selects higher camera output parameters to improve the quality of

		preview video. In this case, it will consume more CPU and memory to do video preprocessing.
TXCameraCaptureManual	Not Defined	Allows the user to set the width and height of the video captured by the local camera.

TXCameraCaptureParam

TXCameraCaptureParam

Camera acquisition parameters

This setting determines the quality of the local preview image.

EnumType	DESC
height	Field description: height of acquired image
mode	Field description: camera acquisition preferences,please see TXCameraCaptureMode
width	Field description: width of acquired image

TXMediaDeviceInfo

TXMediaDeviceInfo

Audio/Video device information (for desktop OS)

This structure describes key information (such as device ID and device name) of an audio/video device, so that users can choose on the UI the device to use.

EnumType	DESC
deviceId	device id (UTF-8)
deviceName	device name (UTF-8)
deviceProperties	device properties
type	device type

ErrorCode

Last updated：2024-06-06 15:47:57

Module: V2TXLiveCode @ TXLiteAVSDK

Function: Definitions of error codes and warning codes of Tencent Cloud LVB

ErrorCode

EnumType

EnumType	DESC
V2TXLiveCode	V2 Error codes and warning codes

V2TXLiveCode

V2TXLiveCode

V2 Error codes and warning codes

Enum	Value	DESC
V2TXLIVE_OK	0	No error.
V2TXLIVE_ERROR_FAILED	-1	Unclassified error.
V2TXLIVE_ERROR_INVALID_PARAMETER	-2	An invalid parameter was input during the API call.
V2TXLIVE_ERROR_REFUSED	-3	The API call was rejected.
V2TXLIVE_ERROR_NOT_SUPPORTED	-4	The API is currently not suppoted.
V2TXLIVE_ERROR_INVALID_LICENSE	-5	Failed to call the API because the license was invalid.

V2TXLIVE_ERROR_REQUEST_TIMEOUT	-6	The server request timed out.
V2TXLIVE_ERROR_SERVER_PROCESS_FAILED	-7	The server cannot process the request.
V2TXLIVE_ERROR_DISCONNECTED	-8	Disconnect.
V2TXLIVE_ERROR_NO_AVAILABLE_HEVC_DECODERS	-2304	could not find available hevc decoder.
V2TXLIVE_WARNING_NETWORK_BUSY	1101	Data upload was jammed because the upstream bandwidth was too low.
V2TXLIVE_WARNING_VIDEO_BLOCK	2105	Blocking occurred during video playback.
V2TXLIVE_WARNING_CAMERA_START_FAILED	-1301	Failed to start the camera.
V2TXLIVE_WARNING_CAMERA_OCCUPIED	-1316	The camera is being occupied.
V2TXLIVE_WARNING_CAMERA_NO_PERMISSION	-1314	The camera is not authorized. This warning usually occurs on mobile devices due to the camera permission is denied by the user.
V2TXLIVE_WARNING_MICROPHONE_START_FAILED	-1302	Failed to enable the mic.
V2TXLIVE_WARNING_MICROPHONE_OCCUPIED	-1319	The mic is being used. If a call is in progress on the mobile device, the mic cannot be enabled.
V2TXLIVE_WARNING_MICROPHONE_NO_PERMISSION	-1317	The mic is not authorized. This warning usually occurs on mobile devices due to the mic permission is denied by the user.
V2TXLIVE_WARNING_SCREEN_CAPTURE_NOT_SUPPORTED	-1309	Screen capture is not supported in curent system.
V2TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	-1308	Failed to enable the screen

		capture.
V2TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	-7001	Screen capture is interrupted by system.
V2TXLIVE_WARNING_CURRENT_ENCODE_TYPE_CHANGED	1104	The codec changed. The additional field `codec_type` in `onWarning` indicates the codec currently in use. `1` indicates H.265, and `0` indicates H.264. This field is not supported on Windows.
V2TXLIVE_WARNING_CURRENT_DECODE_TYPE_CHANGED	2008	The codec changed. The additional field `codec_type` in `onWarning` indicates the codec currently in use. `1` indicates H.265, and `0` indicates H.264. This field is not supported on Windows.

Type Definition

Last updated：2024-06-06 15:47:57

Module: V2TXLiveDef @ TXLiteAVSDK

Function: Key type definitions for Tencent Cloud LVB

Type Definition

StructType

FuncList	DESC
V2TXLiveVideoEncoderParam	Video encoding parameters
V2TXLiveVideoFrame	Video frame information
V2TXLiveAudioFrameObserverFormat	audio callback format
V2TXLivePusherStatistics	Pusher statistics
V2TXLivePlayerStatistics	Player statistics
V2TXLiveMixStream	Position of each subimage in On-Cloud MixTranscoding
V2TXLiveTranscodingConfig	Configure On-Cloud MixTranscoding
V2TXLiveLocalRecordingParams	Configure On-LocalRecording
V2TXLiveSocks5ProxyConfig	Protocol configured with socks5 proxy.
V2TXLiveLogConfig	Log configuration
V2TXLiveStreamInfo	Stream information supporting adaptive handover.

EnumType

EnumType	DESC

V2TXLiveMode	Supported protocol, RTMP is not supported on Windows or macOS.
V2TXLiveVideoResolution	Video resolution.
V2TXLiveVideoResolutionMode	Video aspect ratio mode
V2TXLiveMirrorType	Local camera mirror type.
V2TXLiveFillMode	Image fill mode
V2TXLiveRotation	Clockwise rotation of the video image
V2TXLivePixelFormat	Pixel format of video frames
V2TXLiveBufferType	Video data container format
V2TXLivePictureInPictureState	Picture-in-Picture state
V2TXLiveAudioQuality	Audio quality
V2TXLiveAudioFrameOperationMode	Audio callback data operation mode
V2TXLivePushStatus	Livestream connection status
V2TXAudioRoute	Playback mode
V2TXLiveMixInputType	Specify the type of streams to mix
V2TXLiveRecordMode	Recording audio and video mode
V2TXLiveLogLevel	Log level

V2TXLiveMode

V2TXLiveMode

Supported protocol, RTMP is not supported on Windows or macOS.

Enum	Value	DESC
V2TXLiveMode_RTMP	Not Defined	RTMP protocol.
V2TXLiveMode_RTC	Not Defined	TRTC protocol.

V2TXLiveVideoResolution

V2TXLiveVideoResolution

Video resolution.

Enum	Value	DESC
V2TXLiveVideoResolution160x160	Not Defined	Resolution: 160×160. Bitrate range: 100 Kbps to 150 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution270x270	Not Defined	Resolution: 270×270. Bitrate range: 200 Kbps to 300 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution480x480	Not Defined	Resolution: 480×480. Bitrate range: 350 Kbps to 525 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution320x240	Not Defined	Resolution: 320×240. Bitrate range: 250 Kbps to 375 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution480x360	Not Defined	Resolution: 480×360. Bitrate range: 400 Kbps to 600 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution640x480	Not Defined	Resolution: 640×480. Bitrate range: 600 Kbps to 900 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution320x180	Not Defined	Resolution: 320×180. Bitrate range: 250 Kbps to 400 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution480x270	Not Defined	Resolution: 480×270. Bitrate range: 350 Kbps to 550 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution640x360	Not Defined	Resolution: 640×360. Bitrate range: 500 Kbps to 900 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution960x540	Not Defined	Resolution: 960×540. Bitrate range: 800 Kbps to 1500 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution1280x720	Not Defined	Resolution: 1280×720. Bitrate range: 1000 Kbps to 1800 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution1920x1080	Not Defined	Resolution: 1920×1080. Bitrate range: 2500 Kbps to 3000 Kbps. Frame rate: 15 fps.

V2TXLiveVideoResolutionMode

V2TXLiveVideoResolutionMode

Video aspect ratio mode

Note

Landscape resolution: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModeLandscape = 640 × 360.

Portrait resolution: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModePortrait = 360 × 640.

Enum	Value	DESC
V2TXLiveVideoResolutionModeLandscape	0	Landscape resolution.
V2TXLiveVideoResolutionModePortrait	1	Portrait resolution.

V2TXLiveMirrorType

V2TXLiveMirrorType

Local camera mirror type.

Enum	Value	DESC
V2TXLiveMirrorTypeAuto	Not Defined	Default mirror type. Images from the front camera are mirrored, and images from the rear camera are not mirrored.
V2TXLiveMirrorTypeEnable	Not Defined	Both the front and rear cameras are switched to the mirror mode.
V2TXLiveMirrorTypeDisable	Not Defined	Both the front and rear cameras are switched to the non-mirror mode.

V2TXLiveFillMode

V2TXLiveFillMode

Image fill mode

Enum	Value	DESC
V2TXLiveFillModeFill	Not Defined	The entire screen is covered by the image, without black edges. If the aspect ratio of the image is different from that of the screen, part of the image will be cropped.

V2TXLiveFillModeFit	Not Defined	The image adapts to the screen and is not cropped. If the aspect ratio of the image is different from that of the screen, black edges will appear.
V2TXLiveFillModeScaleFill	Not Defined	The screen is entirely covered by the image. The image will be stretched if screen and image have different aspect ratios.

V2TXLiveRotation

V2TXLiveRotation

Clockwise rotation of the video image

Enum	Value	DESC
V2TXLiveRotation0	Not Defined	No rotation.
V2TXLiveRotation90	Not Defined	Rotate 90 degrees clockwise.
V2TXLiveRotation180	Not Defined	Rotate 180 degrees clockwise.
V2TXLiveRotation270	Not Defined	Rotate 270 degrees clockwise.

V2TXLivePixelFormat

V2TXLivePixelFormat

Pixel format of video frames

Enum	Value	DESC
V2TXLivePixelFormatUnknown	Not Defined	Unknown.
V2TXLivePixelFormatI420	Not Defined	YUV420P I420.
V2TXLivePixelFormatNV12	Not	YUV420SP NV12.

	Defined	
V2TXLivePixelFormatBGRA32	Not Defined	BGRA8888.
V2TXLivePixelFormatTexture2D	Not Defined	Texture2D.

V2TXLiveBufferType

V2TXLiveBufferType

Video data container format

Note

In the custom capture and rendering features, you need to use the following enumerated values to specify the format for containing video data.

PixelBuffer: this is most efficient when used directly. The iOS system provides various APIs to obtain or process PixelBuffer.

NSData: when this is applied to custom rendering, PixelBuffer is copied once to NSData. When it is applied to custom capture, NSData is copied once to PixelBuffer. Therefore, the performance is affected to some extent.

Enum	Value	DESC
V2TXLiveBufferTypeUnknown	Not Defined	Unknown.
V2TXLiveBufferTypePixelBuffer	Not Defined	This is most efficient when used directly. The iOS system provides various APIs to obtain or process PixelBuffer.
V2TXLiveBufferTypeNSData	Not Defined	The performance is affected to some extent. As the SDK internally processes PixelBuffer directly, type switching between NSData and PixelBuffer results in memory copy overhead.
V2TXLiveBufferTypeTexture	Not Defined	Texture.

V2TXLivePictureInPictureState

V2TXLivePictureInPictureState

Picture-in-Picture state

Enum	Value	DESC
V2TXLivePictureInPictureStateUndefined	Not Defined	Undefined.
V2TXLivePictureInPictureStateOccurError	Not Defined	An error occurred in Picture-in-Picture mode.
V2TXLivePictureInPictureStateWillStart	Not Defined	Picture-in-Picture mode will start.
V2TXLivePictureInPictureStateDidStart	Not Defined	Picture-in-Picture mode did start.
V2TXLivePictureInPictureStateWillStop	Not Defined	Picture-in-Picture mode will stop.
V2TXLivePictureInPictureStateDidStop	Not Defined	Picture-in-Picture mode did stop.

V2TXLiveAudioQuality

V2TXLiveAudioQuality

Audio quality

Enum	Value	DESC
V2TXLiveAudioQualitySpeech	Not Defined	Audio: 16k sample rate, mono-channel, 16 Kbps audio raw bitrate. This quality is suitable for scenarios that mainly involve voice calls, such as online meetings and voice calls.
V2TXLiveAudioQualityDefault	Not Defined	General: 48k sample rate, mono-channel, 50 Kbps audio raw bitrate. This quality is the default audio quality of the SDK. We recommend that you choose this option unless you have special requirements.
V2TXLiveAudioQualityMusic	Not Defined	Music: 48k sample rate, dual-channel + full-band, 128 Kbps audio raw bitrate. This quality is suitable for scenarios that require Hi-Fi music transmission, such as karaoke and music livestreams.

V2TXLiveAudioFrameOperationMode

V2TXLiveAudioFrameOperationMode

Audio callback data operation mode

SDK provides two modes of operation for audio callback data.

Read-only mode (ReadOnly): Get audio data only from the callback.

ReadWrite mode (ReadWrite): You can get and modify the audio data of the callback.

Enum	Value	DESC
V2TXLiveAudioFrameOperationModeReadWrite	0	Read-write mode: You can get and modify the audio data of the callback, the default mode.
V2TXLiveAudioFrameOperationModeReadOnly	1	Read-only mode: Get audio data from callback only.

V2TXLivePushStatus

V2TXLivePushStatus

Livestream connection status

Enum	Value	DESC
V2TXLivePushStatusDisconnected	Not Defined	Disconnected from the server.
V2TXLivePushStatusConnecting	Not Defined	Connecting to the server.
V2TXLivePushStatusConnectSuccess	Not Defined	Connected to the server successfully.
V2TXLivePushStatusReconnecting	Not Defined	Reconnecting to the server.

V2TXAudioRoute

V2TXAudioRoute

Playback mode

Enum	Value	DESC
V2TXAudioModeSpeakerphone	Not Defined	Speaker.
V2TXAudioModeEarpiece	Not Defined	Earpiece.

V2TXLiveMixInputType

V2TXLiveMixInputType

Specify the type of streams to mix

Enum	Value	DESC
V2TXLiveMixInputTypeAudioVideo	Not Defined	Audio and video.
V2TXLiveMixInputTypePureVideo	Not Defined	Video only.
V2TXLiveMixInputTypePureAudio	Not Defined	Audio only.

V2TXLiveRecordMode

V2TXLiveRecordMode

Recording audio and video mode

Enum	Value	DESC
V2TXLiveRecordModeBoth	Not Defined	Both mode: Recording audio and video

V2TXLiveLogLevel

V2TXLiveLogLevel

Log level

Enum	Value	DESC
V2TXLiveLogLevelAll	0	Output all levels of log.
V2TXLiveLogLevelDebug	1	Output DEBUG, INFO, WARNING, ERROR and FATAL level log.
V2TXLiveLogLevelInfo	2	Output INFO, WARNING, ERROR and FATAL level log.
V2TXLiveLogLevelWarning	3	Output WARNING, ERROR and FATAL level log.
V2TXLiveLogLevelError	4	Output ERROR and FATAL level log.
V2TXLiveLogLevelFatal	5	Only output FATAL level log.
V2TXLiveLogLevelNULL	6	Does not output any sdk log.

V2TXLiveVideoEncoderParam

V2TXLiveVideoEncoderParam

Video encoding parameters

These settings determine the quality of image viewed by remote users.

EnumType	DESC
minVideoBitrate	`Field description:` minimum video bitrate. The SDK will reduce the bitrate to as low as the value specified by `minVideoBitrate` to ensure the smoothness only if the network conditions are poor. `Recommended value:` you can set the `videoBitrate` and `minVideoBitrate` parameters at the same time to restrict the SDK's adjustment range of the video bitrate: If you set `videoBitrate` and `minVideoBitrate` to the same value, it is equivalent to disabling the adaptive adjustment capability of the SDK for the video bitrate.
videoBitrate	`Field description:` target video bitrate. The SDK encodes streams at the target video bitrate and will actively reduce the bitrate only in weak network environments. `Recommended value:` please see the optimal bitrate for each specification in `V2TXLiveVideoResolution`. You can also slightly increase the optimal bitrate.

	For example, `V2TXLiveVideoResolution1280x720` corresponds to the target bitrate of 1,200 Kbps. You can also set the bitrate to 1,500 Kbps for higher definition. **Note** you can set the `videoBitrate` and `minVideoBitrate` parameters at the same time to restrict the SDK's adjustment range of the video bitrate: If you set `videoBitrate` and `minVideoBitrate` to the same value, it is equivalent to disabling the adaptive adjustment capability of the SDK for the video bitrate.
videoFps	`Field description:` video capturing frame rate. `Recommended value:` 15 or 20 fps. If the frame rate is lower than 5 fps, there will be obvious lagging; if lower than 10 fps but higher than 5 fps, there will be slight lagging; if higher than 20 fps, the bandwidth will be wasted (the frame rate of movies is generally 24 fps).
videoResolution	`Field description:` video resolution. `Recommended value:` For desktop platforms (Windows and macOS), we recommend you select a resolution of 640x360 or above and select `Landscape` (landscape resolution) for `videoResolutionMode`. **Note** to use a portrait resolution, please specify `videoResolutionMode` as `Portrait`; for example, when used together with `Portrait`, 640x360 represents 360x640.
videoResolutionMode	`Field description:` resolution mode (landscape/portrait). `Recommended value:` for desktop platforms (Windows and macOS), `Landscape` is recommended. **Note** to use a portrait resolution, please specify `videoResolutionMode` as `Portrait`; for example, when used together with `Portrait`, 640x360 represents 360x640.

V2TXLiveVideoFrame

V2TXLiveVideoFrame

Video frame information

Note

Used during custom capture and rendering. During custom capture, you need to use V2TXLiveVideoFrame to contain the video frame to be sent. During custom rendering, the video frame contained by V2TXLiveVideoFrame will be returned.

EnumType	DESC
bufferType	`Field description:` Video data container format. `Recommended value:` V2TXLiveBufferTypePixelBuffer.
data	`Field description:` Video data when bufferType is V2TXLiveBufferTypeNSData.
height	`Field description:` Video height.
pixelBuffer	`Field description:` Video data when bufferType is V2TXLiveBufferTypePixelBuffer.
pixelFormat	`Field description:` Video pixel format. `Recommended value:` V2TXLivePixelFormatNV12.
rotation	`Field description:` Clockwise rotation angle of video frames.
textureId	`Field description:` Texture ID.
width	`Field description:` Video width.

V2TXLiveAudioFrameObserverFormat

V2TXLiveAudioFrameObserverFormat

audio callback format

EnumType	DESC
channel	`Field description:` number of sound channels. `Recommended value:` default value: 1, which means mono channel. Valid values: 1: mono channel; 2: dual channel.
mode	`Field description:` audio callback data operation mode. `Recommended value:` V2TXLiveAudioFrameOperationModeReadOnly, get audio data from callback only. The modes that can be set are V2TXLiveAudioFrameOperationModeReadOnly, V2TXLiveAudioFrameOperationModeReadWrite.
sampleRate	`Field description:` sample rate.

	`Recommended value:` default value: 48000 Hz. Valid values: 16000, 32000, 44100, 48000.
samplesPerCall	`Field description:` number of sample points. `Recommended value:` the value must be an integer multiple of sampleRate/100.

V2TXLivePusherStatistics

V2TXLivePusherStatistics

Pusher statistics

EnumType	DESC
appCpu	`Field description:` CPU utilization of the current app (%).
audioBitrate	`Field description:` Audio bitrate (Kbps).
fps	`Field description:` Frame rate (fps).
height	`Field description:` Video height.
netSpeed	`Field description:` upload speed (Kbps).
rtt	`Field description:` Round-trip delay (ms) from the SDK to cloud.
systemCpu	`Field description:` CPU utilization of the current system (%).
videoBitrate	`Field description:` Video bitrate (Kbps).
width	`Field description:` Video width.

V2TXLivePlayerStatistics

V2TXLivePlayerStatistics

Player statistics

EnumType	DESC
appCpu	`Field description:` CPU utilization of the current app (%).
audioBitrate	

	`Field description:` Audio bitrate (Kbps).
audioBlockRate	`Field description` : Audio playback lag rate (%). Audio playback lag rate (audioBlockRate) = cumulative audio playback lag duration (audioTotalBlockTime)/audio playback interval duration (2000ms).
audioPacketLoss	`Field description` : Total packet loss rate (%) of the audio/video stream. Note: Only playback address prefixed with [trtc://] or [webrtc://] are supported.
audioTotalBlockTime	`Field description` : Cumulative audio playback lag duration (ms). The duration is the block duration within 2s.
fps	`Field description:` Frame rate (fps).
height	`Field description:` Video height.
jitterBufferDelay	`Field description` : Playback delay (ms).
netSpeed	`Field description:` donwload speed (Kbps).
rtt	`Field description:` Round-trip delay (ms) from the SDK to cloud. Note: Only playback address prefixed with [trtc://] or [webrtc://] are supported.
systemCpu	`Field description:` CPU utilization of the current system (%).
videoBitrate	`Field description:` Video bitrate (Kbps).
videoBlockRate	`Field description` : Video playback lag rate (%). Video playback lag rate (videoBlockRate) = cumulative video playback lag duration (videoTotalBlockTime)/video playback interval duration (2000ms).
videoPacketLoss	`Field description` : Total packet loss rate (%) of the audio/video stream. Note: Only playback address prefixed with [trtc://] or [webrtc://] are supported.
videoTotalBlockTime	`Field description` : Cumulative video playback lag duration (ms). The duration is the block duration within 2s.
width	`Field description:` Video width.

V2TXLiveMixStream

V2TXLiveMixStream

Position of each subimage in On-Cloud MixTranscoding

EnumType	DESC
height	`Field description:` height (absolute pixels) of the image layer.
inputType	`Field description:` input type of the live stream.
streamId	`Field description:` push `streamId` of users whose streams are mixed. `nil` indicates the current push `streamId`.
userId	`Field description:` `userId` of users whose streams are mixed.
width	`Field description:` width (absolute pixels) of the image layer.
x	`Field description:` x-axis (absolute pixels) of the image layer.
y	`Field description:` y-axis (absolute pixels) of the image layer.
zOrder	`Field description:` layer number (1-15), which must be unique.

V2TXLiveTranscodingConfig

V2TXLiveTranscodingConfig

Configure On-Cloud MixTranscoding

EnumType	DESC
audioBitrate	`Field description:` audio bitrate of the transcoded stream. `Value range:` [32,192]; default value: 64 (Kbps).
audioChannels	`Field description:` number of sound channels of the transcoded stream. `Valid values:` 1 (default), 2.
audioSampleRate	`Field description:` audio sample rate of the transcoded stream. `Valid values:` 12000 Hz, 16000 Hz, 22050 Hz, 24000 Hz, 32000 Hz, 44100 Hz, 48000 Hz (default).
backgroundColor	`Field description:` background color of the mixed video image. The default color is black, and the value is a hex number. For example: "0x61B9F1" represents the RGB color (97,158,241). **Default value:** 0x000000 (black)
backgroundImage	`Field description:` background image of the mixed video.

	Default value: `nil` , which means that no background image is set. **Note** you need to first upload the image in **Application Management** > **Function Configuration** > **Material Management** in the console. You will get an image ID for the image uploaded, which you need to convert to a string and use it as the value of `backgroundImage` . For example, if the image ID is 63, you should set `backgroundImage` to `63` .
mixStreams	`Field description:` position of each channel of subimage.
outputStreamId	`Field description:` ID of the live stream pushed to CDN. If you do not set this parameter, the SDK will execute the default logic, that is, it will mix multiple streams in the room into the video stream of the API caller, i.e., A + B => A. If you set this parameter, the SDK will mix multiple streams in the room into the live stream whose ID you have specified, i.e., A + B => C. `Default value` : `nil` , which indicates that multiple streams in the room are mixed into the video stream of the API caller.
videoBitrate	`Field description:` bitrate (Kbps) for the resolution of the transcoded video. `Recommended value:` if you set it to 0, the backend will calculate a bitrate based on `videoWidth` and `videoHeight` . You can also refer to the remarks for the enumerated value `V2TXLiveVideoResolution` .
videoFramerate	`Field description:` frame rate (fps) for the resolution of the transcoded video. `Value range:` (0,30]; default: 15.
videoGOP	`Field description:` keyframe interval (GOP) for the resolution of the transcoded video. `Value range:` [1,8]; default value: 2 (sec).
videoHeight	`Field description:` height of transcoded video. `Recommended value:` 640 px. If audio-only streams are mixed, the mixing result will carry a video stream that shows a canvas background. To avoid this, set both the width and height to 0 px.
videoWidth	`Field description:` width of transcoded video. `Recommended value:` 360 px. If audio-only streams are mixed, the mixing result will carry a video stream that shows a canvas background. To avoid this, set both the width and height to 0 px.

V2TXLiveLocalRecordingParams

V2TXLiveLocalRecordingParams

Configure On-LocalRecording

EnumType	DESC
filePath	`Field description:` The path of the recorded file (required), please ensure that the path has read and write permissions and is legal, otherwise the recorded file cannot be generated. `Recommended value:` "yourpath/record/test.mp4". The path needs to be accurate to the file name and format suffix. The format suffix is used to determine the recorded file format. The currently supported format is only MP4.
interval	`Field description:` interval Recording information update frequency (optional), in milliseconds, valid range: 1000-10000. `Default value` : `-1` , which means no callback.
recordMode	`Field description:` Media recording mode. `Default value` : `V2TXLiveRecordModeBoth` , which means recording audio and video at the same time.

V2TXLiveSocks5ProxyConfig

V2TXLiveSocks5ProxyConfig

Protocol configured with socks5 proxy.

EnumType	DESC
supportHttps	`Field description:` Indicates whether HTTPS is supported. `Recommended value:` Default value: true.
supportTcp	`Field description:` Indicates whether TCP is supported. `Recommended value:` Default value: true.
supportUdp	`Field description:` Indicates whether UDP is supported. `Recommended value:` Default value: true.

V2TXLiveLogConfig

V2TXLiveLogConfig

Log configuration

EnumType	DESC
enableConsole	`Field description:` Whether to allow the SDK to print Log on the console of the editor (XCoder, Android Studio, Visual Studio, etc.). `Recommended value:` Default value: NO.
enableLogFile	`Field description:` Whether to enable local log file. `Special Instructions:` If not for special needs, please do not close the local log file, otherwise the Tencent Cloud technical team will not be able to track and locate problems when they occur. `Recommended value:` Default value: YES.
enableObserver	`Field description:` Whether to receive the log information to be printed through V2TXLivePremierObserver. `Special Instructions:` If you want to implement Log writing by yourself, you can turn on this switch, Log information will be called back to you V2TXLivePremierObserver#onLog. `Recommended value:` Default value: NO.
logLevel	`Field description:` Set Log level. `Recommended value:` Default value: V2TXLiveLogLevelAll.
logPath	`Field description:` Set the storage directory of the local log, default Log storage location: iOS & Mac: sandbox Documents/log.

V2TXLiveStreamInfo

V2TXLiveStreamInfo

Stream information supporting adaptive handover.

EnumType	DESC
bitrate	`Field description:` Bitrate in bps, default value: 0, means unknown.
framerate	`Field description:` Framerate, default value: 0, means unknown.
height	`Field description:` Video height, default value: 0, means unknown.
url	`Field description:` Stream url.

width	`Field description:` Video width, default value: 0, means unknown.

Android
API Overview

Last updated：2024-06-06 15:47:57

API OVERVIEW

LivePusher Interface

FuncList	DESC
release	Release `V2TXLivePusher` resources
setObserver	Sets the pusher callback
setRenderView	Sets the local camera preview
setRenderMirror	Sets the view mirror of the local camera
setEncoderMirror	Sets the video encoder mirror
setRenderRotation	Sets the rotation angle of the view
setRenderFillMode	Sets the fill mode of the local video image
startCamera	Enables the local camera
stopCamera	Disables the local camera
startMicrophone	Enables the local microphone
stopMicrophone	Disables the microphone
startVirtualCamera	Enables the image streaming
stopVirtualCamera	Disables the image streaming
startScreenCapture	Enables video screen capture
stopScreenCapture	Disables video capture
pauseAudio	Pause the audio stream of the pusher
resumeAudio	Resume the audio stream of the pusher

pauseVideo	Pause the video stream of the pusher
resumeVideo	Resume the video stream of the pusher
startPush	Starts pushing the audio and video data
stopPush	Stops pushing the audio and video data
isPushing	Indicates whether the pusher is currently pushing streams
setAudioQuality	Sets the audio quality for pushing
setVideoQuality	Set the video encoding parameters for pushing
getBeautyManager	Obtains the beauty manager
getAudioEffectManager	Obtains the audio effect manager
getDeviceManager	Obtains the video device manager
snapshot	Captures the local view in the pushing process
setWatermark	Sets the pusher watermark image. By default, the watermark is disabled
enableVolumeEvaluation	Enables volume update
enableCustomVideoProcess	Enables or disables custom video processing
enableCustomVideoCapture	Enables or disables custom video capture
enableCustomAudioCapture	Turn on/off custom audio capture
sendCustomVideoFrame	Sends the collected video data to the SDK in the custom video capture mode
sendCustomAudioFrame	In the custom audio collection mode, send the collected audio data to the SDK
enableAudioProcessObserver	Enables/Disables audio process callback
sendSeiMessage	Use SEI channel to send custom message
startSystemAudioLoopback	Enable system audio capturing
showDebugView	Indicates whether the debug view of the pusher video status information is displayed
setProperty	Calls the advanced API of V2TXLivePusher

setMixTranscodingConfig	Sets On-Cloud MixTranscoding parameters
startLocalRecording	Start recording audio and video stream
stopLocalRecording	Stop recording audio and video stream
enableVoiceActivityDetection	Enable voice activity detection

Live pusher Event Callback

FuncList	DESC
onError	Live pusher error notification, which is called back when the pusher encounters an error
onWarning	Live pusher warning notification
onCaptureFirstAudioFrame	Callback notification indicating that collection of the first audio frame is complete
onCaptureFirstVideoFrame	Callback notification indicating that collection of the first video frame is complete
onMicrophoneVolumeUpdate	Microphone-collected volume callback
onPushStatusUpdate	Callback notification of the pusher connection status
onStatisticsUpdate	Live pusher statistics callback
onSnapshotComplete	Screenshot callback
onGLContextCreated	Callback of created the OpenGL context in the SDK
onProcessAudioFrame	Audio data captured by the local mic, pre-processed by the audio module, effect-processed and BGM-mixed
onProcessVideoFrame	Custom video processing callback
onGLContextDestroyed	Callback of destroying the OpenGL context in the SDK
onSetMixTranscodingConfig	Callback of setting On-Cloud MixTranscoding parameters, which corresponds to the setMixTranscodingConfig API
onScreenCaptureStarted	The SDK returns this callback when you call startScreenCapture and other APIs to start screen sharing.

onScreenCaptureStopped	The SDK returns this callback when you call stopScreenCapture to stop screen sharing
onLocalRecordBegin	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete	The SDK returns this callback when you call stopLocalRecording to start local recording.
onVoiceActivityDetectionUpdate	After calling enableVoiceActivityDetection to turn on voice activity detection, you will receive this callback notification when the anchor starts or stops speaking.

V2TXLivePlayer Interface

FuncList	DESC
setObserver	Sets the player callback
setRenderView	Sets the rendering view of the player. This control is responsible for presenting the video content
setRenderRotation	Sets the rotation angle of the player view
setRenderFillMode	Sets the fill mode of the view
startLivePlay	Starts playing the audio and video streams
stopPlay	Stops playing the audio and video streams
isPlaying	Indicates whether the player is playing the audio and video streams
pauseAudio	Pauses the audio stream of the player
resumeAudio	Resumes the audio stream of the player
pauseVideo	Pauses the video stream of the player
resumeVideo	Resumes the video stream of the player
setPlayoutVolume	Sets the volume
setCacheParams	Set the minimum time and maximum time (unit: s) for auto adjustment of the

	player cache
switchStream	Seamlessly switch live stream urls, supporting FLV and LEB protocols
getStreamList	Get Stream Info List
enableVolumeEvaluation	Enables playback volume update
snapshot	Captures the video view in the playback process
enableObserveVideoFrame	Turn on/off the monitoring callback of the video frame
enableObserveAudioFrame	Turn on/off the monitoring callback of the audio frame
enableReceiveSeiMessage	Enables the receiving of SEI messages
showDebugView	Indicates whether the debug view of the player video status information is displayed
setProperty	Calls the advanced API of V2TXLivePlayer
startLocalRecording	Start recording audio and video stream
stopLocalRecording	Stop recording audio and video stream

Live Player Event Callback

FuncList	DESC
onError	live player error notification, which is called back when the player encounters an error
onWarning	live player warning notification
onVideoResolutionChanged	live player resolution change notification
onConnected	live player has successfully connected to the server notification
onVideoPlaying	Video playback event
onAudioPlaying	Audio playback event
onVideoLoading	Video loading event
onAudioLoading	Audio loading event
onPlayoutVolumeUpdate	Player playback volume callback

onStatisticsUpdate	Live player statistics callback
onSnapshotComplete	Screenshot callback
onRenderVideoFrame	Custom video rendering callback
onPlayoutAudioFrame	Audio Data callback
onReceiveSeiMessage	Callback of receiving an SEI message. The sender calls `sendSeiMessage` in V2TXLivePusher to send an SEI
onStreamSwitched	Resolution stream switch callback
onLocalRecordBegin	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete	The SDK returns this callback when you call stopLocalRecording to start local recording.

V2TXLive High-level interface

FuncList	DESC
getSDKVersionStr	Get the SDK version number
setObserver	Set V2TXLivePremier callback interface
setLogConfig	Set Log configuration information
setEnvironment	Set up SDK access environment
setLicence	Set SDK authorization license
setSocks5Proxy	Set SDK socks5 proxy config
enableAudioCaptureObserver	Enables/Disables audio capture callback
enableAudioPlayoutObserver	Enables/Disables audio playout callback
enableVoiceEarMonitorObserver	Enables/Disables in-ear monitoring callback
setUserId	Set user id

callExperimentalAPI	Call experimental APIs

V2TXLive Advanced callback interface

FuncList	DESC
onLog	Custom Log output callback interface
onLicenceLoaded	setLicence result callback interface
onCaptureAudioFrame	Raw audio data captured locally
onPlayoutAudioFrame	Data mixed from each channel before being submitted to the system for playback
onVoiceEarMonitorAudioFrame	In-ear monitoring data

Background music preload event callback

FuncList	DESC
onLoadProgress	Background music preload progress
onLoadError	Background music preload error

Callback of playing background music

FuncList	DESC
onStart	Background music started.
onPlayProgress	Playback progress of background music
onComplete	Background music ended

Voice effect APIs

FuncList	DESC

enableVoiceEarMonitor	Enabling in-ear monitoring
setVoiceEarMonitorVolume	Setting in-ear monitoring volume
setVoiceReverbType	Setting voice reverb effects
setVoiceChangerType	Setting voice changing effects
setVoiceCaptureVolume	Setting speech volume
setVoicePitch	Setting speech pitch

Background music APIs

FuncList	DESC
setMusicObserver	Setting the background music callback
startPlayMusic	Starting background music
stopPlayMusic	Stopping background music
pausePlayMusic	Pausing background music
resumePlayMusic	Resuming background music
setAllMusicVolume	Setting the local and remote playback volume of background music
setMusicPublishVolume	Setting the remote playback volume of a specific music track
setMusicPlayoutVolume	Setting the local playback volume of a specific music track
setMusicPitch	Adjusting the pitch of background music
setMusicSpeedRate	Changing the speed of background music
getMusicCurrentPosInMS	Getting the playback progress (ms) of background music
getMusicDurationInMS	Getting the total length (ms) of background music
seekMusicToPosInMS	Setting the playback progress (ms) of background music
setMusicScratchSpeedRate	Adjust the speed change effect of the scratch disc
setPreloadObserver	Setting music preload callback
preloadMusic	Preload background music

getMusicTrackCount	Get the number of tracks of background music
setMusicTrack	Specify the playback track of background music

beauty interface

FuncList	DESC
setBeautyStyle	Sets the beauty (skin smoothing) filter algorithm.
setBeautyLevel	Sets the strength of the beauty filter.
setWhitenessLevel	Sets the strength of the brightening filter.
enableSharpnessEnhancement	Enables clarity enhancement.
setRuddyLevel	Sets the strength of the rosy skin filter.
setFilter	Sets color filter.
setFilterStrength	Sets the strength of color filter.
setGreenScreenFile	Sets green screen video
setEyeScaleLevel	Sets the strength of the eye enlarging filter.
setFaceSlimLevel	Sets the strength of the face slimming filter.
setFaceVLevel	Sets the strength of the chin slimming filter.
setChinLevel	Sets the strength of the chin lengthening/shortening filter.
setFaceShortLevel	Sets the strength of the face shortening filter.
setFaceNarrowLevel	Sets the strength of the face narrowing filter.
setNoseSlimLevel	Sets the strength of the nose slimming filter.
setEyeLightenLevel	Sets the strength of the eye brightening filter.
setToothWhitenLevel	Sets the strength of the teeth whitening filter.
setWrinkleRemoveLevel	Sets the strength of the wrinkle removal filter.
setPounchRemoveLevel	Sets the strength of the eye bag removal filter.

setSmileLinesRemoveLevel	Sets the strength of the smile line removal filter.
setForeheadLevel	Sets the strength of the hairline adjustment filter.
setEyeDistanceLevel	Sets the strength of the eye distance adjustment filter.
setEyeAngleLevel	Sets the strength of the eye corner adjustment filter.
setMouthShapeLevel	Sets the strength of the mouth shape adjustment filter.
setNoseWingLevel	Sets the strength of the nose wing narrowing filter.
setNosePositionLevel	Sets the strength of the nose position adjustment filter.
setLipsThicknessLevel	Sets the strength of the lip thickness adjustment filter.
setFaceBeautyLevel	Sets the strength of the face shape adjustment filter.
setMotionTmpl	Selects the AI animated effect pendant.
setMotionMute	Sets whether to mute during animated effect playback.

Device APIs

FuncList	DESC
isFrontCamera	Querying whether the front camera is being used
switchCamera	Switching to the front/rear camera (for mobile OS)
getCameraZoomMaxRatio	Getting the maximum zoom ratio of the camera (for mobile OS)
setCameraZoomRatio	Setting the camera zoom ratio (for mobile OS)
isAutoFocusEnabled	Querying whether automatic face detection is supported (for mobile OS)
enableCameraAutoFocus	Enabling auto focus (for mobile OS)
setCameraFocusPosition	Adjusting the focus (for mobile OS)
enableCameraTorch	Enabling/Disabling flash, i.e., the torch mode (for mobile OS)
setAudioRoute	Setting the audio route (for mobile OS)
setExposureCompensation	Set the exposure parameters of the camera, ranging from - 1 to 1
setCameraCapturerParam	Set camera acquisition preferences

Disused APIs

FuncList	DESC
setSystemVolumeType	Setting the system volume type (for mobile OS)

V2TXLivePusher

Last updated：2024-06-06 15:47:57

Module: V2TXLivePusher @ TXLiteAVSDK

Function: Tencent Cloud live pusher

Function

Tencent Cloud Live Pusher

Introduce

It is mainly responsible for encoding the local audio and video images and pushing them to the specified streaming address, supporting any streaming server.
Flowmakers include the following capabilities:
Customized video capture, allowing you to customize your own audio and video data sources according to project needs.
Beautification, filters, stickers, including multiple sets of beautification and microdermabrasion algorithms (natural & smooth) and a variety of color space filters (support custom filters).
Qos flow control technology, with uplink network adaptive capability, can adjust the amount of audio and video data in real time according to the specific conditions of the host network.
Face shape adjustment, animation pendants, support face shape fine-tuning and animation pendant effects based on Youtu AI face recognition technology such as big eyes, thin face, nose augmentation, etc. You only need to purchase Youtu License to easily achieve rich live broadcast effects.

V2TXLivePusher

V2TXLivePusher

FuncList	DESC

release	Release `V2TXLivePusher` resources
setObserver	Sets the pusher callback
setRenderView	Sets the local camera preview
setRenderView	Sets the local camera preview
setRenderView	Sets the local camera preview
setRenderMirror	Sets the view mirror of the local camera
setEncoderMirror	Sets the video encoder mirror
setRenderRotation	Sets the rotation angle of the view
setRenderFillMode	Sets the fill mode of the local video image
startCamera	Enables the local camera
stopCamera	Disables the local camera
startMicrophone	Enables the local microphone
stopMicrophone	Disables the microphone
startVirtualCamera	Enables the image streaming
stopVirtualCamera	Disables the image streaming
startScreenCapture	Enables video screen capture
stopScreenCapture	Disables video capture
pauseAudio	Pause the audio stream of the pusher
resumeAudio	Resume the audio stream of the pusher
pauseVideo	Pause the video stream of the pusher
resumeVideo	Resume the video stream of the pusher
startPush	Starts pushing the audio and video data
stopPush	Stops pushing the audio and video data
isPushing	Indicates whether the pusher is currently pushing streams
setAudioQuality	Sets the audio quality for pushing

setVideoQuality	Set the video encoding parameters for pushing
getBeautyManager	Obtains the beauty manager
getAudioEffectManager	Obtains the audio effect manager
getDeviceManager	Obtains the video device manager
snapshot	Captures the local view in the pushing process
setWatermark	Sets the pusher watermark image. By default, the watermark is disabled
enableVolumeEvaluation	Enables volume update
enableCustomVideoProcess	Enables or disables custom video processing
enableCustomVideoCapture	Enables or disables custom video capture
enableCustomAudioCapture	Turn on/off custom audio capture
sendCustomVideoFrame	Sends the collected video data to the SDK in the custom video capture mode
sendCustomAudioFrame	In the custom audio collection mode, send the collected audio data to the SDK
enableAudioProcessObserver	Enables/Disables audio process callback
sendSeiMessage	Use SEI channel to send custom message
startSystemAudioLoopback	Enable system audio capturing
showDebugView	Indicates whether the debug view of the pusher video status information is displayed
setProperty	Calls the advanced API of V2TXLivePusher
setMixTranscodingConfig	Sets On-Cloud MixTranscoding parameters
startLocalRecording	Start recording audio and video stream
stopLocalRecording	Stop recording audio and video stream
enableVoiceActivityDetection	Enable voice activity detection

release

release

Release `V2TXLivePusher` resources

setObserver

setObserver

void setObserver	(V2TXLivePusherObserver observer)

Sets the pusher callback

By setting the callback, you can listen to some callback events of V2TXLivePusher,
including the pusher status, volume callback, statistics, warnings, and error messages.

Param	DESC
observer	Callback target of the pusher. For more information, see V2TXLivePusherObserver.

setRenderView

setRenderView

int setRenderView	(TXCloudVideoView view)

Sets the local camera preview

Images collected by the local camera will be eventually displayed on the view that is passed in after it is overlaid by multiple effects, such as beauty filters, facial feature adjustments, and filters.

Param	DESC
view	Local camera preview.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderView

setRenderView

int setRenderView	(TextureView view)

Sets the local camera preview

Images collected by the local camera will be eventually displayed on the view that is passed in after it is overlaid by multiple effects, such as beauty filters, facial feature adjustments, and filters.

Param	DESC
view	Local camera preview.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderView

setRenderView

int setRenderView	(SurfaceView view)

Sets the local camera preview

Images collected by the local camera will be eventually displayed on the view that is passed in after it is overlaid by multiple effects, such as beauty filters, facial feature adjustments, and filters.

Param	DESC
view	Local camera preview.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderMirror

setRenderMirror

int setRenderMirror	(V2TXLiveMirrorType mirrorType)

Sets the view mirror of the local camera

Local cameras are divided into the front camera and the rear camera. By default, images from the front camera are mirrored, and images from the rear camera are not mirrored. Here, you can modify the default mirror type of the front or rear camera.

Param	DESC
mirrorType	Mirror type of the camera V2TXLiveMirrorType. V2TXLiveMirrorTypeAuto `Default` : default mirror type. In this case, images from the front camera are mirrored, and images from the rear camera are not mirrored. V2TXLiveMirrorTypeEnable: both the front camera and rear camera are switched to mirror mode. V2TXLiveMirrorTypeDisable: both the front camera and rear camera are switched to non-mirror mode.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

setEncoderMirror

setEncoderMirror

int setEncoderMirror	(boolean mirror)

Sets the video encoder mirror

Param	DESC
mirror	Specifies whether the mirrored images are viewed. false `Default` : non-mirrored images are viewed on the player side. true: mirrored images are viewed on the player side.

Note

The encoder mirror only influences video effects on the audience side.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderRotation

setRenderRotation

int setRenderRotation	(V2TXLiveRotation rotation)

Sets the rotation angle of the view

Param	DESC
rotation	Rotation angle of the view V2TXLiveRotation. V2TXLiveRotation0 `Default` : 0 degrees, which means the view is not rotated. V2TXLiveRotation90: rotate 90 degrees clockwise. V2TXLiveRotation180: rotate 180 degrees clockwise. V2TXLiveRotation270: rotate 270 degrees clockwise.

Note

Only the view is rotated, and images that are pushed are not affected.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

setRenderFillMode

setRenderFillMode

int setRenderFillMode	(V2TXLiveFillMode mode)

Sets the fill mode of the local video image

Param	DESC
mode	Fill mode of the view V2TXLiveFillMode. V2TXLiveFillModeFill: **Default**: fill the screen with the image without leaving any black edges. If the aspect ratio of the view is different from that of the screen, part of the view will be cropped. V2TXLiveFillModeFit make the view fit the screen without cropping. If the aspect ratio of the view is different from that of the screen, black edges will appear. V2TXLiveFillModeScaleFill fill the screen with the stretched image, thus the length and width may not change proportionally.

Return Desc:

Return code V2TXLiveCode
V2TXLIVE_OK: successful

startCamera

startCamera

int startCamera	(boolean frontCamera)

Enables the local camera

Param	DESC
frontCamera	Specifies whether to switch to the front camera. true `Default` : switch to the front camera. false: switch to the rear camera.

Note

startVirtualCamera, startCamera, startScreenCapture, if use the same Pusher instance, only one can publish. To switch between different capture sources, first stop the previous capture source, and then start the next capture source to ensure that start and stop of the same capture source are called in pairs. eg: when the capture source is switched from Camera to VirtualCamera, the call sequence is startCamera -> stopCamera -> startVirtualCamera.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

stopCamera

stopCamera

Disables the local camera

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startMicrophone

startMicrophone

Enables the local microphone

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

stopMicrophone

stopMicrophone

Disables the microphone

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startVirtualCamera

startVirtualCamera

int startVirtualCamera	(Bitmap image)

Enables the image streaming

Param	DESC
image	image.

Note

startVirtualCamera, startCamera, startScreenCapture, if use the same Pusher instance, only one can publish. To switch between different capture sources, first stop the previous capture source, and then start the next capture source to ensure that start and stop of the same capture source are called in pairs. eg: when the capture source is switched from Camera to VirtualCamera, the call sequence is startCamera -> stopCamera -> startVirtualCamera.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

stopVirtualCamera

stopVirtualCamera

Disables the image streaming

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startScreenCapture

startScreenCapture

Enables video screen capture

Note

startVirtualCamera, startCamera, startScreenCapture, if use the same Pusher instance, only one can publish. To switch between different capture sources, first stop the previous capture source, and then start the next capture source to ensure that start and stop of the same capture source are called in pairs. eg: when the capture source is switched from Camera to ScreenCapture, the call sequence is startCamera -> stopCamera -> startScreenCapture.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

stopScreenCapture

stopScreenCapture

Disables video capture

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

pauseAudio

pauseAudio

Pause the audio stream of the pusher

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeAudio

resumeAudio

Resume the audio stream of the pusher

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

pauseVideo

pauseVideo

Pause the video stream of the pusher

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeVideo

resumeVideo

Resume the video stream of the pusher

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

startPush

startPush

int startPush	(String url)

Starts pushing the audio and video data

Param	DESC
url	Push URL, which can be any push server.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: operation succeeded. The pusher starts connecting to the target push URL.
V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The URL is invalid.
V2TXLIVE_ERROR_INVALID_LICENSE: operation failed. The license is invalid and authentication failed.
V2TXLIVE_ERROR_REFUSED: operation failed. Duplicate streamId, please ensure that no other player or pusher is using this streamId now.

stopPush

stopPush

Stops pushing the audio and video data

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

isPushing

isPushing

Indicates whether the pusher is currently pushing streams

Return Desc:

Indicates whether the pusher is pushing streams.

1: yes.

0: no.

setAudioQuality

setAudioQuality

int setAudioQuality	(V2TXLiveAudioQuality quality)

Sets the audio quality for pushing

Param	DESC
quality	Audio quality V2TXLiveAudioQuality. V2TXLiveAudioQualityDefault `Default` : universal. V2TXLiveAudioQualitySpeech: speech. V2TXLiveAudioQualityMusic: music.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_REFUSED: the audio quality cannot be adjusted in the pushing process.

setVideoQuality

setVideoQuality

int setVideoQuality	(V2TXLiveVideoEncoderParam param)

Set the video encoding parameters for pushing

Param	DESC
param	video encoding parameters V2TXLiveVideoEncoderParam.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

getBeautyManager

getBeautyManager

Obtains the beauty manager

With the beauty manager, you can use the following features:
Set the following cosmetic effects: beauty style, whitening, ruddy, big eyes, slim face, V-shape face, chin, short face, small nose, bright eyes, white teeth, remove eye bags, remove wrinkles, remove laugh lines.
Adjust the hairline, eye spacing, eye corners, mouth shape, nose wings, nose position, lip thickness, and face shape.
Set animated effects such as face widgets (materials).
Add makeup effects.
Recognize gestures.

please see TXBeautyManager

getAudioEffectManager

getAudioEffectManager

Obtains the audio effect manager

With the audio effect manager, you can use the following features:
Adjust the volume of human voice collected by the microphone.
Set the reverb and voice changing effects.
Start the headphone monitor, and set the volume of the headphone monitor.
Add the BGM, and adjust the playback effect of BGM.

please see TXAudioEffectManager

getDeviceManager

getDeviceManager

Obtains the video device manager

With the device manager, you can use the following features:
Switch between the front and rear cameras.
Set the auto focus.
Adjust the camera magnification.
Turn the flash on or off.
Switch between the earphone and speaker.
Modify the volume type (media volume or conversation volume).

please see TXDeviceManager

snapshot

snapshot

Captures the local view in the pushing process

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.
V2TXLIVE_ERROR_REFUSED: pushing is stopped, and the snapshot operation cannot be called.

setWatermark

setWatermark

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

Sets the pusher watermark image. By default, the watermark is disabled

Param	DESC
image	Watermark image. If the value is null, it is equivalent to disabling the watermark.

scale	Scaling ratio of the watermark. Valid range: 0 - 1.
x	Display position of the watermark. Valid range: 0 - 1.
y	Display position of the watermark. Valid range: 0 - 1.

Return Desc:

Return code for V2TXLiveCode

V2TXLIVE_OK: successful

enableVolumeEvaluation

enableVolumeEvaluation

int enableVolumeEvaluation	(int intervalMs)

Enables volume update

After this feature is enabled, you can obtain the volume evaluation through the onMicrophoneVolumeUpdate callback.

Param	DESC
intervalMs	Interval for triggering the volume callback. The unit is ms. The minimum interval is 100 ms. If the value is equal to or smaller than 0, the callback is disabled. We recommend that you set this parameter to 300 ms. `Default` : 0.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

enableCustomVideoProcess

enableCustomVideoProcess

int enableCustomVideoProcess	(boolean enable
	V2TXLivePixelFormat pixelFormat
	V2TXLiveBufferType bufferType)

Enables or disables custom video processing

Param	DESC
enable	`true` : enable; `false` : disable (**default**).

Note

Supported format combinations:

V2TXLivePixelFormatTexture2D+V2TXLiveBufferTypeTexture

V2TXLivePixelFormatI420+V2TXLiveBufferTypeByteBuffer

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_NOT_SUPPORTED` : unsupported format.

enableCustomVideoCapture

enableCustomVideoCapture

int enableCustomVideoCapture	(boolean enable)

Enables or disables custom video capture

In the custom video capture mode, the SDK no longer captures images from cameras. Only the encoding and sending capabilities are retained.

Param	DESC
enable	`true` : enable custom video capture; `false` (**default**): disable custom video capture.

Note

This API takes effect only when it is called before startPush.

Return Desc:

Return code for V2TXLiveCode.

V2TXLIVE_OK: successful.

enableCustomAudioCapture

enableCustomAudioCapture

int enableCustomAudioCapture	(boolean enable)

Turn on/off custom audio capture

@brief Turn on/off custom audio capture.
In the custom audio capture mode, the SDK no longer collects sound from the microphone, and only retains the encoding and sending capabilities.

@note It needs to be called before startPush to take effect.

@param enable true: Open custom capture; false: Close custom capture. `Default value` : `false` .
@return Return code for V2TXLiveCode.
`V2TXLIVE_OK` : successful.

sendCustomVideoFrame

sendCustomVideoFrame

int sendCustomVideoFrame	(V2TXLiveVideoFrame videoFrame)

Sends the collected video data to the SDK in the custom video capture mode

In the custom video capture mode, the SDK no longer captures images from cameras. Only the encoding and sending capabilities are retained.
You can pack collected SampleBuffer packets into V2TXLiveVideoFrame and periodically send them through this API.

Param	DESC
videoFrame	Video frames sent to the SDK V2TXLiveVideoFrame.

Note

You must call enableCustomVideoCapture to enable custom video capture before startPush .

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.
V2TXLIVE_ERROR_INVALID_PARAMETER: The video frames fail to be sent because they are invalid.

sendCustomAudioFrame

sendCustomAudioFrame

int sendCustomAudioFrame	(V2TXLiveAudioFrame audioFrame)

In the custom audio collection mode, send the collected audio data to the SDK

Param	DESC
audioFrame	Audio frame data sent to SDK V2TXLiveAudioFrame.

Note

You need to call enableCustomAudioCapture(boolean) before startPush to enable custom capture.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_INVALID_PARAMETER` : The audio frames fail to be sent because they are invalid.

enableAudioProcessObserver

enableAudioProcessObserver

int enableAudioProcessObserver	(boolean enable
	V2TXLiveDef.V2TXLiveAudioFrameObserverFormat format)

Enables/Disables audio process callback

Param	DESC
enable	`true` : enable; `false` (**default**): disable.
format	audio frame format.

Note

This API works only if you call it before startPush.

sendSeiMessage

sendSeiMessage

int sendSeiMessage	(int payloadType
	byte[] data)

Use SEI channel to send custom message

The player end V2TXLivePlayer can receive the message via `onReceiveSeiMessage` callback in V2TXLivePlayerObserver.

Param	DESC
data	Data to be sent.
payloadType	Payload type. Valid values: `5` , `242` , `242` recommended.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.

startSystemAudioLoopback

startSystemAudioLoopback

Enable system audio capturing

Captures the audio of the entire OS.
The audio is then mixed into the audio captured by the mic before being published to the cloud.
Note
1. This interface only works on Android API 29 and above.
2. You need to use this interface to enable system sound capture first, and it will take effect only when you call startScreenCapture to enable screen sharing.
3. You need to add a foreground service to ensure that the system sound capture is not silenced, and set android:foregroundServiceType="mediaProjection".
4. The SDK only capture audio of applications that satisfies the capture strategy and audio usage. Currently, the audio usage captured by the SDK includes USAGE_MEDIA, USAGE_GAME。

showDebugView

showDebugView

void showDebugView	(boolean isShow)

Indicates whether the debug view of the pusher video status information is displayed

Param	DESC
isShow	Specifies whether to display the debug view. `Default` : false.

setProperty

setProperty

int setProperty	(String key
	Object value)

Calls the advanced API of V2TXLivePusher

Param	DESC
key	Key of the advanced API, please see V2TXLiveProperty.
value	Parameter needed to call the advanced API corresponding to the key.

Note

This API is used to call some advanced features.

Return Desc:

Return code for V2TXLiveCode.
V2TXLIVE_OK: successful.
V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The key cannot be nil.

setMixTranscodingConfig

setMixTranscodingConfig

int setMixTranscodingConfig	(V2TXLiveTranscodingConfig config)

Sets On-Cloud MixTranscoding parameters

If you have enabled relayed push on the "Function Configuration" page of the TRTC console, then each stream in a room will have a default CDN address.

There may be multiple anchors in a room, each sending their own video and audio, but CDN audience needs only one live stream.
Therefore, you need to mix multiple audio/video streams into one standard live stream, which requires mixing and transcoding.

When you call the `setMixTranscodingConfig()` API, the SDK will send a command to the Tencent Cloud transcoding server to combine multiple audio/video streams in the room into one stream.
You can use the `mixStreams` parameter to set the position of each channel of image and specify whether to mix only audio. You can also set the encoding parameters of the mixed stream, including `videoWidth` , `videoHeight` , and `videoBitrate` .

Image 1 => decoding ====> \\
\\
Image 2=> decoding => image mixing => encoding => **mixed image**
/
Image 3 => decoding ====> /

Audio 1 => decoding ====> \\
\\
Audio 2 => decoding => audio mixing => encoding => **mixed audio**
/
Audio 3 => decoding ====> /

For more information, please see On-Cloud MixTranscoding.

Param	DESC
config	Please see the description of V2TXLiveTranscodingConfig in `V2TXLiveDef.h` . Passing in `nil` will cancel On-Cloud MixTranscoding.

Note

Notes:

Only supported RTC mode.

On-Cloud MixTranscoding will increase the delay of CDN live streaming by about 1-2 seconds.

If you call this API, the streams of co-anchors will be mixed into your stream or the `streamId` specified in `config`.

If you are still in the room but do not need to mix streams anymore, make sure that you pass in `nil` to cancel On-Cloud MixTranscoding. The On-Cloud MixTranscoding module starts working the moment you enable On-Cloud MixTranscoding. You may incur additional costs if you do not cancel it in a timely manner.

When you leave the room, mixing will be canceled automatically.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_REFUSED` : failed to set On-Cloud MixTranscoding parameters as stream pushing has not started.

startLocalRecording

startLocalRecording

int startLocalRecording	(V2TXLiveLocalRecordingParams params)

Start recording audio and video stream

Note

The recording can only be started after the push stream is started, and it is invalid to start the recording in the non-push state.

Do not dynamically switch the resolution and soft/hard editing during the recording process, as there is a high probability that the generated video will be abnormal.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_INVALID_PARAMETER` : The parameter is invalid, such as filePath is empty.

`V2TXLIVE_ERROR_REFUSED` : API refuse, you must first call startPush to start publishing streaming.

stopLocalRecording

stopLocalRecording

Stop recording audio and video stream

Note

When the push stream is stopped, if the video is still being recorded, the SDK will automatically end the recording.

enableVoiceActivityDetection

enableVoiceActivityDetection

void enableVoiceActivityDetection	(boolean enable)

Enable voice activity detection

Note

After turning on, you can get the start and stop of voice activities in the OnVoiceActivityDetectionUpdate callback

V2TXLivePusherObserver

Last updated：2024-06-06 15:47:57

Module: V2TXLivePusherObserver @ TXLiteAVSDK

Function: Tencent Cloud live pusher callback notification

Function

Callback notification for push streaming of Tencent Cloud Live.

Introduce

You can receive some push notifications from the V2TXLivePusher pusher, including the connection status of the pusher, callback of the first frame of audio and video, statistical data, warning and error messages, etc.

V2TXLivePusherObserver

V2TXLivePusherObserver

FuncList	DESC
onError	Live pusher error notification, which is called back when the pusher encounters an error
onWarning	Live pusher warning notification
onCaptureFirstAudioFrame	Callback notification indicating that collection of the first audio frame is complete
onCaptureFirstVideoFrame	Callback notification indicating that collection of the first video frame is complete
onMicrophoneVolumeUpdate	Microphone-collected volume callback

onPushStatusUpdate	Callback notification of the pusher connection status
onStatisticsUpdate	Live pusher statistics callback
onSnapshotComplete	Screenshot callback
onGLContextCreated	Callback of created the OpenGL context in the SDK
onProcessAudioFrame	Audio data captured by the local mic, pre-processed by the audio module, effect-processed and BGM-mixed
onProcessVideoFrame	Custom video processing callback
onGLContextDestroyed	Callback of destroying the OpenGL context in the SDK
onSetMixTranscodingConfig	Callback of setting On-Cloud MixTranscoding parameters, which corresponds to the setMixTranscodingConfig API
onScreenCaptureStarted	The SDK returns this callback when you call startScreenCapture and other APIs to start screen sharing.
onScreenCaptureStopped	The SDK returns this callback when you call stopScreenCapture to stop screen sharing
onLocalRecordBegin	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete	The SDK returns this callback when you call stopLocalRecording to start local recording.
onVoiceActivityDetectionUpdate	After calling enableVoiceActivityDetection to turn on voice activity detection, you will receive this callback notification when the anchor starts or stops speaking.

onError

onError

void onError	(int code
	String msg
	Bundle extraInfo)

Live pusher error notification, which is called back when the pusher encounters an error

Param	DESC
code	Error code V2TXLiveCode.
extraInfo	Extended information.
msg	Error message.

onWarning

onWarning

void onWarning	(int code
	String msg
	Bundle extraInfo)

Live pusher warning notification

Param	DESC
code	Warning code V2TXLiveCode.
extraInfo	Extended information.
msg	Warning message.

onCaptureFirstAudioFrame

onCaptureFirstAudioFrame

Callback notification indicating that collection of the first audio frame is complete

onCaptureFirstVideoFrame

onCaptureFirstVideoFrame

Callback notification indicating that collection of the first video frame is complete

onMicrophoneVolumeUpdate

onMicrophoneVolumeUpdate

void onMicrophoneVolumeUpdate	(int volume)

Microphone-collected volume callback

Param	DESC
volume	Current volume value for collection.

Note

This callback notification is received after enableVolumeEvaluation is called.

onPushStatusUpdate

onPushStatusUpdate

void onPushStatusUpdate	(V2TXLivePushStatus status
	String msg
	Bundle extraInfo)

Callback notification of the pusher connection status

Param	DESC
extraInfo	Extended information.
msg	Connection status message.
status	Pusher connection status V2TXLivePushStatus .

onStatisticsUpdate

onStatisticsUpdate

void onStatisticsUpdate	(V2TXLivePusherStatistics statistics)

Live pusher statistics callback

Param	DESC
statistics	Pusher statistics V2TXLivePusherStatistics .

onSnapshotComplete

onSnapshotComplete

void onSnapshotComplete	(Bitmap image)

Screenshot callback

Param	DESC
image	Captured video image.

Note

This callback notification will be received after calling V2TXLivePusher#snapshot .

onGLContextCreated

onGLContextCreated

Callback of created the OpenGL context in the SDK

onProcessAudioFrame

onProcessAudioFrame

void onProcessAudioFrame	(V2TXLiveDef.V2TXLiveAudioFrame frame)

Audio data captured by the local mic, pre-processed by the audio module, effect-processed and BGM-mixed

After you configure the callback of custom audio processing, the SDK will return via this callback the data captured, pre-processed (ANS, AEC, and AGC), effect-processed and BGM-mixed in PCM format, before it is submitted to the network module for encoding.

The audio data returned via this callback is in PCM format and has a fixed frame length (time) of 0.02s.

The formula to convert a frame length in seconds to one in bytes is **sample rate * frame length in seconds * number of sound channels * audio bit depth**.

Assume that the audio is recorded on a single channel with a sample rate of 48,000 Hz and audio bit depth of 16 bits, which are the default settings of TRTC. The frame length in bytes will be **48000 * 0.02s * 1 * 16 bits = 15360 bits = 1920 bytes**.

Param	DESC
frame	Audio frames in PCM format

Note

1. Please avoid time-consuming operations in this callback function. The SDK processes an audio frame every 20 ms, so if your operation takes more than 20 ms, it will cause audio exceptions.
2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.

onProcessVideoFrame

onProcessVideoFrame

int onProcessVideoFrame	(V2TXLiveVideoFrame srcFrame
	V2TXLiveVideoFrame dstFrame)

Custom video processing callback

Param	DESC
dstFrame	For images after processing.
srcFrame	For images before processing.

Note

You will receive this callback only after you call enableCustomVideoProcess to enable custom video processing.

Case 1: The beauty filter component generates new textures.

If the beauty filter component you use generates a new texture frame (for the processed image) during image processing, please set `dstFrame.textureId` to a new texture ID in the callback API.

```
@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();
}
```

Case 2: The third-party beauty filter component doesn't generate new textures.

If the third-party beauty filter component you use does not generate new textures and you need to manually set an input texture and an output texture for the component, please consider the following scheme:

```
int onProcessVideoFrame(V2TXLiveVideoFrame srcFrame, V2TXLiveVideoFrame dstFrame) {
thirdparty_process(srcFrame.texture.textureId, srcFrame.width, srcFrame.height, dstFrame.texture.textureId);
return 0;
}
```

onGLContextDestroyed

onGLContextDestroyed

Callback of destroying the OpenGL context in the SDK

onSetMixTranscodingConfig

onSetMixTranscodingConfig

void onSetMixTranscodingConfig	(int code
	String msg)

Callback of setting On-Cloud MixTranscoding parameters, which corresponds to the {@link setMixTranscodingConfig} API

Param	DESC

code	0: successful; other values: failed.
msg	Error message.

onScreenCaptureStarted

onScreenCaptureStarted

The SDK returns this callback when you call {@link startScreenCapture} and other APIs to start screen sharing.

onScreenCaptureStopped

onScreenCaptureStopped

void onScreenCaptureStopped	(int reason)

The SDK returns this callback when you call {@link stopScreenCapture} to stop screen sharing

Param	DESC
Reason	for stop. `0` : Screen capture stopped by user. `1` : On iOS platform means the screen recording is interrupted by the system; Mac, Windows means the screen sharing window is closed. `2` : On windows platform indicates that the display screen status of screen sharing is changed (such as the interface is pulled out, the projection mode is changed, etc.); other platforms do not throw.

onLocalRecordBegin

onLocalRecordBegin

void onLocalRecordBegin	(int code
	String storagePath)

The SDK returns this callback when you call {@link startLocalRecording} to start local recording.

Param	DESC

code	status. 0: successful. -1: failed. -2: unsupported format. -6: recording has been started. Stop recording first. -7: recording file already exists and needs to be deleted. -8: recording directory does not have the write permission. Please check the directory permission.
storagePath	recording filePath.

onLocalRecording

onLocalRecording

void onLocalRecording	(long durationMs
	String storagePath)

The SDK returns this callback when you call {@link startLocalRecording} to start local recording, which means recording task in progress.

Param	DESC
durationMs	recording duration.
storagePath	recording filePath.

onLocalRecordComplete

onLocalRecordComplete

void onLocalRecordComplete	(int code
	String storagePath)

The SDK returns this callback when you call {@link stopLocalRecording} to start local recording.

Param	DESC
code	status 0: successful.

	-1: failed. -2: Switching resolution or horizontal and vertical screen causes the recording to stop. -3: recording duration is too short or no video or audio data is received. Check the recording duration or whether audio or video capture is enabled.
storagePath	recording filePath.

onVoiceActivityDetectionUpdate

onVoiceActivityDetectionUpdate

void onVoiceActivityDetectionUpdate	(boolean active)

After calling {@link enableVoiceActivityDetection} to turn on voice activity detection, you will receive this callback notification when the anchor starts or stops speaking.

Param	DESC
active	The voice starts or stops.

V2TXLivePlayer

Last updated：2024-06-06 15:47:57

Module: V2TXLivePlayer @ TXLiteAVSDK

Function: Tencent Cloud live player

Function

Tencent Cloud Live Player.
It is mainly responsible for pulling audio and video data from the specified live stream address, decoding and rendering locally.

Introduce

The player includes the following capabilities:
Support RTMP, HTTP-FLV, HLS, TRTC, WebRTC protocols.
Screen capture, you can capture the video screen of the current live stream.
Delay adjustment, you can set the minimum and maximum time for automatic adjustment of the player cache.
Customized video data processing, you can process the video data in the live stream according to the needs of the project, and then render and play it.

V2TXLivePlayer

V2TXLivePlayer

FuncList	DESC
setObserver	Sets the player callback
setRenderView	Sets the rendering view of the player. This control is responsible for presenting the video content

setRenderView	Sets the rendering view of the player. This control is responsible for presenting the video content
setRenderView	Sets the rendering view of the player. This control is responsible for presenting the video content
setRenderRotation	Sets the rotation angle of the player view
setRenderFillMode	Sets the fill mode of the view
startLivePlay	Starts playing the audio and video streams
stopPlay	Stops playing the audio and video streams
isPlaying	Indicates whether the player is playing the audio and video streams
pauseAudio	Pauses the audio stream of the player
resumeAudio	Resumes the audio stream of the player
pauseVideo	Pauses the video stream of the player
resumeVideo	Resumes the video stream of the player
setPlayoutVolume	Sets the volume
setCacheParams	Set the minimum time and maximum time (unit: s) for auto adjustment of the player cache
switchStream	Seamlessly switch live stream urls, supporting FLV and LEB protocols
getStreamList	Get Stream Info List
enableVolumeEvaluation	Enables playback volume update
snapshot	Captures the video view in the playback process
enableObserveVideoFrame	Turn on/off the monitoring callback of the video frame
enableObserveAudioFrame	Turn on/off the monitoring callback of the audio frame
enableReceiveSeiMessage	Enables the receiving of SEI messages
showDebugView	Indicates whether the debug view of the player video status information is displayed
setProperty	Calls the advanced API of V2TXLivePlayer
startLocalRecording	Start recording audio and video stream

stopLocalRecording	Stop recording audio and video stream

setObserver

setObserver

void setObserver	(V2TXLivePlayerObserver observer)

Sets the player callback

By setting the callback, you can listen to some callback events of V2TXLivePlayer,
including the player status, playback volume callback, first frame audio/video callback, statistics, warnings, and error messages.

Param	DESC
observer	Callback target of the player. For more information, see V2TXLivePlayerObserver.

setRenderView

setRenderView

int setRenderView	(TXCloudVideoView view)

Sets the rendering view of the player. This control is responsible for presenting the video content

Param	DESC
view	Player rendering view.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderView

setRenderView

int setRenderView	(TextureView view)

Sets the rendering view of the player. This control is responsible for presenting the video content

Param	DESC
view	Player rendering view.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderView

setRenderView

int setRenderView	(SurfaceView view)

Sets the rendering view of the player. This control is responsible for presenting the video content

Param	DESC
view	Player rendering view.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderRotation

setRenderRotation

int setRenderRotation	(V2TXLiveRotation rotation)

Sets the rotation angle of the player view

Param	DESC
rotation	Rotation angle of the view V2TXLiveRotation. V2TXLiveRotation0 **Default**: 0 degrees, which means the view is not rotated. V2TXLiveRotation90: rotate 90 degrees clockwise. V2TXLiveRotation180: rotate 180 degrees clockwise.

	V2TXLiveRotation270: rotate 270 degrees clockwise.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

setRenderFillMode

setRenderFillMode

int setRenderFillMode	(V2TXLiveFillMode mode)

Sets the fill mode of the view

Param	DESC
mode	Fill mode of the view V2TXLiveFillMode. V2TXLiveFillModeFill: **Default**: fill the screen with the image without leaving any black edges. If the aspect ratio of the view is different from that of the screen, part of the view will be cropped. V2TXLiveFillModeFit make the view fit the screen without cropping. If the aspect ratio of the view is different from that of the screen, black edges will appear. V2TXLiveFillModeScaleFill fill the screen with the stretched image, thus the length and width may not change proportionally.

Return Desc:

Return code V2TXLiveCode
V2TXLIVE_OK: successful

startLivePlay

startLivePlay

int startLivePlay	(String url)

Starts playing the audio and video streams

Param	DESC
url	URL of the audio and video streams to be played. The RTMP, HTTP-FLV and TRTC

	streaming protocols are supported.

Note

Starting from version 10.7, the Licence needs to be set through setLicence or setLicence before it can be played successfully, otherwise the playback will fail (black screen), and it can only be set once globally. Live Licence, UGC Licence, and Player Licence can all be used. If you have not obtained the above Licence, you can `quickly apply for a beta Licence for free` To play, the official licence needs to be `purchased` .

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: operation succeeded. The player starts connecting to the URL and playing the audio and video streams.
V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The URL is invalid.
V2TXLIVE_ERROR_REFUSED: operation failed. Duplicate streamId, please ensure that no other player or pusher is using this streamId now.
V2TXLIVE_ERROR_INVALID_LICENSE: The licence is invalid and the playback fails.

stopPlay

stopPlay

Stops playing the audio and video streams

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

isPlaying

isPlaying

Indicates whether the player is playing the audio and video streams

Return Desc:

Indicates whether the player is playing the audio and video streams.
1: yes.
0: no.

pauseAudio

pauseAudio

Pauses the audio stream of the player

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeAudio

resumeAudio

Resumes the audio stream of the player

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

pauseVideo

pauseVideo

Pauses the video stream of the player

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

resumeVideo

resumeVideo

Resumes the video stream of the player

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

setPlayoutVolume

setPlayoutVolume

int setPlayoutVolume	(int volume)

Sets the volume

Param	DESC
volume	Volume. Valid range: 0 - 100. **Default**: 100.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

setCacheParams

setCacheParams

int setCacheParams	(float minTime
	float maxTime)

Set the minimum time and maximum time (unit: s) for auto adjustment of the player cache

Param	DESC
maxTime	Maximum time for auto cache adjustment. The value must be greater than 0. **Default**: 5.
minTime	Minimum time for auto cache adjustment. The value must be greater than 0. **Default**: 1.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. MinTime and maxTime must be greater than 0.

V2TXLIVE_ERROR_REFUSED: operation failed. Change of cache is not suppoted when playing.

switchStream

switchStream

int switchStream	(String newUrl)

Seamlessly switch live stream urls, supporting FLV and LEB protocols

Param	DESC
newUrl	New pull address.

getStreamList

getStreamList

Get Stream Info List

enableVolumeEvaluation

enableVolumeEvaluation

int enableVolumeEvaluation	(int intervalMs)

Enables playback volume update

After this feature is enabled, you can obtain the SDK's volume evaluation through the onPlayoutVolumeUpdate callback.

Param	DESC
intervalMs	Interval for triggering the volume callback. The unit is ms. The minimum interval is 100 ms. If the value is equal to or smaller than 0, the callback is disabled. We recommend that you set this parameter to 300 ms. **Default**: 0.

Return Desc:

Return code V2TXLiveCode.
V2TXLIVE_OK: successful.

snapshot

snapshot

Captures the video view in the playback process

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_REFUSED: playback is stopped, the snapshot operation cannot be called.

enableObserveVideoFrame

enableObserveVideoFrame

int enableObserveVideoFrame	(boolean enable
	V2TXLivePixelFormat pixelFormat
	V2TXLiveBufferType bufferType)

Turn on/off the monitoring callback of the video frame

The SDK will no longer render the video after you turn on this switch. You can get the video frame through V2TXLivePlayerObserver and execute custom rendering logic.

Param	DESC
bufferType	Video data format for custom rendering callback V2TXLiveBufferType。
enable	Whether to enable custom rendering. **Default**: false.
pixelFormat	Video pixel format for custom rendering callback V2TXLivePixelFormat。

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_NOT_SUPPORTED: the pixel format or data format is not supported.

enableObserveAudioFrame

enableObserveAudioFrame

int enableObserveAudioFrame	(boolean enable)

Turn on/off the monitoring callback of the audio frame

if you turn on this switch, You can get the audio frame through V2TXLivePlayerObserver and execute custom logic.

Param	DESC
enable	Whether to enable the callback of the audio frame. **Default**: false.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

enableReceiveSeiMessage

enableReceiveSeiMessage

int enableReceiveSeiMessage	(boolean enable
	int payloadType)

Enables the receiving of SEI messages

Param	DESC
enable	`true` : enable; `false` (**default**): disable.
payloadType	The payload type of SEI messages. Valid values: `5` , `242` , `243` , please be consistent with the payload type of the sender.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

showDebugView

showDebugView

void showDebugView	(boolean isShow)

Indicates whether the debug view of the player video status information is displayed

Param	DESC
isShow	Specifies whether to display the debug view. **Default**: false.

setProperty

setProperty

int setProperty	(String key
	Object value)

Calls the advanced API of V2TXLivePlayer

Param	DESC
key	Key of the advanced API.
value	Parameter needed to call the advanced API corresponding to the key.

Note

This API is used to call some advanced features.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed. The key cannot be null.

startLocalRecording

startLocalRecording

int startLocalRecording	(V2TXLiveLocalRecordingParams params)

Start recording audio and video stream

Note

The recording can only be started after the play stream is started, and it is invalid to start the recording in the non-play state.

Do not dynamically switch soft/hard decoding during the recording process, as there is a high probability that the generated video will be abnormal.

Return Desc:

Return code for V2TXLiveCode.

`V2TXLIVE_OK` : successful.

`V2TXLIVE_ERROR_INVALID_PARAMETER` : The parameter is invalid, such as filePath is empty.

`V2TXLIVE_ERROR_REFUSED` : API refuse, you must first call startLivePlay to start playing streaming.

stopLocalRecording

stopLocalRecording

Stop recording audio and video stream

Note

When the play stream is stopped, if the video is still being recorded, the SDK will automatically end the recording.

V2TXLivePlayerObserver

Last updated：2024-06-06 15:47:57

Module: V2TXLivePlayerObserver @ TXLiteAVSDK

Function: Tencent Cloud live player callback notification

Function

Player callback notification for Tencent Cloud Live.

Introduce

You can receive some callback notifications from the V2TXLivePlayer player, including player status, playback volume callback, audio and video first frame callback, statistical data, warning and error messages, etc.

V2TXLivePlayerObserver

V2TXLivePlayerObserver

FuncList	DESC
onError	live player error notification, which is called back when the player encounters an error
onWarning	live player warning notification
onVideoResolutionChanged	live player resolution change notification
onConnected	live player has successfully connected to the server notification
onVideoPlaying	Video playback event
onAudioPlaying	Audio playback event

onVideoLoading	Video loading event
onAudioLoading	Audio loading event
onPlayoutVolumeUpdate	Player playback volume callback
onStatisticsUpdate	Live player statistics callback
onSnapshotComplete	Screenshot callback
onRenderVideoFrame	Custom video rendering callback
onPlayoutAudioFrame	Audio Data callback
onReceiveSeiMessage	Callback of receiving an SEI message. The sender calls `sendSeiMessage` in V2TXLivePusher to send an SEI
onStreamSwitched	Resolution stream switch callback
onLocalRecordBegin	The SDK returns this callback when you call startLocalRecording to start local recording.
onLocalRecording	The SDK returns this callback when you call startLocalRecording to start local recording, which means recording task in progress.
onLocalRecordComplete	The SDK returns this callback when you call stopLocalRecording to start local recording.

onError

onError

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

live player error notification, which is called back when the player encounters an error

Param	DESC
code	Error code V2TXLiveCode.

extraInfo	Extended information.
msg	Error message.
player	Player object that calls back this notification.

onWarning

onWarning

void onWarning	(V2TXLivePlayer player
	int code
	String msg
	Bundle extraInfo)

live player warning notification

Param	DESC
code	Warning code V2TXLiveCode.
extraInfo	Extended information.
msg	Warning message.
player	Player object that calls back this notification.

onVideoResolutionChanged

onVideoResolutionChanged

void onVideoResolutionChanged	(V2TXLivePlayer player
	int width
	int height)

live player resolution change notification

Param	DESC

height	Video height.
player	Player object that calls back this notification.
width	Video width.

onConnected

onConnected

void onConnected	(V2TXLivePlayer player
	Bundle extraInfo)

live player has successfully connected to the server notification

Param	DESC
extraInfo	Extended information.
player	Player object that calls back this notification.

onVideoPlaying

onVideoPlaying

void onVideoPlaying	(V2TXLivePlayer player
	boolean firstPlay
	Bundle extraInfo)

Video playback event

Param	DESC
extraInfo	Extended information.
firstPlay	Play for the first time.
player	Player object that calls back this notification.

onAudioPlaying

onAudioPlaying

void onAudioPlaying	(V2TXLivePlayer player
	boolean firstPlay
	Bundle extraInfo)

Audio playback event

Param	DESC
extraInfo	Extended information.
firstPlay	Play for the first time.
player	Player object that calls back this notification.

onVideoLoading

onVideoLoading

void onVideoLoading	(V2TXLivePlayer player
	Bundle extraInfo)

Video loading event

Param	DESC
extraInfo	Extended information.
player	Player object that calls back this notification.

onAudioLoading

onAudioLoading

void onAudioLoading	(V2TXLivePlayer player
	Bundle extraInfo)

Audio loading event

Param	DESC
extraInfo	Extended information.
player	Player object that calls back this notification.

onPlayoutVolumeUpdate

onPlayoutVolumeUpdate

void onPlayoutVolumeUpdate	(V2TXLivePlayer player
	int volume)

Player playback volume callback

Param	DESC
player	Player object that calls back this notification.
volume	Current playback volume.

Note

This callback notification is received after enableVolumeEvaluation is called to enable playback volume display.

onStatisticsUpdate

onStatisticsUpdate

void onStatisticsUpdate	(V2TXLivePlayer player
	V2TXLivePlayerStatistics statistics)

Live player statistics callback

Param	DESC
player	Player object that calls back this notification.

statistics	Player statistics V2TXLivePlayerStatistics.

onSnapshotComplete

onSnapshotComplete

void onSnapshotComplete	(V2TXLivePlayer player
	Bitmap image)

Screenshot callback

Param	DESC
image	Captured video image.
player	Player object that calls back this notification.

onRenderVideoFrame

onRenderVideoFrame

void onRenderVideoFrame	(V2TXLivePlayer player
	V2TXLiveVideoFrame videoFrame)

Custom video rendering callback

Param	DESC
player	Player object that calls back this notification.
videoFrame	Video frame data V2TXLiveVideoFrame.

Note

Need you call enableObserveVideoFrame to turn on the callback switch.

onPlayoutAudioFrame

onPlayoutAudioFrame

void onPlayoutAudioFrame	(V2TXLivePlayer player
	V2TXLiveAudioFrame audioFrame)

Audio Data callback

Param	DESC
aduioFrame	Audio frame data V2TXLiveAudioFrame.
player	Player object that calls back this notification.

Note

Need you call enableObserveAudioFrame to turn on the callback switch. Please use the data of audioFrame in the current callback.

onReceiveSeiMessage

onReceiveSeiMessage

void onReceiveSeiMessage	(V2TXLivePlayer player
	int payloadType
	byte[] data)

Callback of receiving an SEI message. The sender calls `sendSeiMessage` in {@link V2TXLivePusher} to send an SEI

Param	DESC
data	sei message data.
payloadType	The payload type of the received SEI message.
player	Player object that calls back this notification.

Note

You will receive this callback after calling `enableReceiveSeiMessage` in V2TXLivePlayer to enable the receiving of SEI.

onStreamSwitched

onStreamSwitched

void onStreamSwitched	(V2TXLivePlayer player
	String url
	int code)

Resolution stream switch callback

Param	DESC
code	Status code, 0:success, -1:timeout, -2:failed, server error, -3:failed, client error.
player	Player object that calls back this notification.
url	Switched playback address.

Note

This callback notification is received after switchStream is called to switch stream.

onLocalRecordBegin

onLocalRecordBegin

void onLocalRecordBegin	(V2TXLivePlayer player
	int code
	String storagePath)

The SDK returns this callback when you call {@link startLocalRecording} to start local recording.

Param	DESC
code	status. 0: successful. -1: failed. -2: unsupported format. -6: recording has been started. Stop recording first. -7: recording file already exists and needs to be deleted.

	-8: recording directory does not have the write permission. Please check the directory permission.
player	Player object that calls back this notification.
storagePath	recording filePath.

onLocalRecording

onLocalRecording

void onLocalRecording	(V2TXLivePlayer player
	long durationMs
	String storagePath)

The SDK returns this callback when you call {@link startLocalRecording} to start local recording, which means recording task in progress.

Param	DESC
durationMs	recording duration.
player	Player object that calls back this notification.
storagePath	recording filePath.

onLocalRecordComplete

onLocalRecordComplete

void onLocalRecordComplete	(V2TXLivePlayer player
	int code
	String storagePath)

The SDK returns this callback when you call {@link stopLocalRecording} to start local recording.

Param	DESC
code	status

	0: successful. -1: failed. -2: Switching resolution or horizontal and vertical screen causes the recording to stop. -3: recording duration is too short or no video or audio data is received. Check the recording duration or whether audio or video capture is enabled.
player	Player object that calls back this notification.
storagePath	recording filePath.

V2TXLivePremier

Last updated：2024-06-06 15:47:57

Module: V2TXLivePremier @ TXLiteAVSDK

Function: V2TXLive High-level interface

V2TXLivePremier

V2TXLivePremierObserver

FuncList	DESC
onLog	Custom Log output callback interface
onLicenceLoaded	setLicence result callback interface
onCaptureAudioFrame	Raw audio data captured locally
onPlayoutAudioFrame	Data mixed from each channel before being submitted to the system for playback
onVoiceEarMonitorAudioFrame	In-ear monitoring data

V2TXLivePremier

FuncList	DESC
getSDKVersionStr	Get the SDK version number
setObserver	Set V2TXLivePremier callback interface
setLogConfig	Set Log configuration information
setEnvironment	Set up SDK access environment
setLicence	Set SDK authorization license

setSocks5Proxy	Set SDK socks5 proxy config
enableAudioCaptureObserver	Enables/Disables audio capture callback
enableAudioPlayoutObserver	Enables/Disables audio playout callback
enableVoiceEarMonitorObserver	Enables/Disables in-ear monitoring callback
setUserId	Set user id
callExperimentalAPI	Call experimental APIs

onLog

onLog

void onLog	(int level
	String log)

Custom Log output callback interface

onLicenceLoaded

onLicenceLoaded

void onLicenceLoaded	(int result
	String reason)

setLicence result callback interface

Param	DESC
reason	the reason for failure.
result	the result of setLicence interface, 0 succeeds, negative number fails.

onCaptureAudioFrame

onCaptureAudioFrame

void onCaptureAudioFrame	(V2TXLiveDef.V2TXLiveAudioFrame frame)

Raw audio data captured locally

Param	DESC
frame	Audio frames in PCM format.

Note

1. Please avoid time-consuming operations in this callback function. The SDK processes an audio frame every 20 ms, so if your operation takes more than 20 ms, it will cause audio exceptions.
2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.
3. The audio data returned via this callback **does not include** pre-processing effects like background music, audio effects, or reverb, and therefore has a very short delay.

onPlayoutAudioFrame

onPlayoutAudioFrame

void onPlayoutAudioFrame	(V2TXLiveDef.V2TXLiveAudioFrame frame)

Data mixed from each channel before being submitted to the system for playback

After you configure the callback of custom audio processing, the SDK will return to you via this callback the data (PCM format) mixed from each channel before it is submitted to the system for playback.

The audio data returned via this callback is in PCM format and has a fixed frame length (time) of 0.02s.

The formula to convert a frame length in seconds to one in bytes is **sample rate * frame length in seconds * number of sound channels * audio bit depth**.

Assume that the audio is recorded on a single channel with a sample rate of 48,000 Hz and audio bit depth of 16 bits, which are the default settings of SDK. The frame length in bytes will be **48000 * 0.02s * 1 * 16 bits = 15360 bits = 1920 bytes**.

Param	DESC
frame	Audio frames in PCM format.

Note

1. Please avoid time-consuming operations in this callback function. The SDK processes an audio frame every 20 ms, so if your operation takes more than 20 ms, it will cause audio exceptions.

2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.
3. The audio data returned via this callback is the audio data mixed from each channel before it is played. It does not include the in-ear monitoring data.

onVoiceEarMonitorAudioFrame

onVoiceEarMonitorAudioFrame

void onVoiceEarMonitorAudioFrame	(V2TXLiveDef.V2TXLiveAudioFrame frame)

In-ear monitoring data

After you configure the callback of custom audio processing, the SDK will return to you via this callback the in-ear monitoring data (PCM format) before it is submitted to the system for playback.
The audio returned is in PCM format and has a not-fixed frame length (time).
The formula to convert a frame length in seconds to one in bytes is **sample rate * frame length in seconds * number of sound channels * audio bit depth**.
Assume that the audio is recorded on a single channel with a sample rate of 48,000 Hz and audio bit depth of 16 bits, which are the default settings of TRTC. The length of 0.02s frame in bytes will be **48000 * 0.02s * 1 * 16 bits = 15360 bits = 1920 bytes**.

Param	DESC
frame	Audio frames in PCM format

Note

1. Please avoid time-consuming operations in this callback function, or it will cause audio exceptions.
2. The audio data returned via this callback can be read and modified, but please keep the duration of your operation short.

getSDKVersionStr

getSDKVersionStr

Get the SDK version number

setObserver

setObserver

void setObserver	(V2TXLivePremierObserver observer)

Set V2TXLivePremier callback interface

setLogConfig

setLogConfig

void setLogConfig	(V2TXLiveDef.V2TXLiveLogConfig config)

Set Log configuration information

setEnvironment

setEnvironment

void setEnvironment	(String env)

Set up SDK access environment

Param	DESC
env	currently supports two parameters "default" and "GDPR". default: In the default environment, the SDK will find the best access point in the world for access. GDPR: All audio and video data and quality statistics will not pass through servers in mainland China.

Note

If your application has no special requirements, please do not call this interface for setting.

setLicence

setLicence

void setLicence	(Context context
	String url

	String key)

Set SDK authorization license

Try and Purchase a License: https://www.tencentcloud.com/document/product/1071/38546.

Param	DESC
context	
key	the key of licence.
url	the url of licence.

setSocks5Proxy

setSocks5Proxy

void setSocks5Proxy	(String host
	int port
	String username
	String password
	V2TXLiveDef.V2TXLiveSocks5ProxyConfig config)

Set SDK socks5 proxy config

Param	DESC
config	protocol configured with socks5 proxy.
host	socks5 proxy host.
password	socks5 proxy password.
port	socks5 proxy port.
username	socks5 proxy username.

enableAudioCaptureObserver

enableAudioCaptureObserver

void enableAudioCaptureObserver	(boolean enable
	V2TXLiveDef.V2TXLiveAudioFrameObserverFormat format)

Enables/Disables audio capture callback

Param	DESC
enable	`true` : enable; `false` (**default**): disable.
format	audio frame format.

Note

This API works only if you call it before startPush.

enableAudioPlayoutObserver

enableAudioPlayoutObserver

void enableAudioPlayoutObserver	(boolean enable
	V2TXLiveDef.V2TXLiveAudioFrameObserverFormat format)

Enables/Disables audio playout callback

Param	DESC
enable	`true` : enable; `false` (**default**): disable.
format	audio frame format.

enableVoiceEarMonitorObserver

enableVoiceEarMonitorObserver

void enableVoiceEarMonitorObserver	(boolean enable)

Enables/Disables in-ear monitoring callback

Param	DESC

enable	`true` : enable; `false` (**default**): disable.

setUserId

setUserId

void setUserId	(String userId)

Set user id

Param	DESC
userId	User/device id maintained by the service side itself.

callExperimentalAPI

callExperimentalAPI

int callExperimentalAPI	(String jsonStr)

Call experimental APIs

Param	DESC
jsonStr	JSON string describing interface and parameters.

Return Desc:

Return code V2TXLiveCode.

V2TXLIVE_OK: successful.

V2TXLIVE_ERROR_INVALID_PARAMETER: operation failed because of illegal parameter.

TXAudioEffectManager

Last updated：2024-06-06 15:47:57

Module: management class for background music, short audio effects, and voice effects

Description: sets background music, short audio effects, and voice effects

TXAudioEffectManager

TXMusicPreloadObserver

FuncList	DESC
onLoadProgress	Background music preload progress
onLoadError	Background music preload error

TXMusicPlayObserver

FuncList	DESC
onStart	Background music started.
onPlayProgress	Playback progress of background music
onComplete	Background music ended

TXAudioEffectManager

FuncList	DESC
enableVoiceEarMonitor	Enabling in-ear monitoring
setVoiceEarMonitorVolume	Setting in-ear monitoring volume

setVoiceReverbType	Setting voice reverb effects
setVoiceChangerType	Setting voice changing effects
setVoiceCaptureVolume	Setting speech volume
setVoicePitch	Setting speech pitch
setMusicObserver	Setting the background music callback
startPlayMusic	Starting background music
stopPlayMusic	Stopping background music
pausePlayMusic	Pausing background music
resumePlayMusic	Resuming background music
setAllMusicVolume	Setting the local and remote playback volume of background music
setMusicPublishVolume	Setting the remote playback volume of a specific music track
setMusicPlayoutVolume	Setting the local playback volume of a specific music track
setMusicPitch	Adjusting the pitch of background music
setMusicSpeedRate	Changing the speed of background music
getMusicCurrentPosInMS	Getting the playback progress (ms) of background music
getMusicDurationInMS	Getting the total length (ms) of background music
seekMusicToPosInMS	Setting the playback progress (ms) of background music
setMusicScratchSpeedRate	Adjust the speed change effect of the scratch disc
setPreloadObserver	Setting music preload callback
preloadMusic	Preload background music
getMusicTrackCount	Get the number of tracks of background music
setMusicTrack	Specify the playback track of background music

StructType

FuncList	DESC

AudioMusicParam	Background music playback information

EnumType

EnumType	DESC
TXVoiceReverbType	Reverb effects
TXVoiceChangerType	Voice changing effects

onLoadProgress

onLoadProgress

void onLoadProgress	(int id
	int progress)

Background music preload progress

onLoadError

onLoadError

void onLoadError	(int id
	int errorCode)

Background music preload error

Param	DESC
errorCode	-4001: Failed to open the file, such as invalid data found when processing input, ffmpeg protocol not found, etc; -4002: Decoding failure, such as audio file corruption, inaccessible network audio file server, etc; -4003: The number of preloads exceeded the limit， Please call stopPlayMusic first to release the useless preload；-4005: Invalid path, Please check whether the path you passed points to a legal music file；-4006: Invalid URL, Please use a browser to check whether the URL address you passed in can download the desired music file；-4007: No audio stream, Please confirm whether the file you passed is a legal audio file and whether the file is damaged；-4008: Unsupported format, Please confirm whether the

	file format you passed is a supported file format. The mobile version supports [mp3, aac, m4a, wav, ogg, mp4, mkv], and the desktop version supports [mp3, aac, m4a, wav, mp4, mkv].

onStart

onStart

void onStart	(int id
	int errCode)

Background music started.

Called after the background music starts.

Param	DESC
errCode	0: Start playing successfully; -4001: Failed to open the file, such as invalid data found when processing input, ffmpeg protocol not found, etc; -4005: Invalid path, Please check whether the path you passed points to a legal music file；-4006: Invalid URL, Please use a browser to check whether the URL address you passed in can download the desired music file；-4007: No audio stream, Please confirm whether the file you passed is a legal audio file and whether the file is damaged；-4008: Unsupported format, Please confirm whether the file format you passed is a supported file format. The mobile version supports [mp3, aac, m4a, wav, ogg, mp4, mkv], and the desktop version supports [mp3, aac, m4a, wav, mp4, mkv].
id	music ID.

onPlayProgress

onPlayProgress

void onPlayProgress	(int id
	long curPtsMS
	long durationMS)

Playback progress of background music

onComplete

onComplete

void onComplete	(int id
	int errCode)

Background music ended

Called when the background music playback ends or an error occurs.

Param	DESC
errCode	0: End of play; -4002: Decoding failure, such as audio file corruption, inaccessible network audio file server, etc.
id	music ID.

enableVoiceEarMonitor

enableVoiceEarMonitor

void enableVoiceEarMonitor	(boolean enable)

Enabling in-ear monitoring

After enabling in-ear monitoring, anchors can hear in earphones their own voice captured by the mic. This is designed for singing scenarios.

In-ear monitoring cannot be enabled for Bluetooth earphones. This is because Bluetooth earphones have high latency. Please ask anchors to use wired earphones via a UI reminder.
Given that not all phones deliver excellent in-ear monitoring effects, we have blocked this feature on some phones.

Param	DESC
enable	`true:` enable; `false` : disable

Note

In-ear monitoring can be enabled only when earphones are used. Please remind anchors to use wired earphones.

setVoiceEarMonitorVolume

setVoiceEarMonitorVolume

void setVoiceEarMonitorVolume	(int volume)

Setting in-ear monitoring volume

This API is used to set the volume of in-ear monitoring.

Param	DESC
volume	Volume. Value range: 0-100; default: 100

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setVoiceReverbType

setVoiceReverbType

void setVoiceReverbType	(TXVoiceReverbType type)

Setting voice reverb effects

This API is used to set reverb effects for human voice. For the effects supported, please see TXVoiceReverbType.

Note

Effects become invalid after room exit. If you want to use the same effect after you enter the room again, you need to set the effect again using this API.

setVoiceChangerType

setVoiceChangerType

void setVoiceChangerType	(TXVoiceChangerType type)

Setting voice changing effects

This API is used to set voice changing effects. For the effects supported, please see TXVoiceChangeType.

Note

Effects become invalid after room exit. If you want to use the same effect after you enter the room again, you need to set the effect again using this API.

setVoiceCaptureVolume

setVoiceCaptureVolume

void setVoiceCaptureVolume	(int volume)

Setting speech volume

This API is used to set the volume of speech. It is often used together with the music volume setting API setAllMusicVolume to balance between the volume of music and speech.

Param	DESC
volume	Volume. Value range: 0-100; default: 100

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setVoicePitch

setVoicePitch

void setVoicePitch	(double pitch)

Setting speech pitch

This API is used to set the pitch of speech.

Param	DESC
pitch	Ptich, Value range: -1.0f~1.0f; default: 0.0f。

setMusicObserver

setMusicObserver

void setMusicObserver	(int id
	TXMusicPlayObserver observer)

Setting the background music callback

Before playing background music, please use this API to set the music callback, which can inform you of the playback progress.

Param	DESC

musicId	Music ID
observer	For more information, please see the APIs defined in `ITXMusicPlayObserver` .

Note

1. If the ID does not need to be used, the observer can be set to NULL to release it completely.

startPlayMusic

startPlayMusic

boolean startPlayMusic	(final AudioMusicParam musicParam)

Starting background music

You must assign an ID to each music track so that you can start, stop, or set the volume of music tracks by ID.

Param	DESC
musicParam	Music parameter

Note

1. If you play the same music track multiple times, please use the same ID instead of a separate ID for each playback.
2. If you want to play different music tracks at the same time, use different IDs for them.
3. If you use the same ID to play a music track different from the current one, the SDK will stop the current one before playing the new one.

stopPlayMusic

stopPlayMusic

void stopPlayMusic	(int id)

Stopping background music

Param	DESC
id	Music ID

pausePlayMusic

pausePlayMusic

void pausePlayMusic	(int id)

Pausing background music

Param	DESC
id	Music ID

resumePlayMusic

resumePlayMusic

void resumePlayMusic	(int id)

Resuming background music

Param	DESC
id	Music ID

setAllMusicVolume

setAllMusicVolume

void setAllMusicVolume	(int volume)

Setting the local and remote playback volume of background music

This API is used to set the local and remote playback volume of background music.
Local volume: the volume of music heard by anchors
Remote volume: the volume of music heard by audience

Param	DESC
volume	Volume. Value range: 0-100; default: 60

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setMusicPublishVolume

setMusicPublishVolume

void setMusicPublishVolume	(int id
	int volume)

Setting the remote playback volume of a specific music track

This API is used to control the remote playback volume (the volume heard by audience) of a specific music track.

Param	DESC
id	Music ID
volume	Volume. Value range: 0-100; default: 60

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setMusicPlayoutVolume

setMusicPlayoutVolume

void setMusicPlayoutVolume	(int id
	int volume)

Setting the local playback volume of a specific music track

This API is used to control the local playback volume (the volume heard by anchors) of a specific music track.

Param	DESC
id	Music ID
volume	Volume. Value range: 0-100. default: 60

Note

If 100 is still not loud enough for you, you can set the volume to up to 150, but there may be side effects.

setMusicPitch

setMusicPitch

void setMusicPitch	(int id
	float pitch)

Adjusting the pitch of background music

Param	DESC
id	Music ID
pitch	Pitch. Value range: floating point numbers in the range of [-1, 1]; default: 0.0f

setMusicSpeedRate

setMusicSpeedRate

void setMusicSpeedRate	(int id
	float speedRate)

Changing the speed of background music

Param	DESC
id	Music ID
speedRate	Music speed. Value range: floating point numbers in the range of [0.5, 2]; default: 1.0f

getMusicCurrentPosInMS

getMusicCurrentPosInMS

long getMusicCurrentPosInMS	(int id)

Getting the playback progress (ms) of background music

Param	DESC

id	Music ID

Return Desc:

The milliseconds that have passed since playback started. -1 indicates failure to get the the playback progress.

getMusicDurationInMS

getMusicDurationInMS

long getMusicDurationInMS	(String path)

Getting the total length (ms) of background music

Param	DESC
path	Path of the music file.

Return Desc:

The length of the specified music file is returned. -1 indicates failure to get the length.

seekMusicToPosInMS

seekMusicToPosInMS

void seekMusicToPosInMS	(int id
	int pts)

Setting the playback progress (ms) of background music

Param	DESC
id	Music ID
pts	Unit: millisecond

Note

Do not call this API frequently as the music file may be read and written to each time the API is called, which can be time-consuming.

Wait till users finish dragging the progress bar before you call this API.
The progress bar controller on the UI tends to update the progress at a high frequency as users drag the progress bar. This will result in poor user experience unless you limit the frequency.

setMusicScratchSpeedRate

setMusicScratchSpeedRate

void setMusicScratchSpeedRate	(int id
	float scratchSpeedRate)

Adjust the speed change effect of the scratch disc

Param	DESC
id	Music ID
scratchSpeedRate	Scratch disc speed, the default value is 1.0f, the range is: a floating point number between [-12.0 ~ 12.0], the positive/negative speed value indicates the direction is positive/negative, and the absolute value indicates the speed.

Note

Precondition preloadMusic succeeds.

setPreloadObserver

setPreloadObserver

void setPreloadObserver	(TXMusicPreloadObserver observer)

Setting music preload callback

Before preload music, please use this API to set the preload callback, which can inform you of the preload status.

Param	DESC
observer	For more information, please see the APIs defined in `ITXMusicPreloadObserver` .

preloadMusic

preloadMusic

boolean preloadMusic	(final AudioMusicParam preloadParam)

Preload background music

You must assign an ID to each music track so that you can start, stop, or set the volume of music tracks by ID.

Param	DESC
musicParam	Music parameter

Note

1. Preload supports up to 2 preloads with different IDs at the same time, and the preload time does not exceed 10 minutes,you need to stopPlayMusic after use, otherwise the memory will not be released.
2. If the music corresponding to the ID is being played, the preloading fails, and stopPlayMusic must be called first.
3. When the musicParam passed to startPlayMusic is exactly the same, preloading works.

getMusicTrackCount

getMusicTrackCount

int getMusicTrackCount	(int id)

Get the number of tracks of background music

Param	DESC
id	Music ID

setMusicTrack

setMusicTrack

void setMusicTrack	(int id
	int trackIndex)

Specify the playback track of background music

Param	DESC

id	Music ID
index	Specify which track to play (the first track is played by default). Value range [0, total number of tracks).

Note

The total number of tracks can be obtained through the getMusicTrackCount interface.

TXVoiceReverbType

TXVoiceReverbType

Reverb effects

Reverb effects can be applied to human voice. Based on acoustic algorithms, they can mimic voice in different environments. The following effects are supported currently:
0: original; 1: karaoke; 2: room; 3: hall; 4: low and deep; 5: resonant; 6: metal; 7: husky; 8: ethereal; 9: studio; 10: melodious; 11: studio2;

Enum	Value	DESC
TXLiveVoiceReverbType_0	0	disable
TXLiveVoiceReverbType_1	1	KTV
TXLiveVoiceReverbType_2	2	small room
TXLiveVoiceReverbType_3	3	great hall
TXLiveVoiceReverbType_4	4	deep voice
TXLiveVoiceReverbType_5	5	loud voice
TXLiveVoiceReverbType_6	6	metallic sound
TXLiveVoiceReverbType_7	7	magnetic sound
TXLiveVoiceReverbType_8	8	ethereal
TXLiveVoiceReverbType_9	9	studio
TXLiveVoiceReverbType_10	10	melodious
TXLiveVoiceReverbType_11	11	studio2

TXVoiceChangeType

TXVoiceChangeType

Voice changing effects

Voice changing effects can be applied to human voice. Based on acoustic algorithms, they change the tone of voice. The following effects are supported currently:
0: original; 1: child; 2: little girl; 3: middle-aged man; 4: metal; 5: nasal; 6: foreign accent; 7: trapped beast; 8: otaku; 9: electric; 10: robot; 11: ethereal

Enum	Value	DESC
TXLiveVoiceChangerType_0	0	disable
TXLiveVoiceChangerType_1	1	naughty kid
TXLiveVoiceChangerType_2	2	Lolita
TXLiveVoiceChangerType_3	3	uncle
TXLiveVoiceChangerType_4	4	heavy metal
TXLiveVoiceChangerType_5	5	catch cold
TXLiveVoiceChangerType_6	6	foreign accent
TXLiveVoiceChangerType_7	7	caged animal trapped beast
TXLiveVoiceChangerType_8	8	indoorsman
TXLiveVoiceChangerType_9	9	strong current
TXLiveVoiceChangerType_10	10	heavy machinery
TXLiveVoiceChangerType_11	11	intangible

TXAudioMusicParam

TXAudioMusicParam

Background music playback information

The information, including playback ID, file path, and loop times, is passed in the startPlayMusic API.
1. If you play the same music track multiple times, please use the same ID instead of a separate ID for each playback.

2. If you want to play different music tracks at the same time, use different IDs for them.

3. If you use the same ID to play a music track different from the current one, the SDK will stop the current one before playing the new one.

EnumType	DESC
endTimeMS	`Field description:` the point in time in milliseconds for ending music playback. 0 indicates that playback continues till the end of the music track.
id	`Field description:` music ID **Note** the SDK supports playing multiple music tracks. IDs are used to distinguish different music tracks and control their start, end, volume, etc.
isShortFile	`Field description:` whether the music played is a short music track `Valid values:` `true` : short music track that needs to be looped; `false` (default): normal-length music track
loopCount	`Field description:` number of times the music track is looped `Valid values:` 0 or any positive integer. 0 (default) indicates that the music is played once, 1 twice, and so on.
path	`Field description:` absolute path of the music file or url.the mp3,aac,m4a,wav supported.
publish	`Field description:` whether to send the music to remote users `Valid values:` `true` : remote users can hear the music played locally; `false` (default): only the local user can hear the music.
startTimeMS	`Field description:` the point in time in milliseconds for starting music playback

TXBeautyManager

Last updated：2024-06-06 15:47:57

Module: beauty filter and image processing parameter configurations

Function: you can modify parameters such as beautification, filter, and green screen

TXBeautyManager

TXBeautyManager

FuncList	DESC
setBeautyStyle	Sets the beauty (skin smoothing) filter algorithm.
setBeautyLevel	Sets the strength of the beauty filter.
setWhitenessLevel	Sets the strength of the brightening filter.
enableSharpnessEnhancement	Enables clarity enhancement.
setRuddyLevel	Sets the strength of the rosy skin filter.
setFilter	Sets color filter.
setFilterStrength	Sets the strength of color filter.
setGreenScreenFile	Sets green screen video
setEyeScaleLevel	Sets the strength of the eye enlarging filter.
setFaceSlimLevel	Sets the strength of the face slimming filter.
setFaceVLevel	Sets the strength of the chin slimming filter.
setChinLevel	Sets the strength of the chin lengthening/shortening filter.
setFaceShortLevel	Sets the strength of the face shortening filter.
setFaceNarrowLevel	Sets the strength of the face narrowing filter.

setNoseSlimLevel	Sets the strength of the nose slimming filter.
setEyeLightenLevel	Sets the strength of the eye brightening filter.
setToothWhitenLevel	Sets the strength of the teeth whitening filter.
setWrinkleRemoveLevel	Sets the strength of the wrinkle removal filter.
setPounchRemoveLevel	Sets the strength of the eye bag removal filter.
setSmileLinesRemoveLevel	Sets the strength of the smile line removal filter.
setForeheadLevel	Sets the strength of the hairline adjustment filter.
setEyeDistanceLevel	Sets the strength of the eye distance adjustment filter.
setEyeAngleLevel	Sets the strength of the eye corner adjustment filter.
setMouthShapeLevel	Sets the strength of the mouth shape adjustment filter.
setNoseWingLevel	Sets the strength of the nose wing narrowing filter.
setNosePositionLevel	Sets the strength of the nose position adjustment filter.
setLipsThicknessLevel	Sets the strength of the lip thickness adjustment filter.
setFaceBeautyLevel	Sets the strength of the face shape adjustment filter.
setMotionTmpl	Selects the AI animated effect pendant.
setMotionMute	Sets whether to mute during animated effect playback.

EnumType

EnumType	DESC
TXBeautyStyle	Beauty (skin smoothing) filter algorithm

setBeautyStyle

setBeautyStyle

void setBeautyStyle	(int beautyStyle)

Sets the beauty (skin smoothing) filter algorithm.

TRTC has multiple built-in skin smoothing algorithms. You can select the one most suitable for your product needs:

Param	DESC
beautyStyle	Beauty filter style. `TXBeautyStyleSmooth` : smooth; `TXBeautyStyleNature` : natural; `TXBeautyStylePitu` : Pitu

setBeautyLevel

setBeautyLevel

void setBeautyLevel	(float beautyLevel)

Sets the strength of the beauty filter.

Param	DESC
beautyLevel	Strength of the beauty filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

setWhitenessLevel

setWhitenessLevel

void setWhitenessLevel	(float whitenessLevel)

Sets the strength of the brightening filter.

Param	DESC
whitenessLevel	Strength of the brightening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

enableSharpnessEnhancement

enableSharpnessEnhancement

void enableSharpnessEnhancement	(boolean enable)

Enables clarity enhancement.

setRuddyLevel

setRuddyLevel

void setRuddyLevel	(float ruddyLevel)

Sets the strength of the rosy skin filter.

Param	DESC
ruddyLevel	Strength of the rosy skin filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

setFilter

setFilter

void setFilter	(Bitmap image)

Sets color filter.

The color filter is a color lookup table image containing color mapping relationships. You can find several predefined filter images in the official demo we provide.
The SDK performs secondary processing on the original video image captured by the camera according to the mapping relationships in the lookup table to achieve the expected filter effect.

Param	DESC
image	Color lookup table containing color mapping relationships. The image must be in PNG format.

setFilterStrength

setFilterStrength

void setFilterStrength	(float strength)

Sets the strength of color filter.

The larger this value, the more obvious the effect of the color filter, and the greater the color difference between the video image processed by the filter and the original video image.
The default strength is 0.5, and if it is not sufficient, it can be adjusted to a value above 0.5. The maximum value is 1.

Param	DESC
strength	Value range: 0–1. The greater the value, the more obvious the effect. Default value: 0.5

setGreenScreenFile

setGreenScreenFile

int setGreenScreenFile	(String path)

Sets green screen video

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.
The green screen feature enabled by this API is not capable of intelligent keying. It requires that there be a green screen behind the videoed person or object for further chroma keying.

Param	DESC
path	Path of the video file in MP4 format. An empty value indicates to disable the effect.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeScaleLevel

setEyeScaleLevel

int setEyeScaleLevel	(float eyeScaleLevel)

Sets the strength of the eye enlarging filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
eyeScaleLevel	Strength of the eye enlarging filter. Value range: 0–9. `0` indicates to disable the

	filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceSlimLevel

setFaceSlimLevel

int setFaceSlimLevel	(float faceSlimLevel)

Sets the strength of the face slimming filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceSlimLevel	Strength of the face slimming filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceVLevel

setFaceVLevel

int setFaceVLevel	(float faceVLevel)

Sets the strength of the chin slimming filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceVLevel	Strength of the chin slimming filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setChinLevel

setChinLevel

int setChinLevel	(float chinLevel)

Sets the strength of the chin lengthening/shortening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
chinLevel	Strength of the chin lengthening/shortening filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates that the chin is shortened, and a value greater than 0 indicates that the chin is lengthened.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceShortLevel

setFaceShortLevel

int setFaceShortLevel	(float faceShortLevel)

Sets the strength of the face shortening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceShortLevel	Strength of the face shortening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceNarrowLevel

setFaceNarrowLevel

int setFaceNarrowLevel	(float faceNarrowLevel)

Sets the strength of the face narrowing filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
level	Strength of the face narrowing filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setNoseSlimLevel

setNoseSlimLevel

int setNoseSlimLevel	(float noseSlimLevel)

Sets the strength of the nose slimming filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
noseSlimLevel	Strength of the nose slimming filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeLightenLevel

setEyeLightenLevel

int setEyeLightenLevel	(float eyeLightenLevel)

Sets the strength of the eye brightening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
eyeLightenLevel	Strength of the eye brightening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setToothWhitenLevel

setToothWhitenLevel

int setToothWhitenLevel	(float toothWhitenLevel)

Sets the strength of the teeth whitening filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
toothWhitenLevel	Strength of the teeth whitening filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setWrinkleRemoveLevel

setWrinkleRemoveLevel

int setWrinkleRemoveLevel	(float wrinkleRemoveLevel)

Sets the strength of the wrinkle removal filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
wrinkleRemoveLevel	Strength of the wrinkle removal filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setPounchRemoveLevel

setPounchRemoveLevel

int setPounchRemoveLevel	(float pounchRemoveLevel)

Sets the strength of the eye bag removal filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
pounchRemoveLevel	Strength of the eye bag removal filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setSmileLinesRemoveLevel

setSmileLinesRemoveLevel

int setSmileLinesRemoveLevel	(float smileLinesRemoveLevel)

Sets the strength of the smile line removal filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
smileLinesRemoveLevel	Strength of the smile line removal filter. Value range: 0–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setForeheadLevel

setForeheadLevel

int setForeheadLevel	(float foreheadLevel)

Sets the strength of the hairline adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
foreheadLevel	Strength of the hairline adjustment filter. Value range: -9–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeDistanceLevel

setEyeDistanceLevel

int setEyeDistanceLevel	(float eyeDistanceLevel)

Sets the strength of the eye distance adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC

eyeDistanceLevel	Strength of the eye distance adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to widen, and a value greater than 0 indicates to narrow.

Return Desc:

0: Success; -5: feature of license not supported.

setEyeAngleLevel

setEyeAngleLevel

int setEyeAngleLevel	(float eyeAngleLevel)

Sets the strength of the eye corner adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
eyeAngleLevel	Strength of the eye corner adjustment filter. Value range: -9–9. `0` indicates to disable the filter, and `9` indicates the most obvious effect.

Return Desc:

0: Success; -5: feature of license not supported.

setMouthShapeLevel

setMouthShapeLevel

int setMouthShapeLevel	(float mouthShapeLevel)

Sets the strength of the mouth shape adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
mouthShapeLevel	Strength of the mouth shape adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to widen, and a value greater

	than 0 indicates to narrow.

Return Desc:

0: Success; -5: feature of license not supported.

setNoseWingLevel

setNoseWingLevel

int setNoseWingLevel	(float noseWingLevel)

Sets the strength of the nose wing narrowing filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
noseWingLevel	Strength of the nose wing adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to widen, and a value greater than 0 indicates to narrow.

Return Desc:

0: Success; -5: feature of license not supported.

setNosePositionLevel

setNosePositionLevel

int setNosePositionLevel	(float nosePositionLevel)

Sets the strength of the nose position adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
nosePositionLevel	Strength of the nose position adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to lift, and a value greater than 0 indicates to lower.

Return Desc:

0: Success; -5: feature of license not supported.

setLipsThicknessLevel

setLipsThicknessLevel

int setLipsThicknessLevel	(float lipsThicknessLevel)

Sets the strength of the lip thickness adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
lipsThicknessLevel	Strength of the lip thickness adjustment filter. Value range: -9–9. `0` indicates to disable the filter, a value smaller than 0 indicates to thicken, and a value greater than 0 indicates to thin.

Return Desc:

0: Success; -5: feature of license not supported.

setFaceBeautyLevel

setFaceBeautyLevel

int setFaceBeautyLevel	(float faceBeautyLevel)

Sets the strength of the face shape adjustment filter.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
faceBeautyLevel	Strength of the face shape adjustment filter. Value range: 0–9. `0` indicates to disable the filter, and the greater the value, the more obvious the effect.

Return Desc:

0: Success; -5: feature of license not supported.

setMotionTmpl

setMotionTmpl

void setMotionTmpl	(String tmplPath)

Selects the AI animated effect pendant.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect.

Param	DESC
tmplPath	Directory of the animated effect material file

setMotionMute

setMotionMute

void setMotionMute	(boolean motionMute)

Sets whether to mute during animated effect playback.

This interface is only available in the enterprise version SDK (the old version has been offline, if you need to use the advanced beauty function in the new version SDK, please refer to Tencent Beauty Effect SDK) in effect. Some animated effects have audio effects, which can be disabled through this API when they are played back.

Param	DESC
motionMute	`true` : mute; `false` : unmute

TXBeautyStyle

TXBeautyStyle

Beauty (skin smoothing) filter algorithm

TRTC has multiple built-in skin smoothing algorithms. You can select the one most suitable for your product needs.

Enum	Value	DESC
TXBeautyStyleSmooth	0	Smooth style, which uses a more radical algorithm for more obvious effect and is suitable for show live streaming.

TXBeautyStyleNature	1	Natural style, which retains more facial details for more natural effect and is suitable for most live streaming use cases.
TXBeautyStylePitu	2	Pitu style, which is provided by YouTu Lab. Its skin smoothing effect is between the smooth style and the natural style, that is, it retains more skin details than the smooth style and has a higher skin smoothing degree than the natural style.

TXDeviceManager

Last updated：2024-06-06 15:47:57

Module: audio/video device management module

Description: manages audio/video devices such as camera, mic, and speaker.

TXDeviceManager

TXDeviceManager

FuncList	DESC
isFrontCamera	Querying whether the front camera is being used
switchCamera	Switching to the front/rear camera (for mobile OS)
getCameraZoomMaxRatio	Getting the maximum zoom ratio of the camera (for mobile OS)
setCameraZoomRatio	Setting the camera zoom ratio (for mobile OS)
isAutoFocusEnabled	Querying whether automatic face detection is supported (for mobile OS)
enableCameraAutoFocus	Enabling auto focus (for mobile OS)
setCameraFocusPosition	Adjusting the focus (for mobile OS)
enableCameraTorch	Enabling/Disabling flash, i.e., the torch mode (for mobile OS)
setAudioRoute	Setting the audio route (for mobile OS)
setExposureCompensation	Set the exposure parameters of the camera, ranging from - 1 to 1
setCameraCapturerParam	Set camera acquisition preferences
setSystemVolumeType	Setting the system volume type (for mobile OS)

StructType

FuncList	DESC
TXCameraCaptureParam	Camera acquisition parameters

EnumType

EnumType	DESC
TXSystemVolumeType	System volume type
TXAudioRoute	Audio route (the route via which audio is played)
TXCameraCaptureMode	Camera acquisition preferences

isFrontCamera

isFrontCamera

Querying whether the front camera is being used

switchCamera

switchCamera

int switchCamera	(boolean frontCamera)

Switching to the front/rear camera (for mobile OS)

getCameraZoomMaxRatio

getCameraZoomMaxRatio

Getting the maximum zoom ratio of the camera (for mobile OS)

setCameraZoomRatio

setCameraZoomRatio

int setCameraZoomRatio	(float zoomRatio)

Setting the camera zoom ratio (for mobile OS)

Param	DESC
zoomRatio	Value range: 1-5. 1 indicates the widest angle of view (original), and 5 the narrowest angle of view (zoomed in).The maximum value is recommended to be 5. If the value exceeds 5, the video will become blurred.

isAutoFocusEnabled

isAutoFocusEnabled

Querying whether automatic face detection is supported (for mobile OS)

enableCameraAutoFocus

enableCameraAutoFocus

int enableCameraAutoFocus	(boolean enabled)

Enabling auto focus (for mobile OS)

After auto focus is enabled, the camera will automatically detect and always focus on faces.

setCameraFocusPosition

setCameraFocusPosition

int setCameraFocusPosition	(int x
	int y)

Adjusting the focus (for mobile OS)

This API can be used to achieve the following:

1. A user can tap on the camera preview.

2. A rectangle will appear where the user taps, indicating the spot the camera will focus on.

3. The user passes the coordinates of the spot to the SDK using this API, and the SDK will instruct the camera to focus as required.

Param	DESC
position	The spot to focus on. Pass in the coordinates of the spot you want to focus on.

Note

Before using this API, you must first disable auto focus using enableCameraAutoFocus.

Return Desc:

0: operation successful; negative number: operation failed.

enableCameraTorch

enableCameraTorch

boolean enableCameraTorch	(boolean enable)

Enabling/Disabling flash, i.e., the torch mode (for mobile OS)

setAudioRoute

setAudioRoute

int setAudioRoute	(TXAudioRoute route)

Setting the audio route (for mobile OS)

A mobile phone has two audio playback devices: the receiver at the top and the speaker at the bottom.
If the audio route is set to the receiver, the volume is relatively low, and audio can be heard only when the phone is put near the ear. This mode has a high level of privacy and is suitable for answering calls.
If the audio route is set to the speaker, the volume is relatively high, and there is no need to put the phone near the ear. This mode enables the "hands-free" feature.

setExposureCompensation

setExposureCompensation

int setExposureCompensation	(float value)

Set the exposure parameters of the camera, ranging from - 1 to 1

setCameraCapturerParam

setCameraCapturerParam

void setCameraCapturerParam	(TXCameraCaptureParam params)

Set camera acquisition preferences

setSystemVolumeType

setSystemVolumeType

int setSystemVolumeType	(TXSystemVolumeType type)

Setting the system volume type (for mobile OS)

@deprecated This API is not recommended after v9.5. Please use the `startLocalAudio(quality)` API in `TRTCCloud` instead, which param `quality` is used to decide audio quality.

TXSystemVolumeType(Deprecated)

TXSystemVolumeType(Deprecated)

System volume type

Enum	Value	DESC
TXSystemVolumeTypeAuto	Not Defined	Auto
TXSystemVolumeTypeMedia	Not Defined	Media volume
TXSystemVolumeTypeVOIP	Not Defined	Call volume

TXAudioRoute

TXAudioRoute

Audio route (the route via which audio is played)

Audio route is the route (speaker or receiver) via which audio is played. It applies only to mobile devices such as mobile phones.
A mobile phone has two speakers: one at the top (receiver) and the other the bottom.
If the audio route is set to the receiver, the volume is relatively low, and audio can be heard only when the phone is put near the ear. This mode has a high level of privacy and is suitable for answering calls.
If the audio route is set to the speaker, the volume is relatively high, and there is no need to put the phone near the ear. This mode enables the "hands-free" feature.

Enum	Value	DESC
TXAudioRouteSpeakerphone	Not Defined	Speakerphone: the speaker at the bottom is used for playback (hands-free). With relatively high volume, it is used to play music out loud.
TXAudioRouteEarpiece	Not Defined	Earpiece: the receiver at the top is used for playback. With relatively low volume, it is suitable for call scenarios that require privacy.

TXCameraCaptureMode

TXCameraCaptureMode

Camera acquisition preferences

This enum is used to set camera acquisition parameters.

Enum	Value	DESC
TXCameraResolutionStrategyAuto	Not Defined	Auto adjustment of camera capture parameters. SDK selects the appropriate camera output parameters according to the actual acquisition device performance and network situation, and maintains a balance between device performance and video preview quality.
TXCameraResolutionStrategyPerformance	Not	Give priority to equipment performance.

	Defined	SDK selects the closest camera output parameters according to the user's encoder resolution and frame rate, so as to ensure the performance of the device.
TXCameraResolutionStrategyHighQuality	Not Defined	Give priority to the quality of video preview. SDK selects higher camera output parameters to improve the quality of preview video. In this case, it will consume more CPU and memory to do video preprocessing.
TXCameraCaptureManual	Not Defined	Allows the user to set the width and height of the video captured by the local camera.

TXCameraCaptureParam

TXCameraCaptureParam

Camera acquisition parameters

This setting determines the quality of the local preview image.

EnumType	DESC
height	Field description: height of acquired image
mode	Field description: camera acquisition preferences,please see TXCameraCaptureMode
width	Field description: width of acquired image

ErrorCode

Last updated：2024-06-06 15:47:57

Module: V2TXLiveCode @ TXLiteAVSDK

Function: Definitions of error codes and warning codes of Tencent Cloud LVB

ErrorCode

EnumType

EnumType	DESC
V2TXLiveCode	V2 Error codes and warning codes

V2TXLiveCode

V2TXLiveCode

V2 Error codes and warning codes

Enum	Value	DESC
V2TXLIVE_OK	0	No error.
V2TXLIVE_ERROR_FAILED	-1	Unclassified error.
V2TXLIVE_ERROR_INVALID_PARAMETER	-2	An invalid parameter was input during the API call.
V2TXLIVE_ERROR_REFUSED	-3	The API call was rejected.
V2TXLIVE_ERROR_NOT_SUPPORTED	-4	The API is currently not suppoted.
V2TXLIVE_ERROR_INVALID_LICENSE	-5	Failed to call the API because the license was invalid.

V2TXLIVE_ERROR_REQUEST_TIMEOUT	-6	The server request timed out.
V2TXLIVE_ERROR_SERVER_PROCESS_FAILED	-7	The server cannot process the request.
V2TXLIVE_ERROR_DISCONNECTED	-8	Disconnect.
V2TXLIVE_ERROR_NO_AVAILABLE_HEVC_DECODERS	-2304	could not find available hevc decoder.
V2TXLIVE_WARNING_NETWORK_BUSY	1101	Data upload was jammed because the upstream bandwidth was too low.
V2TXLIVE_WARNING_VIDEO_BLOCK	2105	Blocking occurred during video playback.
V2TXLIVE_WARNING_CAMERA_START_FAILED	-1301	Failed to start the camera.
V2TXLIVE_WARNING_CAMERA_OCCUPIED	-1316	The camera is being occupied.
V2TXLIVE_WARNING_CAMERA_NO_PERMISSION	-1314	The camera is not authorized. This warning usually occurs on mobile devices due to the camera permission is denied by the user.
V2TXLIVE_WARNING_MICROPHONE_START_FAILED	-1302	Failed to enable the mic.
V2TXLIVE_WARNING_MICROPHONE_OCCUPIED	-1319	The mic is being used. If a call is in progress on the mobile device, the mic cannot be enabled.
V2TXLIVE_WARNING_MICROPHONE_NO_PERMISSION	-1317	The mic is not authorized. This warning usually occurs on mobile devices due to the mic permission is denied by the user.
V2TXLIVE_WARNING_SCREEN_CAPTURE_NOT_SUPPORTED	-1309	Screen capture is not supported in curent system.
V2TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	-1308	Failed to enable the screen

		capture.
V2TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	-7001	Screen capture is interrupted by system.
V2TXLIVE_WARNING_CURRENT_ENCODE_TYPE_CHANGED	1104	The codec changed. The additional field `codec_type` in `onWarning` indicates the codec currently in use. `1` indicates H.265, and `0` indicates H.264. This field is not supported on Windows.
V2TXLIVE_WARNING_CURRENT_DECODE_TYPE_CHANGED	2008	The codec changed. The additional field `codec_type` in `onWarning` indicates the codec currently in use. `1` indicates H.265, and `0` indicates H.264. This field is not supported on Windows.

Property Definition

Last updated：2024-06-06 15:47:57

Module: V2TXLiveProperty @ TXLiteAVSDK

Function: Keys supported by V2TXLive setProperty

Property Definition

StructType

FuncList	DESC
V2TXLiveProperty	V2 live property

V2TXLiveProperty

V2TXLiveProperty

V2 live property

EnumType	DESC
kV2ClearLastImage	Enable/Disable clear the last image. Default value: true. Value: true/false.
kV2EnableHardwareAcceleration	Enable/Disable hardware acceleration[RTMP, Player]. Default value: true. Value: true/false.
kV2EnableHevcEncode	Enable/Disable Hevc Encode[RTMP/RTC, Pusher]. Default value: false. Value: true/false.
kV2EnableIPMultiplexing	Enable/Disable IP Multiplexing[FLV, Player]. Default value: false. Value: true/false.

kV2MaxNumberOfReconnection	Set the number of reconnections[RTMP, Player]. Default value: 3. Value: int.
kV2SecondsBetweenReconnection	Set reconnection interval[RTMP, Player]. Unit: second. Default value: 3. Value: int.

kV2SetHeaders	Set play request headers[FLV， Player].
	Value：JSON string.
	Example:
	{
	"headers": [
	{
	"key": "key1",
	"value": "value1"
	},
	{
	"key": "key2",
	"value": "value2"
	}
	]
	}

kV2SetMetaData	Set Push Meta Info[RTMP, Pusher].
	Value：JSON string.
	Example:
	{
	"metadata": [
	{
	"key": "key1",
	"value": "value1"
	},
	{
	"key": "key2",
	"value": "value2"
	}
	]
	}

kV2SetVideoQualityEx	Set custom encoding parameters[RTMP/RTC， Pusher]. Value: JSON string. Example: {

	"videoWidth":360, "videoHeight":640, "videoFps":15, "videoBitrate":1000, "minVideoBitrate":1000 }

Type Definition

Last updated：2024-06-06 15:47:57

Module: V2TXLiveDef @ TXLiteAVSDK

Function: Key type definitions for Tencent Cloud LVB

Type Definition

StructType

FuncList	DESC
V2TXLiveVideoEncoderParam	Video encoding parameters
V2TXLiveVideoFrame	Video frame information
V2TXLiveAudioFrameObserverFormat	audio callback format
V2TXLivePusherStatistics	Pusher statistics
V2TXLivePlayerStatistics	Player statistics
V2TXLiveMixStream	Position of each subimage in On-Cloud MixTranscoding
V2TXLiveTranscodingConfig	Configure On-Cloud MixTranscoding
V2TXLiveLocalRecordingParams	Configure On-LocalRecording
V2TXLiveSocks5ProxyConfig	Protocol configured with socks5 proxy.
V2TXLiveLogConfig	Log configuration
V2TXLiveStreamInfo	Stream information supporting adaptive handover.

EnumType

EnumType	DESC

V2TXLiveMode	Supported protocol, RTMP is not supported on Windows or macOS.
V2TXLiveVideoResolution	Video resolution.
V2TXLiveVideoResolutionMode	Video aspect ratio mode
V2TXLiveMirrorType	Local camera mirror type.
V2TXLiveFillMode	Image fill mode
V2TXLiveRotation	Clockwise rotation of the video image
V2TXLivePixelFormat	Pixel format of video frames
V2TXLiveBufferType	Video data container format
V2TXLiveTexture	Video texture container
V2TXLiveAudioQuality	Audio quality
V2TXLiveAudioFrameOperationMode	Audio callback data operation mode
V2TXLivePushStatus	Livestream connection status
V2TXLiveMixInputType	Specify the type of streams to mix
V2TXLiveRecordMode	Recording audio and video mode
V2TXLiveLogLevel	Log level

V2TXLiveMode

V2TXLiveMode

Supported protocol, RTMP is not supported on Windows or macOS.

Enum	Value	DESC
TXLiveMode_RTMP	Not Defined	RTMP protocol.
TXLiveMode_RTC	Not Defined	TRTC protocol.

V2TXLiveVideoResolution

V2TXLiveVideoResolution

Video resolution.

Enum	Value	DESC
V2TXLiveVideoResolution160x160	Not Defined	Resolution: 160×160. Bitrate range: 100 Kbps to 150 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution270x270	Not Defined	Resolution: 270×270. Bitrate range: 200 Kbps to 300 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution480x480	Not Defined	Resolution: 480×480. Bitrate range: 350 Kbps to 525 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution320x240	Not Defined	Resolution: 320×240. Bitrate range: 250 Kbps to 375 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution480x360	Not Defined	Resolution: 480×360. Bitrate range: 400 Kbps to 600 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution640x480	Not Defined	Resolution: 640×480. Bitrate range: 600 Kbps to 900 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution320x180	Not Defined	Resolution: 320×180. Bitrate range: 250 Kbps to 400 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution480x270	Not Defined	Resolution: 480×270. Bitrate range: 350 Kbps to 550 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution640x360	Not Defined	Resolution: 640×360. Bitrate range: 500 Kbps to 900 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution960x540	Not Defined	Resolution: 960×540. Bitrate range: 800 Kbps to 1500 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution1280x720	Not Defined	Resolution: 1280×720. Bitrate range: 1000 Kbps to 1800 Kbps. Frame rate: 15 fps.
V2TXLiveVideoResolution1920x1080	Not Defined	Resolution: 1920×1080. Bitrate range: 2500 Kbps to 3000 Kbps. Frame rate: 15 fps.

V2TXLiveVideoResolutionMode

V2TXLiveVideoResolutionMode

Video aspect ratio mode

Note

Landscape resolution: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModeLandscape = 640 × 360.

Portrait resolution: V2TXLiveVideoResolution640x360 + V2TXLiveVideoResolutionModePortrait = 360 × 640.

Enum	Value	DESC
V2TXLiveVideoResolutionModeLandscape	Not Defined	Landscape resolution.
V2TXLiveVideoResolutionModePortrait	Not Defined	Portrait resolution.

V2TXLiveMirrorType

V2TXLiveMirrorType

Local camera mirror type.

Enum	Value	DESC
V2TXLiveMirrorTypeAuto	Not Defined	Default mirror type. Images from the front camera are mirrored, and images from the rear camera are not mirrored.
V2TXLiveMirrorTypeEnable	Not Defined	Both the front and rear cameras are switched to the mirror mode.
V2TXLiveMirrorTypeDisable	Not Defined	Both the front and rear cameras are switched to the non-mirror mode.

V2TXLiveFillMode

V2TXLiveFillMode

Image fill mode

Enum	Value	DESC
V2TXLiveFillModeFill	Not Defined	The entire screen is covered by the image, without black edges. If the aspect ratio of the image is different from

		that of the screen, part of the image will be cropped.
V2TXLiveFillModeFit	Not Defined	The image adapts to the screen and is not cropped. If the aspect ratio of the image is different from that of the screen, black edges will appear.
V2TXLiveFillModeScaleFill	Not Defined	The screen is entirely covered by the image. The image will be stretched if screen and image have different aspect ratios.

V2TXLiveRotation

V2TXLiveRotation

Clockwise rotation of the video image

Enum	Value	DESC
V2TXLiveRotation0	Not Defined	No rotation.
V2TXLiveRotation90	Not Defined	Rotate 90 degrees clockwise.
V2TXLiveRotation180	Not Defined	Rotate 180 degrees clockwise.
V2TXLiveRotation270	Not Defined	Rotate 270 degrees clockwise.

V2TXLivePixelFormat

V2TXLivePixelFormat

Pixel format of video frames

Enum	Value	DESC
V2TXLivePixelFormatUnknown	Not Defined	Unknown.
V2TXLivePixelFormatI420	Not Defined	YUV420P I420.

V2TXLivePixelFormatTexture2D	Not Defined	Texture2D.

V2TXLiveBufferType

V2TXLiveBufferType

Video data container format

Note

In the custom capture and rendering features, you need to use the following enumerated values to specify the format for containing video data.

Texture: this is most efficient when used directly.

Enum	Value	DESC
V2TXLiveBufferTypeUnknown	Not Defined	Unknown.
V2TXLiveBufferTypeByteBuffer	Not Defined	ByteBuffer.
V2TXLiveBufferTypeByteArray	Not Defined	ByteArray.
V2TXLiveBufferTypeTexture	Not Defined	Texture.

V2TXLiveTexture

V2TXLiveTexture

Video texture container

Enum	Value	DESC
public int textureId	Not Defined	the ID of Texture2D.
public javax.microedition.khronos.egl.EGLContext eglContext10	Not Defined	EGL Context.
public android.opengl.EGLContext eglContext14	Not	EGL Context.

	Defined	

V2TXLiveAudioQuality

V2TXLiveAudioQuality

Audio quality

Enum	Value	DESC
V2TXLiveAudioQualitySpeech	Not Defined	Audio: 16k sample rate, mono-channel, 16 Kbps audio raw bitrate. This quality is suitable for scenarios that mainly involve voice calls, such as online meetings and voice calls.
V2TXLiveAudioQualityDefault	Not Defined	General: 48k sample rate, mono-channel, 50 Kbps audio raw bitrate. This quality is the default audio quality of the SDK. We recommend that you choose this option unless you have special requirements.
V2TXLiveAudioQualityMusic	Not Defined	Music: 48k sample rate, dual-channel + full-band, 128 Kbps audio raw bitrate. This quality is suitable for scenarios that require Hi-Fi music transmission, such as karaoke and music livestreams.

V2TXLiveAudioFrameOperationMode

V2TXLiveAudioFrameOperationMode

Audio callback data operation mode

SDK provides two modes of operation for audio callback data.

Read-only mode (ReadOnly): Get audio data only from the callback.

ReadWrite mode (ReadWrite): You can get and modify the audio data of the callback.

Enum	Value	DESC
V2TXLiveAudioFrameOperationModeReadWrite	Not Defined	Read-write mode: You can get and modify the audio data of the callback, the default mode.
V2TXLiveAudioFrameOperationModeReadOnly	Not Defined	Read-only mode: Get audio data from callback only.

V2TXLivePushStatus

V2TXLivePushStatus

Livestream connection status

Enum	Value	DESC
V2TXLivePushStatusDisconnected	Not Defined	Disconnected from the server.
V2TXLivePushStatusConnecting	Not Defined	Connecting to the server.
V2TXLivePushStatusConnectSuccess	Not Defined	Connected to the server successfully.
V2TXLivePushStatusReconnecting	Not Defined	Reconnecting to the server.

V2TXLiveMixInputType

V2TXLiveMixInputType

Specify the type of streams to mix

Enum	Value	DESC
V2TXLiveMixInputTypeAudioVideo	Not Defined	Audio and video.
V2TXLiveMixInputTypePureVideo	Not Defined	Video only.
V2TXLiveMixInputTypePureAudio	Not Defined	Audio only.

V2TXLiveRecordMode

V2TXLiveRecordMode

Recording audio and video mode

Enum	Value	DESC
V2TXLiveRecordModeBoth	Not Defined	Both mode: Recording audio and video

V2TXLiveLogLevel

V2TXLiveLogLevel

Log level

Enum	Value	DESC
V2TXLiveLogLevelAll	0	Output all levels of log.
V2TXLiveLogLevelDebug	1	Output DEBUG, INFO, WARNING, ERROR and FATAL level log.
V2TXLiveLogLevelInfo	2	Output INFO, WARNING, ERROR and FATAL level log.
V2TXLiveLogLevelWarning	3	Output WARNING, ERROR and FATAL level log.
V2TXLiveLogLevelError	4	Output ERROR and FATAL level log.
V2TXLiveLogLevelFatal	5	Only output FATAL level log.
V2TXLiveLogLevelNULL	6	Does not output any sdk log.

V2TXLiveVideoEncoderParam

V2TXLiveVideoEncoderParam

Video encoding parameters

These settings determine the quality of image viewed by remote users.

EnumType	DESC
minVideoBitrate	`Field description:` minimum video bitrate. The SDK will reduce the bitrate to as low as the value specified by `minVideoBitrate` to ensure the smoothness only if the network conditions are poor. `Recommended value:` you can set the `videoBitrate` and `minVideoBitrate` parameters at the same time to restrict the SDK's adjustment range of the video bitrate:

	If you set `videoBitrate` and `minVideoBitrate` to the same value, it is equivalent to disabling the adaptive adjustment capability of the SDK for the video bitrate.
videoBitrate	`Field description:` target video bitrate. The SDK encodes streams at the target video bitrate and will actively reduce the bitrate only in weak network environments. `Recommended value:` please see the optimal bitrate for each specification in `V2TXLiveVideoResolution`. You can also slightly increase the optimal bitrate. For example, `V2TXLiveVideoResolution1280x720` corresponds to the target bitrate of 1,200 Kbps. You can also set the bitrate to 1,500 Kbps for higher definition. **Note** you can set the `videoBitrate` and `minVideoBitrate` parameters at the same time to restrict the SDK's adjustment range of the video bitrate: If you set `videoBitrate` and `minVideoBitrate` to the same value, it is equivalent to disabling the adaptive adjustment capability of the SDK for the video bitrate.
videoFps	`Field description:` video capturing frame rate. `Recommended value:` 15 or 20 fps. If the frame rate is lower than 5 fps, there will be obvious lagging; if lower than 10 fps but higher than 5 fps, there will be slight lagging; if higher than 20 fps, the bandwidth will be wasted (the frame rate of movies is generally 24 fps).
videoResolution	`Field description:` video resolution. `Recommended value:` For desktop platforms (Windows and macOS), we recommend you select a resolution of 640x360 or above and select `Landscape` (landscape resolution) for `videoResolutionMode`. **Note** to use a portrait resolution, please specify `videoResolutionMode` as `Portrait`; for example, when used together with `Portrait`, 640x360 represents 360x640.
videoResolutionMode	`Field description:` resolution mode (landscape/portrait). `Recommended value:` for desktop platforms (Windows and macOS), `Landscape` is recommended. **Note** to use a portrait resolution, please specify `videoResolutionMode` as `Portrait`; for example, when used together with `Portrait`, 640x360 represents 360x640.

V2TXLiveVideoFrame

V2TXLiveVideoFrame

Video frame information

Note

Used during custom capture and rendering. During custom capture, you need to use V2TXLiveVideoFrame to contain the video frame to be sent. During custom rendering, the video frame contained by V2TXLiveVideoFrame will be returned.

EnumType	DESC
buffer	`Field description:` Video data.
bufferType	`Field description:` Video data container format.
data	`Field description:` Video data.
height	`Field description:` Video height.
pixelFormat	`Field description:` Video pixel format.
rotation	`Field description:` Clockwise rotation angle of video frames.
texture	`Field description:` Video texture container.
width	`Field description:` Video width.

V2TXLiveAudioFrameObserverFormat

V2TXLiveAudioFrameObserverFormat

audio callback format

EnumType	DESC
channel	`Field description:` number of sound channels. `Recommended value:` default value: 1, which means mono channel. Valid values: 1: mono channel; 2: dual channel.
mode	`Field description:` audio callback data operation mode. `Recommended value:` V2TXLiveAudioFrameOperationModeReadOnly, get audio data from callback only. The modes that can be set are

	V2TXLiveAudioFrameOperationModeReadOnly, V2TXLiveAudioFrameOperationModeReadWrite.
sampleRate	`Field description:` sample rate. `Recommended value:` default value: 48000 Hz. Valid values: 16000, 32000, 44100, 48000.
samplesPerCall	`Field description:` number of sample points. `Recommended value:` the value must be an integer multiple of sampleRate/100.

V2TXLivePusherStatistics

V2TXLivePusherStatistics

Pusher statistics

EnumType	DESC
appCpu	`Field description:` CPU utilization of the current app (%).
audioBitrate	`Field description:` Audio bitrate (Kbps).
fps	`Field description:` Frame rate (fps).
height	`Field description:` Video height.
netSpeed	`Field description:` upload speed (Kbps).
rtt	`Field description:` Round-trip delay (ms) from the SDK to cloud.
systemCpu	`Field description:` CPU utilization of the current system (%).
videoBitrate	`Field description:` Video bitrate (Kbps).
width	`Field description:` Video width.

V2TXLivePlayerStatistics

V2TXLivePlayerStatistics

Player statistics

EnumType	DESC
appCpu	`Field description:` CPU utilization of the current app (%).
audioBitrate	`Field description:` Audio bitrate (Kbps).
audioBlockRate	`Field description` : Audio playback lag rate (%). Audio playback lag rate (audioBlockRate) = cumulative audio playback lag duration (audioTotalBlockTime)/audio playback interval duration (2000ms).
audioPacketLoss	`Field description` : Total packet loss rate (%) of the audio/video stream. Note: Only playback address prefixed with [trtc://] or [webrtc://] are supported.
audioTotalBlockTime	`Field description` : Cumulative audio playback lag duration (ms). The duration is the block duration within 2s.
fps	`Field description:` Frame rate (fps).
height	`Field description:` Video height.
jitterBufferDelay	`Field description` : Playback delay (ms).
netSpeed	`Field description:` donwload speed (Kbps).
rtt	`Field description:` Round-trip delay (ms) from the SDK to cloud. Note: Only playback address prefixed with [trtc://] or [webrtc://] are supported.
systemCpu	`Field description:` CPU utilization of the current system (%).
videoBitrate	`Field description:` Video bitrate (Kbps).
videoBlockRate	`Field description` : Video playback lag rate (%). Video playback lag rate (videoBlockRate) = cumulative video playback lag duration (videoTotalBlockTime)/video playback interval duration (2000ms).
videoPacketLoss	`Field description` : Total packet loss rate (%) of the audio/video stream. Note: Only playback address prefixed with [trtc://] or [webrtc://] are supported.
videoTotalBlockTime	`Field description` : Cumulative video playback lag duration (ms). The duration is the block duration within 2s.
width	`Field description:` Video width.

V2TXLiveMixStream

V2TXLiveMixStream

Position of each subimage in On-Cloud MixTranscoding

EnumType	DESC
height	`Field description:` height (absolute pixels) of the image layer.
inputType	`Field description:` input type of the live stream.
streamId	`Field description:` push `streamId` of users whose streams are mixed. `nil` indicates the current push `streamId`.
userId	`Field description:` `userId` of users whose streams are mixed.
width	`Field description:` width (absolute pixels) of the image layer.
x	`Field description:` x-axis (absolute pixels) of the image layer.
y	`Field description:` y-axis (absolute pixels) of the image layer.
zOrder	`Field description:` layer number (1-15), which must be unique.

V2TXLiveTranscodingConfig

V2TXLiveTranscodingConfig

Configure On-Cloud MixTranscoding

EnumType	DESC
audioBitrate	`Field description:` audio bitrate of the transcoded stream. `Value range:` [32,192]; default value: 64 (Kbps).
audioChannels	`Field description:` number of sound channels of the transcoded stream. `Valid values:` 1 (default), 2.
audioSampleRate	`Field description:` audio sample rate of the transcoded stream. `Valid values:` 12000 Hz, 16000 Hz, 22050 Hz, 24000 Hz, 32000 Hz, 44100 Hz, 48000 Hz (default).
backgroundColor	`Field description:` background color of the mixed video image. The default color is black, and the value is a hex number. For example: "0x61B9F1" represents the RGB color (97,158,241). **Default value:** 0x000000 (black)

backgroundImage	`Field description:` background image of the mixed video. **Default value:** `nil` , which means that no background image is set. **Note** you need to first upload the image in **Application Management** > **Function Configuration** > **Material Management** in the console. You will get an image ID for the image uploaded, which you need to convert to a string and use it as the value of `backgroundImage` . For example, if the image ID is 63, you should set `backgroundImage` to `63` .
mixStreams	`Field description:` position of each channel of subimage.
outputStreamId	`Field description:` ID of the live stream pushed to CDN. If you do not set this parameter, the SDK will execute the default logic, that is, it will mix multiple streams in the room into the video stream of the API caller, i.e., A + B => A. If you set this parameter, the SDK will mix multiple streams in the room into the live stream whose ID you have specified, i.e., A + B => C. `Default value` : `nil` , which indicates that multiple streams in the room are mixed into the video stream of the API caller.
videoBitrate	`Field description:` bitrate (Kbps) for the resolution of the transcoded video. `Recommended value:` if you set it to 0, the backend will calculate a bitrate based on `videoWidth` and `videoHeight` . You can also refer to the remarks for the enumerated value `V2TXLiveVideoResolution` .
videoFramerate	`Field description:` frame rate (fps) for the resolution of the transcoded video. `Value range:` (0,30]; default: 15.
videoGOP	`Field description:` keyframe interval (GOP) for the resolution of the transcoded video. `Value range:` [1,8]; default value: 2 (sec).
videoHeight	`Field description:` height of transcoded video. `Recommended value:` 640 px. If audio-only streams are mixed, the mixing result will carry a video stream that shows a canvas background. To avoid this, set both the width and height to 0 px.
videoWidth	`Field description:` width of transcoded video. `Recommended value:` 360 px. If audio-only streams are mixed, the mixing result will carry a video stream that shows a canvas background. To avoid this, set both the width and height to 0 px.

V2TXLiveLocalRecordingParams

V2TXLiveLocalRecordingParams

Configure On-LocalRecording

EnumType	DESC
filePath	`Field description:` The path of the recorded file (required), please ensure that the path has read and write permissions and is legal, otherwise the recorded file cannot be generated. `Recommended value:` "yourpath/record/test.mp4". The path needs to be accurate to the file name and format suffix. The format suffix is used to determine the recorded file format. The currently supported format is only MP4.
interval	`Field description:` interval Recording information update frequency (optional), in milliseconds, valid range: 1000-10000. `Default value` : `-1` , which means no callback.
recordMode	`Field description:` Media recording mode. `Default value` : `V2TXLiveRecordModeBoth` , which means recording audio and video at the same time.

V2TXLiveSocks5ProxyConfig

V2TXLiveSocks5ProxyConfig

Protocol configured with socks5 proxy.

EnumType	DESC
supportHttps	`Field description:` Indicates whether HTTPS is supported. `Recommended value:` Default value: true.
supportTcp	`Field description:` Indicates whether TCP is supported. `Recommended value:` Default value: true.
supportUdp	`Field description:` Indicates whether UDP is supported. `Recommended value:` Default value: true.

V2TXLiveLogConfig

V2TXLiveLogConfig

Log configuration

EnumType	DESC
enableConsole	`Field description:` Whether to allow the SDK to print Log on the console of the editor (XCoder, Android Studio, Visual Studio, etc.). `Recommended value:` Default value: false.
enableLogFile	`Field description:` Whether to enable local log file. `Special Instructions:` If not for special needs, please do not close the local log file, otherwise the Tencent Cloud technical team will not be able to track and locate problems when they occur. `Recommended value:` Default value: true.
enableObserver	`Field description:` Whether to receive the log information to be printed through V2TXLivePremierObserver. `Special Instructions:` If you want to implement Log writing by yourself, you can turn on this switch, Log information will be called back to you V2TXLivePremierObserver#onLog. `Recommended value:` Default value: false.
logLevel	`Field description:` Set Log level. `Recommended value:` Default value: V2TXLiveLogLevel.V2TXLiveLogLevelAll.
logPath	`Field description:` Set the storage directory of the local log, default Log storage location: Android：/sdcard/Android/data/your packagename/files/log/liteav/.

V2TXLiveStreamInfo

V2TXLiveStreamInfo

Stream information supporting adaptive handover.

EnumType	DESC
bitrate	`Field description:` Bitrate in bps, default value: 0, means unknown.
framerate	`Field description:` Framerate, default value: 0, means unknown.
height	`Field description:` Video height, default value: 0, means unknown.

url	`Field description:` Stream url.
width	`Field description:` Video width, default value: 0, means unknown.