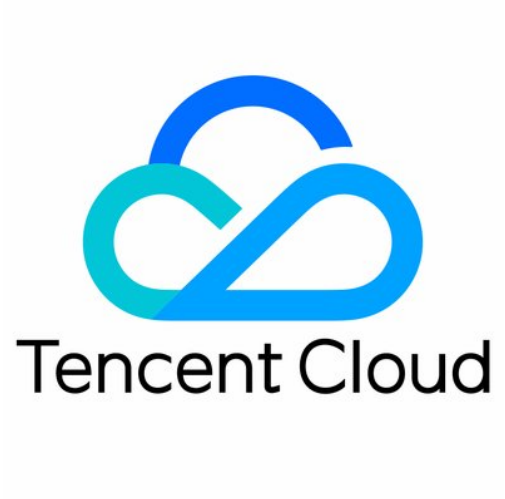
Mobile Live Video Broadcasting

API Documentation

Product Documentation



Contents

API Documentation
iOS
Overview

Last updated：2022-12-23 17:44:57

V2TXLivePlayer

Video player

For details, see V2TXLivePlayer.
The player pulls audio/video data from the specified live streaming URL and plays the data after decoding and local rendering.
The player has the following capabilities:
Playing over protocols including RTMP, HTTP-FLV, TRTC, and WebRTC.
Taking screenshots of streamed video
Delay adjustment. You can set the minimum and maximum cache time for auto adjustment by the player.
Custom video processing. You can process live video based on your project requirements before rendering and playback.

Basic SDK APIs

API	Description
setObserver	Sets player callbacks.

Basic playback APIs

API	Description
setRenderView	Sets the player's rendering view.
startLivePlay	Since v10.7, `startPlay` has been replaced by `startLivePlay`, and you need to call `V2TXLivePremier#setLicence` or `TXLiveBase#setLicence` to set the license to use the live playback feature (you only need to set the license once). Otherwise, playback will fail (black screen). You can use a live stream publishing license, UGSV license, or video playback license to activate the live playback feature. If you don't have any of the licenses, you can buy one or apply for a trial license for free to use the feature.

stopPlay	Stops playback.
isPlaying	Gets whether playback is ongoing.

Video APIs

API	Description
setRenderRotation	Sets video rotation.
setRenderFillMode	Sets the fill mode.
pauseVideo	Pauses the player's video.
resumeVideo	Resumes the player's video.
snapshot	Takes a screenshot of the played video.
enableCustomRendering	Sets the callback for custom video rendering.

Audio APIs

API	Description
pauseAudio	Pauses the player's audio.
resumeAudio	Resumes the player's audio.
setPlayoutVolume	Sets the volume.
enableVolumeEvaluation	Enables the volume reminder for playback.

Other APIs

API	Description
setCacheParams	Sets the minimum and maximum cache time (seconds) for auto adjustment by the player.
showDebugView	Sets whether to show the debug view of player status information.

V2TXLivePlayerObserver

Player callbacks

Basic SDK callback APIs

API	Description
onError	Callback for error. This callback is returned when the player encounters an error.
onWarning	Callback for warning
onConnected	Callback for successfully connecting to the server

Video callback APIs

API	Description
onVideoPlaying	Callback for video playback
onVideoLoading	Callback for loading video
onVideoResolutionChanged	Callback for change of player resolution
onSnapshotComplete	Callback for a screenshot taken
onRenderVideoFrame	Callback for custom video rendering

Audio callback APIs

API	Description
onAudioPlaying	Callback for audio playback
onAudioLoading	Callback for loading audio
onPlayoutVolumeUpdate	Callback of the player's volume

Statistics callback APIs

API	Description
onStatisticsUpdate	Callback of player statistics

V2TXLivePusher

Stream publisher

For details, see V2TXLivePusher.

`V2TXLivePusher` encodes local audio/video and publishes the encoded data to a specified URL. It supports any publishing server.

It has the following capabilities:

Custom video capturing. You can customize audio/video data sources based on your project requirements.

Retouching, filters, and stickers. `V2TXLivePusher` integrates multiple retouching algorithms (natural & smooth) and color space filters (custom filters are supported).

QoS control technology. `V2TXLivePusher` can adapt automatically to different upstream network conditions by controlling audio/video traffic in real time based on the network conditions of hosts.

Facial feature adjustment and animated widgets. Powered by YouTu's AI facial recognition technology, `V2TXLivePusher` supports animated widgets and fine-tuning of facial features, such as eye enlarging, face slimming, and nose reshaping. You need to purchase a **YouTu license** to use these live streaming effects.

Basic SDK APIs

API	Description
setObserver	Sets publisher callbacks.

Basic publishing APIs

API	Description
setRenderView	Sets the rendering view for local camera preview.
startPush	Starts publishing audio/video data.
stopPush	Stops publishing audio/video data.
isPushing	Gets whether publishing is ongoing.

Video APIs

API	Description
setVideoQuality	Sets video encoding parameters for publishing.
setRenderRotation	Sets video rotation for local camera preview.
setRenderMirror	Sets the mirror mode for local camera preview.
startCamera	Turns the local camera on.
stopCamera	Turns the local camera off.

startVirtualCamera	Starts image publishing.
stopVirtualCamera	Stops image publishing.
startScreenCapture	Starts screen capturing.
stopScreenCapture	Stops screen capturing.
snapshot	Takes a screenshot of the published video.
setWatermark	Sets watermarks for the publisher. Watermarking is disabled by default.
setEncoderMirror	Sets the mirror mode for encoded video.
enableCustomVideoCapture	Enables/Disables custom video capturing.
sendCustomVideoFrame	Sends the captured video data to the SDK in the custom video capturing mode.
enableCustomVideoProcess	Enables/Disables custom video processing.
sendSeiMessage	Sends an SEI message.

Beauty filter APIs

API	Description
getBeautyManager	Gets the beauty filter management object TXBeautyManager, which is used to set beauty filters.

Audio APIs

API	Description
startMicrophone	Turns the mic on.
stopMicrophone	Turns the mic off.
setAudioQuality	Sets the quality of published audio.
enableVolumeEvaluation	Enables the volume reminder for capturing.

Audio effect APIs

API	Description

getAudioEffectManager	Gets the audio effect management object.

Device management APIs

API	Description
getDeviceManager	Gets the device management object.

Other APIs

API	Description
setProperty	Calls an advanced API of `V2TXLivePusher` .
setMixTranscodingConfig	Sets On-Cloud MixTranscoding parameters.
showDebugView	Sets whether to display the dashboard.

V2TXLivePusherObserver

Basic SDK callback APIs

API	Description
onError	Callback for error. This callback is returned when the publisher encounters an error.
onWarning	Callback for warning

Video callback APIs

API	Description
onPushStatusUpdate	Callback of the publisher's connection status
onSnapshotComplete	Callback for a screenshot taken
onProcessVideoFrame	Callback for custom video processing
onGLContextDestroyed	Callback for destroying the OpenGL context in the SDK
onCaptureFirstVideoFrame	Callback for capturing the first video frame

Audio callback APIs

API	Description
onCaptureFirstAudioFrame	Callback for capturing the first audio frame
onMicrophoneVolumeUpdate	Callback of mic capturing volume

Mixtranscoding callback APIs

API	Description
onSetMixTranscodingConfig	Callback for setting On-Cloud MixTranscoding parameters

Statistics callback APIs

API	Description
onStatisticsUpdate	Callback of publisher statistics

Publishing

V2TXLivePusher

Last updated：2022-10-13 11:40:06

Overview

Tencent Cloud's live stream publisher

Features

`V2TXLivePusher` encodes local audio/video and publishes the encoded data to a specified URL. It supports any publishing server.

It has the following capabilities:

Custom video capturing. You can customize audio/video data sources based on your project requirements.

Retouching, filters, and stickers. `V2TXLivePusher` integrates multiple retouching algorithms (natural & smooth) and color space filters (custom filters are supported).

QoS control technology. `V2TXLivePusher` can adapt automatically to different upstream network conditions by controlling audio/video traffic in real time based on the network conditions of hosts.

Facial feature adjustment and animated widgets. Powered by YouTu's AI facial recognition technology, `V2TXLivePusher` supports animated widgets and fine-tuning of facial features, such as eye enlarging, face slimming, and nose reshaping. You need to purchase a **YouTu license** to use these live streaming effects.

Basic SDK APIs

initWithLiveMode

This API is used to initialize the publisher.

```
- (instancetype)initWithLiveMode:(V2TXLiveMode)liveMode
```

Parameters

Parameter	Type	Description
liveMode	V2TXLiveMode	Publishing protocol: RTMP (default) or ROOM

setObserver

This API is used to set the callbacks of the publisher. After the setting, you can listen for callback events of `V2TXLivePusher`, including publisher status, volume, statistics, warning and error messages, etc.

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

Parameters

Parameter	Type	Description
observer	V2TXLivePusherObserver	Target object for the publisher's callbacks. For more information, please see V2TXLivePusherObserver.

Basic Publishing APIs

setRenderView

This API is used to set the view for local camera preview. Images captured by the local camera are displayed on the view passed in after retouching, facial feature adjustments, and filter application.

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

Parameters

Parameter	Type	Description
view	TXView *	View for local camera preview

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

startPush

This API is used to start publishing audio/video data.

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

Parameters

Parameter	Type	Description
url	NSString *	Publishing URL. Any publishing server is supported.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the URL is invalid.

`V2TXLIVE_ERROR_REFUSED` : RTC does not support using the same stream ID for publishing and playback on the same device.

stopPush

This API is used to stop publishing audio/video data.

```
- (V2TXLiveCode)stopPush
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

isPushing

This API is used to get whether the publisher is publishing streams.

```
- (int)isPushing
```

Response

Whether streams are being played

`1` : yes

`0` : no

Video APIs

setVideoQuality

// Set video encoding parameters for publishing

```
- (V2TXLiveCode)setVideoQuality:(V2TXLiveVideoEncoderParam *)param;
```

Parameters

Parameter	Type	Description
param	V2TXLiveVideoEncoderParam	Video encoding parameters

setRenderRotation

This API is used to set the rotation of local camera preview.

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

Parameters

Parameter	Type	Description
rotation	V2TXLiveRotation	Degrees by which the image is rotated. Default value: `V2TXLiveRotation0`

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveRotation` enumerated values

Value	Description
V2TXLiveRotation0	No rotation
V2TXLiveRotation90	Rotate 90 degrees clockwise
V2TXLiveRotation180	Rotate 180 degrees clockwise
V2TXLiveRotation270	Rotate 270 degrees clockwise

setRenderMirror

This API is used to set the mirror mode of local camera preview.

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

Parameters

Parameter	Type	Description
mirrorType	V2TXLiveMirrorType	Mirror mode of the camera. Default value: `V2TXLiveMirrorTypeAuto`

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveMirrorType` enumerated values

Value	Description
V2TXLiveMirrorTypeAuto	The default value. In this mode, the front camera is mirrored, but the rear camera is not.
V2TXLiveMirrorTypeEnable	Both the front and rear cameras are mirrored.
V2TXLiveMirrorTypeDisable	Neither the front nor rear camera is mirrored.

startCamera

This API is used to turn the local camera on.

Note:

Among `startVirtualCamera` , `startCamera` , and `startScreenCapture` , only one can be used to publish data under the same pusher instance. If, for example, `startCamera` is called first and then `startVirtualCamera` , the SDK will pause the publishing of camera data and start image publishing.

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

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopCamera

This API is used to turn the local camera off.

```
- (V2TXLiveCode)stopCamera
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

startVirtualCamera

This API is used to start image publishing.

Note:

Among `startVirtualCamera` , `startCamera` , and `startScreenCapture` , only one can be used to publish data under the same pusher instance. If, for example, `startCamera` is called first and then `startVirtualCamera` , the SDK will pause the publishing of camera data and start image publishing.

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

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopVirtualCamera

This API is used to stop image publishing.

```
- (V2TXLiveCode)stopVirtualCamera
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

startScreenCapture

This API is used to start screen recording.

Note:

To start screen recording on iOS, instead of using this API, you need to do the following.

Among `startVirtualCamera` , `startCamera` , and `startScreenCapture` , only one can be used to publish data under the same pusher instance. If, for example, `startCamera` is called first and then `startVirtualCamera` , the SDK will pause the publishing of camera data and start image publishing.

1. First, use Broadcast Upload Extension to start screen recording.
2. Then, call `enableCustomVideoCapture` to enable custom capturing.
3. At last, call `sendCustomVideoFrame` to send the screen images captured by Broadcast Upload Extension.

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

Parameters

Parameter	Type	Description
appGroup	NSString *	The App Group ID shared by the host app and Broadcast Upload Extension. It can be set to `nil` , but setting it as instructed in our documentation will make the feature more reliable.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopScreenCapture

This API is used to stop screen recording.

```
- (V2TXLiveCode)stopScreenCapture
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

snapshot

This API is used to take a screenshot of the local video while it is being published.

Note:

After `V2TXLIVE_OK` is returned, you can get the screenshot taken in the `V2TXLivePusherObserver#onSnapshotComplete:(TXImage *)image` callback.

```
- (V2TXLiveCode)snapshot
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_REFUSED` : failed to take a screenshot because publishing has stopped.

setWatermark

This API is used to set a watermark for the publisher. Watermarking is disabled by default.

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

Parameters

Parameter	Type	Description
image	TXImage *	Watermark image. If this parameter is `null` , it means watermarking is disabled.
x	float	Horizontal coordinate of the watermark

y	float	Vertical coordinate of the watermark
scale	float	Scale ratio of the watermark image

setEncoderMirror

This API is used to set the mirror mode of encoded images.

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

Parameters

Parameter	Type	Description
mirror	BOOL	Whether to mirror encoded images. Default value: `NO`

enableCustomVideoCapture

This API is used to enable/disable custom video capturing. In the custom video capturing mode, the SDK stops capturing images from the camera, but continues to encode and send images.

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

Parameters

Parameter	Type	Description
enable	BOOL	Whether to enable custom capturing. Default value: `NO`

sendCustomVideoFrame

This API is used to send the custom video data captured to the SDK.

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

Parameters

Parameter	Type	Description
videoFrame	V2TXLiveVideoFrame *	Video frames sent to the SDK

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the video data is invalid.

`V2TXLIVE_ERROR_REFUSED` : failed. You must call `enableCustomVideoCapture` to enable custom video capturing first.

enableCustomVideoProcess

This API is used to enable/disable custom video processing.

```
- (V2TXLiveCode)enableCustomVideoProcess:(BOOL)enable pixelFormat:(V2TXLivePixelFor
```

Parameters

Parameter	Type	Description
enable	BOOL	Whether to enable custom video processing. Default value: `NO`
pixelFormat	V2TXLivePixelFormat	Pixel format of video frames
bufferType	V2TXLiveBufferType	Buffer type of video data

`V2TXLivePixelFormat` enumerated values

Value	Description
V2TXLivePixelFormatUnknown	Unknown
V2TXLivePixelFormatI420	YUV420P (I420)
V2TXLivePixelFormatTexture2D	OpenGL 2D texture

`V2TXLiveBufferType` enumerated values

Value	Description
V2TXLiveBufferTypeUnknown	Unknown
V2TXLiveBufferTypeByteBuffer	`DirectBuffer` , which carries buffers in the format of I420 and others and is used at the native layer.
V2TXLiveBufferTypeByteArray	`byte[]` , which carries buffers in the format of I420 and others and is used at the Java layer.
V2TXLiveBufferTypeTexture	Texture ID, which allows direct operation. It delivers the best performance

	and has the smallest impact on video quality.

sendSeiMessage

This API is used to send SEI messages. The player V2TXLivePlayer can receive SEI messages via the `onReceiveSeiMessage` callback in V2TXLivePlayerObserver.

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

Parameters

Parameter	Type	Description
payloadType	int	Data type. Valid values: `5` , `242` (recommended)
data	NSData *	Data to send

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

Beauty Filter APIs

getBeautyManager

This API is used to get the beauty filter management object TXBeautyManager.
You can do the following using `TXBeautyManager` :
Set the beauty filter style and apply effects including skin brightening, rosy skin, eye enlarging, face slimming, chin slimming, chin lengthening/shortening, face shortening, nose narrowing, eye brightening, teeth whitening, eye bag removal, wrinkle removal, and smile line removal.
Adjust the hairline, eye spacing, eye corners, lip shape, nose wings, nose position, lip thickness, and face shape.
Apply animated effects such as face widgets (materials).
Add makeup effects.
Recognize gestures.

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

Audio APIs

startMicrophone

This API is used to turn the mic on.

```
- (V2TXLiveCode)startMicrophone
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopMicrophone

This API is used to turn the mic off.

```
- (V2TXLiveCode)stopMicrophone
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

setAudioQuality

This API is used to set audio quality.

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

Parameters

Parameter	Type	Description
quality	V2TXLiveAudioQuality	Audio quality

`V2TXLiveAudioQuality` enumerated values

Value	Description
V2TXLiveAudioQualitySpeech	Speech. Audio sample rate: 16 kHz; sound channel: mono; bitrate: 16 Kbps This is designed for speech-based call scenarios such as online conferencing and audio call.
V2TXLiveAudioQualityDefault	Default. Audio sample rate: 48 kHz; sound channel: mono; bitrate: 50 Kbps This is the default audio quality of the SDK and is recommended.
V2TXLiveAudioQualityMusic	Music. Audio sample rate: 48 kHz; sound channel: dual; bitrate: 128 Kbps

	This is designed for music-oriented scenarios with hi-fi requirements, such as karaoke and live music streaming.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_REFUSED` : you cannot change audio quality settings when streams are being published.

enableVolumeEvaluation

This API is used to enable the volume reminder for audio capturing.

Note:

After enabling the volume reminder, you can get the SDK's volume evaluation through the `V2TXLivePusherObserver#onMicrophoneVolumeUpdate:(NSInteger)volume` callback.

```
- (V2TXLiveCode)enableVolumeEvaluation:(NSInteger)intervalMs;
```

Parameters

Parameter	Type	Description
intervalMs	NSInteger	Interval (ms) for triggering the `onMicrophoneVolumeUpdate` callback. The minimum interval allowed is 100 ms. If the value is `0` (default) or smaller, the callback is disabled. `300` is recommended.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

Audio Effect APIs

getAudioEffectManager

This API is used to get the audio effect management object TXAudioEffectManager.

You can do the following using `TXAudioEffectManager` :

Set the volume of audio captured by the mic.

Set reverb and voice changing effects.

Enable in-ear monitoring and set its volume.

Add background music and specify how to play it.

```
- (TXAudioEffectManager *)getAudioEffectManager
```

Device Management APIs

getDeviceManager

This API is used to get the device management object TXDeviceManager.

You can do the following using `TXDeviceManager` :

Switch between the front and rear cameras.

Enable autofocus.

Set camera zoom level.

Enable/Disable flashlight.

```
- (TXDeviceManager *)getDeviceManager;
```

Other APIs

setProperty

This API is used to call the advanced APIs of `V2TXLivePusher` .

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

Parameters

Parameter	Type	Description
key	NSString *	Key of the advanced API to call
value	NSObject *	Parameters required by the advanced API

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because `key` is empty.

setMixTranscodingConfig

This API is used to set On-Cloud MixTranscoding parameters. If you have enabled relayed push on the **Function Configuration** page of the TRTC console, then each channel of image will have a default CDN live streaming address. There may be multiple hosts 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 MixTranscoding.

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

Parameters

Parameter	Type	Description
config	V2TXLiveTranscodingConfig *	On-Cloud MixTranscoding configuration

V2TXLiveTranscodingConfig

Value	Description
V2TXLiveBufferTypeUnknown	Unknown
V2TXLiveBufferTypeByteBuffer	`DirectBuffer` , which carries buffers in the format of I420 and others and is used at the native layer.
V2TXLiveBufferTypeByteArray	`byte[]` , which carries buffers in the format of I420 and others and is used at the Java layer.
V2TXLiveBufferTypeTexture	Texture ID, which allows direct operation. It delivers the best performance and has the smallest impact on video quality.

showDebugView

This API is used to set whether to display the dashboard.

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

Parameters

Parameter	Type	Description
isShow	BOOL	Whether to show the debug view. Default value: `NO`

V2TXLivePusherObserver

Last updated：2022-10-13 11:40:06

Overview

Callback notifications for live stream publishing

Features

You can use `V2TXLivePusherObserver` to receive notifications about V2TXLivePusher, including publisher connection status, first audio/video frame, statistics, and warning and error messages.

Basic Callback APIs

onError

Callback for error. This callback is triggered when the publisher encounters an error.

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

Parameters

Parameter	Type	Description
code	V2TXLiveCode	Error code
msg	NSString *	Error message
extraInfo	NSDictionary *	Extra information

onWarning

Callback for warning.

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

Parameters

Parameter	Type	Description
code	V2TXLiveCode	Warning code
msg	NSString *	Warning message
extraInfo	NSDictionary *	Extra information

Video Callback APIs

onPushStatusUpdate

Callback of the publisher's connection status.

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

Parameters

Parameter	Type	Description
status	V2TXLivePushStatus	Status code
msg	NSString *	Status message
extraInfo	NSDictionary *	Extra information

`V2TXLivePushStatus` enumerated values

Value	Description
V2TXLivePushStatusDisconnected	Disconnected from the server
V2TXLivePushStatusConnecting	Connecting to the server
V2TXLivePushStatusConnectSuccess	Connected to the server
V2TXLivePushStatusReconnecting	Reconnecting to the server

onSnapshotComplete

Callback for a screenshot taken.

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

Parameters

Parameter	Type	Description
image	TXImage *	The video image captured

onProcessVideoFrame

Callback for custom video processing.

Note:

You will receive this callback after you call `V2TXLivePusher#enableCustomVideoProcess:(BOOL)enable pixelFormat:(V2TXLivePixelFormat)pixelFormat bufferType:(V2TXLiveBufferType)bufferType` to enable custom video processing.

```
- (void)onProcessVideoFrame:(V2TXLiveVideoFrame * _Nonnull)srcFrame dstFrame:(V2TXL
```

Parameters

Parameter	Type	Description
srcFrame	V2TXLiveVideoFrame *	Images before processing
dstFrame	V2TXLiveVideoFrame *	Images after processing

onGLContextDestroyed

Callback for a GL context for custom video processing being destroyed.

```
- (void)onGLContextDestroyed
```

onCaptureFirstVideoFrame

Callback for capturing the first video frame.

```
- (void)onCaptureFirstVideoFrame
```

Audio Callback APIs

onCaptureFirstAudioFrame

Callback for capturing the first audio frame.

```
- (void)onCaptureFirstAudioFrame
```

onMicrophoneVolumeUpdate

Callback of mic capturing volume.

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

Statistics Callback API

onStatisticsUpdate

Callback of publisher statistics.

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

Parameters

Parameter	Type	Description
statistics	V2TXLivePusherStatistics *	Publisher statistics

MixTranscoding Callback API

onSetMixTranscodingConfig

Callback for setting On-Cloud MixTranscoding parameters.

Note:

You will receive this callback after you call `V2TXLivePusher#setMixTranscodingConfig:(V2TXLiveTranscodingConfig *)config` to set On-Cloud MixTranscoding parameters.

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

Parameter	Type	Description
code	V2TXLiveCode	`0`: successful; other values: failed
msg	NSString *	Error message

TXBeautyManager

Last updated：2022-10-13 11:40:06

Basic APIs for beauty filter, makeup filter, and animated widgets.

setBeautyStyle

This API is used to set the beauty filter type.

```
- (void)setBeautyStyle:(TXBeautyStyle)level
```

Parameters

Parameter	Type	Description
level	TXBeautyStyle	Beauty filter style. `TXBeautyStyleSmooth` indicates smooth; `TXBeautyStyleNature` indicates natural; and `TXBeautyStylePitu` indicates misty.

setFilter

This API is used to specify the material filter effect.

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

Note:

Filter images must be in the PNG format. The filter lookup table image used in the demo is located in `TXLiteAVDemo/Resource/Beauty/filter/FilterResource.bundle` .

setSpecialRatio

This API is used to set the strength of a filter.

```
- (void)setFilterStrength:(float)strength
```

Parameters

Parameter	Type	Description
strength	float	Value range: 0-1. The larger the value, the more obvious the effect. Default value: 0.5.

Introduction

In application scenarios such as shows, a high strength is required to highlight the characteristics of hosts. The default strength is 0.5, and if it is not sufficient, it can be adjusted with the following APIs.

setGreenScreenFile

This API is used to set the green screen effect for videos. It works only in the Enterprise Edition.

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

Parameters

Parameter	Type	Description
file	NString *	Path to the video file, which can be in MP4 format. nil: disable the effect.

Introduction

The green screen feature here is not intelligent keying. It requires that a green screen exists behind the recorded person or object for further chroma keying.

setBeautyLevel

This API is used to set the strength of the beauty filter.

```
- (void)setBeautyLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the beauty filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setWhitenessLevel

This API is used to set the strength of the skin brightening filter.

```
- (void)setWhitenessLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the skin brightening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setRuddyLevel

This API is used to set the strength of the rosy skin filter.

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

Parameters

Parameter	Type	Description
level	float	Strength of the rosy skin filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setEyeScaleLevel

This API is used to set the strength of the eye enlarging filter. It works only in the Enterprise Edition.

```
- (void)setEyeScaleLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the eye enlarging filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setFaceSlimLevel

This API is used to set the strength of the face slimming filter. It works only in the Enterprise Edition.

```
- (void)setFaceSlimLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the face slimming filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setFaceVLevel

This API is used to set the strength of the chin slimming filter. It works only in the Enterprise Edition.

```
- (void)setFaceVLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the chin slimming filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setChinLevel

This API is used to set the strength of the jaw shrinking or expanding filter. It works only in the Enterprise Edition.

```
- (void)setChinLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the jaw shrinking or expanding filter. Value range: -9 to 9. 0 means disabling the filter. Values less than 0 mean shrinking the jaw, and values greater than 0 mean expanding the jaw.

setFaceShortLevel

This API is used to set the strength of the face shortening filter. It works only in the Enterprise Edition.

```
- (void)setFaceShortLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the face shortening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setNoseSlimLevel

This API is used to set the strength of the nose slimming filter. It works only in the Enterprise Edition.

```
- (void)setNoseSlimLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the nose slimming filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setEyeLightenLevel

This API is used to set the strength of the eye brightening filter. It works only in the Enterprise Edition.

```
- (void)setEyeLightenLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the eye brightening filter. Value range: 0-9. 0 means disabling the filter.

		The larger the value, the more obvious the effect.

setToothWhitenLevel

This API is used to set the strength of the teeth whitening filter. It works only in the Enterprise Edition.

```
- (void)setToothWhitenLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the teeth whitening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setWrinkleRemoveLevel

This API is used to set the strength of the wrinkle removal filter. It works only in the Enterprise Edition.

```
- (void)setWrinkleRemoveLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the wrinkle removal filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setPounchRemoveLevel

This API is used to set the strength of the eye bag removal filter. It works only in the Enterprise Edition.

```
- (void)setPounchRemoveLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the eye bag removal filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setSmileLinesRemoveLevel

This API is used to set the strength of the smile line removal filter. It works only in the Enterprise Edition.

```
- (void)setSmileLinesRemoveLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the smile line removal filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setForeheadLevel

This API is used to set the strength of the hairline lowering filter. It works only in the Enterprise Edition.

```
- (void)setForeheadLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the hairline lowering filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the lower the hairline.

setEyeDistanceLevel

This API is used to set the strength of the eye distance shortening filter. It works only in the Enterprise Edition.

```
- (void)setEyeDistanceLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the eye distance shortening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the shorter the eye distance.

setEyeAngleLevel

This API is used to set the strength of the eye tilting filter. It works only in the Enterprise Edition.

```
- (void)setEyeAngleLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the eye tilting filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more upward the outer eye corners, and the more downward the inner eye corners.

setMouthShapeLevel

This API is used to set the strength of the mouth narrowing filter. It works only in the Enterprise Edition.

```
- (void)setMouthShapeLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the mouth narrowing filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the smaller the mouth.

setNoseWingLevel

This API is used to set the strength of the nose wing narrowing filter. It works only in the Enterprise Edition.

```
- (void)setNoseWingLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the nose wing narrowing filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the smaller the nose wing.

setNosePositionLevel

This API is used to set nose position. It works only in the Enterprise Edition.

```
- (void)setNosePositionLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Nose position level. Value range: 0-9. 0 means disabling the filter. The larger the value, the lower the nose position.

setLipsThicknessLevel

This API is used to set the strength of the lip thickening filter. It works only in the Enterprise Edition.

```
- (void)setLipsThicknessLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the lip thickening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the thicker the lips.

setFaceBeautyLevel

This API is used to set the strength of the face shape filter. It works only in the Enterprise Edition.

```
- (void)setFaceBeautyLevel:(float)level
```

Parameters

Parameter	Type	Description
level	float	Strength of the face shape filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setMotionTmpl

This API is used to select AI animated effect widgets. It works only in the Enterprise Edition.

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

Parameters

Parameter	Type	Description
tmplDir	NSString *	Directory of the animated effect.
tmplName	NSString *	Animated effect name.

setMotionMute

This API is used to mute animated effects. It works only in the Enterprise Edition. Some animated effects have audio effects, which can be disabled through this API when they are played back.

```
- (void)setMotionMute:(BOOL)motionMute
```

Parameters

Parameter	Type	Description
motionMute	BOOL	YES: mute. NO: unmute.

Playback
V2TXLivePlayer

Last updated：2022-10-20 15:57:07

Overview

Tencent Cloud's live stream player

The player pulls audio/video data from the specified live streaming URL and plays the data after decoding and local rendering.

Features

The player has the following capabilities:

Playing over protocols including RTMP, HTTP-FLV, TRTC, and WebRTC.

Taking screenshots of streamed video.

Delay adjustment. You can set the minimum and maximum cache time for auto adjustment by the player.

Custom video processing. You can process live video based on your project requirements before rendering and playback.

Basic SDK APIs

setObserver

This API is used to set the callbacks of the player. After setting, you can listen for callback events of `V2TXLivePlayer`, including player status, playback volume, first audio/video frame, statistics, and warning and error messages.

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

Parameters

Parameter	Type	Description
observer	V2TXLivePlayerObserver	Target object for the player's callbacks. For details, please see V2TXLivePlayerObserver.

Basic Playback APIs

setRenderView

This API is used to set the rendering view to display video.

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

Parameters

Parameter	Type	Description
view	TXView *	The player's rendering view

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

startLivePlay

This API is used to start playing audio/video streams.

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

Note:

Since v10.7, `startPlay` has been replaced by `startLivePlay` , and you need to call `V2TXLivePremier#setLicence` or `TXLiveBase#setLicence` to set the license to use the live playback feature (you only need to set the license once). Otherwise, playback will fail (black screen). You can use a live stream publishing license, UGSV license, or video playback license to activate the live playback feature. If you don't have any of the licenses, you can buy one or apply for a trial license for free.

Parameters

Parameter	Type	Description
url	NSString *	Playback URL of audio/video streams, which supports RTMP, HTTP-FLV, TRTC, and WebRTC

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the URL is invalid.

`V2TXLIVE_ERROR_REFUSED` : RTC does not support using the same stream ID for publishing and playback on the same device.

stopPlay

This API is used to stop playing audio/video streams.

```
- (V2TXLiveCode)stopPlay;
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

isPlaying

This API is used to get whether the player is playing streams.

```
- (int)isPlaying
```

Response

Whether streams are being played

`1` : yes

`0` : no

Video APIs

setRenderRotation

This API is used to set the rotation of images displayed by the player.

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

Parameters

Parameter	Type	Description
rotation	V2TXLiveRotation	The degrees by which images are rotated. Default value: 0.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveRotation` enumerated values

Value	Description

V2TXLiveRotation0	No rotation
V2TXLiveRotation90	Rotate 90 degrees clockwise
V2TXLiveRotation180	Rotate 180 degrees clockwise
V2TXLiveRotation270	Rotate 270 degrees clockwise

setRenderFillMode

This API is used to set the image fill mode.

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

Parameters

Parameter	Type	Description
mode	V2TXLiveFillMode	The image fill mode. Default value: `V2TXLiveFillModeFit` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveFillMode` enumerated values

Value	Description
V2TXLiveFillModeFit	The image fits the screen without cropping. If the aspect ratio of the image and the screen do not match, there will be black bars.
V2TXLiveFillModeFill	The image fills the entire screen, without black bars. If the aspect ratio of the image and screen do not match, the parts of the image that don't fit will be cropped.

pauseVideo

This API is used to pause playing video streams.

```
- (V2TXLiveCode)pauseVideo
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

resumeVideo

This API is used to resume playing video streams.

```
- (V2TXLiveCode)resumeVideo
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

snapshot

This API is used to take a screenshot of streamed video.

Note:

After `V2TXLIVE_OK` is returned, you can get the screenshot taken in the `[V2TXLivePlayerObserver onSnapshotComplete: image:]` callback.

```
- (V2TXLiveCode)snapshot
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_REFUSED` : failed to take a screenshot because playback has stopped.

enableCustomRendering

This API is used to set custom video rendering.

You can use this API to obtain each frame of decoded video for custom rendering.

Note:

After custom rendering is enabled, you can get video frames in the `[V2TXLivePlayerObserver onRenderVideoFrame:frame:]` callback.

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

Parameters

Parameter	Type	Description
enable	BOOL	Whether to enable custom rendering. Default value: `NO` .
pixelFormat	V2TXLivePixelFormat	Pixel format of the video called back for custom rendering

bufferType	V2TXLiveBufferType	Buffer type of the video called back for custom rendering

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_NOT_SUPPORTED` : unsupported pixel format or data format.

`V2TXLivePixelFormat` enumerated values

Value	Description
V2TXLivePixelFormatUnknown	Unknown
V2TXLivePixelFormatI420	YUV420P (I420)
V2TXLivePixelFormatNV12	YUV420SP (NV12)
V2TXLivePixelFormatBGRA32	BGRA8888
V2TXLivePixelFormatTexture2D	OpenGL 2D texture

`V2TXLiveBufferType` enumerated values

Value	Description
V2TXLiveBufferTypeUnknown	Unknown
V2TXLiveBufferTypePixelBuffer	This type can be used directly and is the most efficient. The iOS system provides various APIs to get or process pixel buffers.
V2TXLiveBufferTypeNSData	This type affects performance to some extent because the process of converting `NSData` to `PixelBuffer` (the format the SDK can handle directly) involves memory copying.
V2TXLiveBufferTypeTexture	Texture ID, which allows direct operation and delivers the best performance.

Audio APIs

pauseAudio

This API is used to pause playing audio streams.

```
- (V2TXLiveCode)pauseAudio
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

resumeAudio

This API is used to resume playing audio streams.

```
- (V2TXLiveCode)resumeAudio
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

setPlayoutVolume

This API is used to set volume.

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

Parameters

Parameter	Type	Description
volume	NSUInteger	Volume. Value range: 0-100. Default value: `100` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

enableVolumeEvaluation

This API is used to enable the volume reminder for playback.

Note:

After enabling the volume reminder, you can get the SDK's volume evaluation through the `[V2TXLivePlayerObserver onPlayoutVolumeUpdate:volume:]` callback.

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

Parameters

Parameter	Type	Description

intervalMs	NSUInteger	Interval (ms) for triggering the `onPlayoutVolumeUpdate` callback. The minimum interval allowed is 100 ms. If the value is `0` (default) or smaller, the callback is disabled. `300` is recommended.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

Other APIs

setCacheParams

This API is used to set the minimum and maximum cache time (seconds) for auto adjustment by the player.

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

Parameters

Parameter	Type	Description
minTime	CGFloat	The minimum cache time for auto adjustment by the player. The value must be greater than `0` . Default value: `1`
maxTime	CGFloat	The maximum cache time for auto adjustment by the player. The value must be greater than `0` . Default value: `5`

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

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

`V2TXLIVE_ERROR_REFUSED` : you cannot change the cache policy while streams are being played.

showDebugView

This API is used to set whether to show the debug view for the player status information.

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

Parameters

Parameter	Type	Description
isShow	BOOL	Whether to show the debug view. Default value: `NO` .

V2TXLivePlayerObserver

Last updated：2022-10-13 11:40:06

Overview

Callbacks of Tencent Cloud's live stream player.

Features

You can use `V2TXLivePlayerObserver` to receive callbacks of V2TXLivePlayer, including player status, playback volume, first audio/video frame, statistics, warning and error messages, etc.

Basic Callback APIs

onError

Callback for error

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
code	V2TXLiveCode	Error code
msg	NSString *	Error message
extraInfo	NSDictionary *	Extra information

onWarning

Callback for warning

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

Parameters

Parameter	Type	Description

player	V2TXLivePlayer	The player object sending the callback
code	V2TXLiveCode	Warning code
msg	NSString *	Warning message
extraInfo	NSDictionary *	Extra information

onConnected

Callback for successfully connecting to the server

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
extraInfo	NSDictionary *	Extra information

Video Callback APIs

onVideoPlaying

Callback for video playback

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
firstPlay	BOOL	Whether it is the first playback
extraInfo	NSDictionary *	Extra information

onVideoLoading

Callback for loading video

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
extraInfo	NSDictionary *	Extra information

onVideoResolutionChanged

Callback for change of player resolution

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
width	NSInteger	Video width
height	NSInteger	Video height

onSnapshotComplete

Callback for a screenshot taken

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
image	TXImage *	The video image captured

onRenderVideoFrame

Callback for custom video rendering

Note:

You will receive this callback after calling `[V2TXLivePlayer enableCustomRendering:pixelFormat:bufferType:]` to enable custom video rendering.

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
videoFrame	V2TXLiveVideoFrame *	Video frame

Audio Callback APIs

onAudioPlaying

Callback for audio playback

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
firstPlay	BOOL	Whether it is the first playback
extraInfo	NSDictionary *	Extra information

onAudioLoading

Callback for loading audio

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback

extraInfo	NSDictionary *	Extra information

onPlayoutVolumeUpdate

Callback of the player's volume

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
volume	NSInteger	Volume. Value range: 0-100

Statistics Callback API

onStatisticsUpdate

Callback of the player's statistics

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
statistics	V2TXLivePlayerStatistics	Player statistics

Demo

Last updated：2022-10-20 15:57:07

Tencent Cloud offers a straightforward and easy-to-understand API example project regarding frequently asked questions among developers. You can use it to quickly learn how to use different APIs.

Download Address

Platform	GitHub Address
iOS	GitHub

Contents

The demo project covers the following features:

Basic Features:

Publishing from Camera

Publishing from Screen

Playback

Same-Room Communication

Cross-Room Communication

Advanced Features:

Custom Video Capturing

Third-Party Beauty Filters

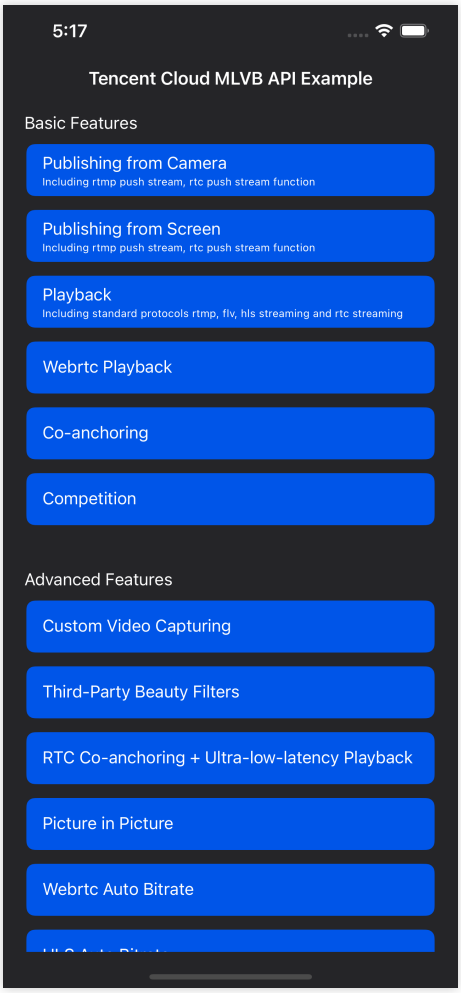
RTC-based Co-anchoring + Ultra-Low-Latency Playback

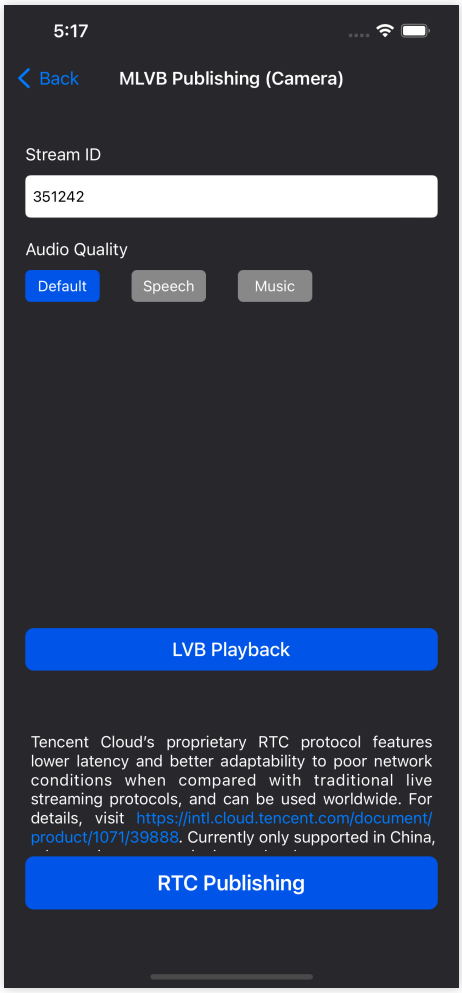
Note:

For clarity purposes, the naming of folders in the project may differ slightly from a standard Xcode project in terms of letter case.

UI Screenshots

Main View	Publishing View	Playb





Android
Overview

Last updated：2022-10-20 15:57:07

V2TXLivePlayer

Video player

For details, see V2TXLivePlayer.
The player pulls audio/video data from the specified live streaming URL and plays the data after decoding and local rendering.
The player has the following capabilities:
Playing over protocols including RTMP, HTTP-FLV, TRTC, and WebRTC.
Taking screenshots of streamed video.
Delay adjustment. You can set the minimum and maximum cache time for auto adjustment by the player.
Custom video processing. You can process live video based on your project requirements before rendering and playback.

Basic SDK APIs

API	Description
setObserver	Sets player callbacks.

Basic playback APIs

API	Description
setRenderView	Sets the player's rendering view `TXCloudVideoView`.
setRenderView	Sets the player's rendering view `TextureView`.
setRenderView	Sets the player's rendering view `SurfaceView`.
startLivePlay	Since v10.7, `startPlay` has been replaced by `startLivePlay`, and you need to call `V2TXLivePremier#setLicence` or `TXLiveBase#setLicence` to set the license to use the live playback feature (you only need to set the license once). Otherwise, playback will fail (black screen). You can use a live stream publishing license, UGSV license, or video playback license to activate the live playback feature. If you don't have any of the licenses, you can buy one or apply for a trial license for free to use the feature.

stopPlay	Stops playback.
isPlaying	Gets whether playback is ongoing.

Video APIs

API	Description
setRenderRotation	Sets video rotation.
setRenderFillMode	Sets the fill mode.
pauseVideo	Pauses the player's video.
resumeVideo	Resumes the player's video.
snapshot	Takes a screenshot of the played video.
enableObserveVideoFrame	Sets the callback for custom video rendering.

Audio APIs

API	Description
pauseAudio	Pauses the player's audio.
resumeAudio	Resumes the player's audio.
setPlayoutVolume	Sets the volume.
enableVolumeEvaluation	Enables the volume reminder for playback.

Other APIs

API	Description
setCacheParams	Sets the minimum and maximum cache time (seconds) for auto adjustment by the player.
showDebugView	Sets whether to show the debug view of player status information.

V2TXLivePlayerObserver

Player callbacks

Basic SDK callback APIs

API	Description
onError	Callback for error. This callback is returned when the player encounters an error.
onWarning	Callback for warning
onConnected	Callback for successfully connecting to the server

Video callback APIs

API	Description
onVideoResolutionChanged	Callback for change of player resolution
onVideoLoading	Callback for loading video
onVideoPlaying	Callback for video playback
onSnapshotComplete	Callback for a screenshot taken
onRenderVideoFrame	Callback for custom video rendering

Audio callback APIs

API	Description
onAudioLoading	Callback for loading audio
onAudioPlaying	Callback for audio playback
onPlayoutVolumeUpdate	Callback of the player's volume

Statistics callback APIs

API	Description
onStatisticsUpdate	Callback of player statistics

V2TXLivePusher

Stream publisher

For details, see V2TXLivePusher.

`V2TXLivePusher` encodes local audio/video and publishes the encoded data to a specified URL. It supports any publishing server.

It has the following capabilities:

Custom video capturing. You can customize audio/video data sources based on your project requirements.

Retouching, filters, and stickers. `V2TXLivePusher` integrates multiple retouching algorithms (natural & smooth) and color space filters (custom filters are supported).

QoS control technology. `V2TXLivePusher` can adapt automatically to different upstream network conditions by controlling audio/video traffic in real time based on the network conditions of hosts.

Facial feature adjustment and animated widgets. Powered by YouTu's AI facial recognition technology, `V2TXLivePusher` supports animated widgets and fine-tuning of facial features, such as eye enlarging, face slimming, and nose reshaping. You need to purchase a **YouTu license** to use these live streaming effects.

Basic SDK APIs

API	Description
setObserver	Sets publisher callbacks.

Basic publishing APIs

API	Description
setRenderView	Sets the rendering view for local camera preview.
setRenderView	Sets the rendering view for local camera preview.
setRenderView	Sets the rendering view for local camera preview.
startPush	Starts publishing audio/video data.
stopPush	Stops publishing audio/video data.
isPushing	Gets whether publishing is ongoing.

Video APIs

API	Description
setVideoQuality	Sets video encoding parameters for publishing.
setRenderRotation	Sets video rotation for local camera preview.
setRenderMirror	Sets the mirror mode for local camera preview.

startCamera	Turns the local camera on.
stopCamera	Turns the local camera off.
startVirtualCamera	Starts image publishing.
stopVirtualCamera	Stops image publishing.
startScreenCapture	Starts screen capturing.
stopScreenCapture	Stops screen capturing.
snapshot	Takes a screenshot of the published video.
setWatermark	Sets watermarks for the publisher. Watermarking is disabled by default.
setEncoderMirror	Sets the mirror mode for encoded video.
enableCustomVideoCapture	Enables/Disables custom video capturing.
sendCustomVideoFrame	Sends the captured video data to the SDK in the custom video capturing mode.
enableCustomVideoProcess	Enables/Disables custom video processing.
sendSeiMessage	Sends an SEI message.

Beauty filter APIs

API	Description
getBeautyManager	Gets the beauty filter management object TXBeautyManager, which is used to set beauty filters.

Audio APIs

API	Description
startMicrophone	Turns the mic on.
stopMicrophone	Turns the mic off.
setAudioQuality	Sets the quality of published audio.
enableVolumeEvaluation	Enables the volume reminder for capturing.

Audio effect APIs

API	Description
getAudioEffectManager	Gets the audio effect management object.

Device management APIs

API	Description
getDeviceManager	Gets the device management object.

Other APIs

API	Description
setProperty	Calls an advanced API of `V2TXLivePusher`.
setMixTranscodingConfig	Sets On-Cloud MixTranscoding parameters.
showDebugView	Sets whether to display the dashboard.

V2TXLivePusherObserver

Basic SDK callback APIs

API	Description
onError	Callback for error. This callback is returned when the publisher encounters an error.
onWarning	Callback for warning

Video callback APIs

API	Description
onPushStatusUpdate	Callback of the publisher's connection status
onSnapshotComplete	Callback for a screenshot taken
onProcessVideoFrame	Callback for custom video processing
onGLContextCreated	Callback for creating an OpenGL context in the SDK
onGLContextDestroyed	Callback for destroying the OpenGL context in the SDK

onCaptureFirstVideoFrame	Callback for capturing the first video frame

Audio callback APIs

API	Description
onCaptureFirstAudioFrame	Callback for capturing the first audio frame
onMicrophoneVolumeUpdate	Callback of mic capturing volume

Mixtranscoding callback APIs

API	Description
onSetMixTranscodingConfig	Callback for setting On-Cloud MixTranscoding parameters

Statistics callback APIs

API	Description
onStatisticsUpdate	Callback of publisher statistics

Publishing

V2TXLivePusher

Last updated：2022-10-13 11:40:06

Overview

Tencent Cloud's live stream publisher

Features

`V2TXLivePusher` encodes local audio/video and publishes the encoded data to a specified URL. It supports any publishing server.

It has the following capabilities:

Custom video capturing. You can customize audio and video data sources based on your project requirements.

Retouching, filters, and stickers. `V2TXLivePusher` integrates multiple retouching algorithms (natural & smooth) and color space filters (custom filters are supported).

QoS control technology. `V2TXLivePusher` can adapt automatically to different upstream network conditions by controlling audio/video traffic in real time based on the network conditions of hosts.

Facial feature adjustment and animated widgets. Powered by YouTu's AI facial recognition technology, `V2TXLivePusher` supports animated widgets and fine-tuning of facial features, such as eye enlarging, face slimming, and nose reshaping. You need to purchase a **YouTu license** to use these live streaming effects.

Basic SDK APIs

setObserver

This API is used to set the callbacks of the publisher. After the setting, you can listen for the callback events of V2TXLivePusher, including publisher status, volume, statistics, alerts, and error messages.

```
public abstract void setObserver(V2TXLivePusherObserver observer);
```

Parameters

Parameter	Type	Description
observer	V2TXLivePusherObserver	Target object for the publisher's callbacks. For more information, please see V2TXLivePusherObserver.

Basic Publishing APIs

setRenderView

This API is used to set the view for local camera preview. Images captured by the local camera are displayed on the view passed in after retouching, facial feature adjustments, and filter application.

```
public abstract int setRenderView(TXCloudVideoView view);
```

Parameters

Parameter	Type	Description
view	TXCloudVideoView	View for local camera preview

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

setRenderView

This API is used to set the view for local camera preview. Images captured by the local camera are displayed on the view passed in after retouching, facial feature adjustments, and filter application.

```
public abstract int setRenderView(SurfaceView view);
```

Parameters

Parameter	Type	Description
view	SurfaceView	View for local camera preview

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

setRenderView

This API is used to set the view for local camera preview. Images captured by the local camera are displayed on the view passed in after retouching, facial feature adjustments, and filter application.

```
public abstract int setRenderView(TextureView view);
```

Parameters

Parameter	Type	Description
view	TextureView	View for local camera preview

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

startPush

This API is used to start publishing audio/video data.

```
public abstract int startPush(String url);
```

Parameters

Parameter	Type	Description
url	String	URL to which data is published. Any publishing server is supported.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the URL is invalid.

`V2TXLIVE_ERROR_REFUSED` : RTC does not support using the same stream ID for publishing and playback on the same device.

stopPush

This API is used to stop publishing audio/video data.

```
public abstract int stopPush();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

isPushing

This API is used to get whether the publisher is publishing streams.

```
public abstract int isPushing();
```

Response

Whether streams are being played

`1` : yes

`0` : no

Video APIs

setVideoQuality

// Set video encoding parameters for publishing

```
public abstract int setVideoQuality(V2TXLiveVideoEncoderParam param);
```

Parameters

Parameter	Type	Description
param	V2TXLiveVideoEncoderParam	Video encoding parameters

setRenderRotation

This API is used to set the rotation of local camera preview.

```
public abstract int setRenderRotation(V2TXLiveRotation rotation);
```

Parameters

Parameter	Type	Description
rotation	V2TXLiveRotation	Degrees by which the image is rotated. Default value: `V2TXLiveRotation0`

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveRotation` enumerated values

Value	Description
V2TXLiveRotation0	No rotation

V2TXLiveRotation90	Rotate 90 degrees clockwise
V2TXLiveRotation180	Rotate 180 degrees clockwise
V2TXLiveRotation270	Rotate 270 degrees clockwise

setRenderMirror

This API is used to set the mirror mode of local camera preview.

```
public abstract int setRenderMirror(V2TXLiveMirrorType mirrorType);
```

Parameters

Parameter	Type	Description
mirrorType	V2TXLiveMirrorType	Mirror mode of the camera. Default value: `V2TXLiveMirrorTypeAuto`

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

`V2TXLiveMirrorType` enumerated values

Value	Description
V2TXLiveMirrorTypeAuto	The default value. In this mode, the front camera is mirrored, but the rear camera is not.
V2TXLiveMirrorTypeEnable	Both the front and rear cameras are mirrored.
V2TXLiveMirrorTypeDisable	Neither the front nor rear camera is mirrored.

startCamera

This API is used to turn the local camera on.

Note:

Among `startVirtualCamera` , `startCamera` , and `startScreenCapture` , only one can be used to publish data under the same pusher instance. If, for example, `startCamera` is called first and then `startVirtualCamera` , the SDK will pause the publishing of camera data and start image publishing.

```
public abstract int startCamera(boolean frontCamera);
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopCamera

This API is used to turn the local camera off.

```
public abstract int stopCamera();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

startVirtualCamera

This API is used to start image publishing.

Note:

Among `startVirtualCamera` , `startCamera` , and `startScreenCapture` , only one can be used to publish data under the same pusher instance. If, for example, `startCamera` is called first and then `startVirtualCamera` , the SDK will pause the publishing of camera data and start image publishing.

```
public abstract int startVirtualCamera(Bitmap image);
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopVirtualCamera

This API is used to stop image publishing.

```
public abstract int stopVirtualCamera();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

startScreenCapture

This API is used to start screen recording.

Note:

Among `startVirtualCamera`, `startCamera`, and `startScreenCapture`, only one can be used to publish data under the same pusher instance. If, for example, `startCamera` is called first and then `startVirtualCamera`, the SDK will pause the publishing of camera data and start image publishing.

```
public abstract int startScreenCapture();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

stopScreenCapture

This API is used to stop screen recording.

```
public abstract int stopScreenCapture();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

snapshot

This API is used to take a screenshot of the local video while it is being published.

Note:

After `V2TXLIVE_OK` is returned, you can get the screenshot taken in the `V2TXLivePusherObserver.onSnapshotComplete` callback.

```
public abstract int snapshot();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

`V2TXLIVE_ERROR_REFUSED`: failed to take a screenshot because publishing has stopped.

setWatermark

This API is used to set a watermark for the publisher. Watermarking is disabled by default.

```
public abstract int setWatermark(Bitmap image, float x, float y, float scale);
```

Parameters

Parameter	Type	Description
image	Bitmap	Watermark image. If this parameter is `null`, it means watermarks are disabled.
x	float	Horizontal coordinate of the watermark
y	float	Vertical coordinate of the watermark
scale	float	Scale ratio of the watermark image

setEncoderMirror

This API is used to set the mirror mode of encoded images.

```
public abstract int setEncoderMirror(boolean mirror);
```

Parameters

Parameter	Type	Description
mirror	Boolean	Whether to mirror encoded images. Default value: `false`

enableCustomVideoCapture

This API is used to enable/disable custom video capturing. In the custom video capturing mode, the SDK stops capturing images from the camera, but continues to encode and send images.

```
public abstract int enableCustomVideoCapture(boolean enable);
```

Parameters

Parameter	Type	Description
enable	Boolean	Whether to enable custom video capturing. Default value: `false`.

sendCustomVideoFrame

This API is used to send the custom video data captured to the SDK.

```
public abstract int sendCustomVideoFrame(V2TXLiveVideoFrame videoFrame);
```

Parameters

Parameter	Type	Description

videoFrame	V2TXLiveVideoFrame	Video frame sent to the SDK

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the video data is invalid.

`V2TXLIVE_ERROR_REFUSED` : failed. You must call `enableCustomVideoCapture` to enable custom video capturing first.

enableCustomVideoProcess

This API is used to enable/disable custom video processing.

```
public abstract int enableCustomVideoProcess(boolean enable, V2TXLivePixelFormat pi
```

Parameters

Parameter	Type	Description
enable	Boolean	Whether to enable custom video processing. Default value: `false`
pixelFormat	V2TXLivePixelFormat	Pixel format of video frames
bufferType	V2TXLiveBufferType	Video data buffer type

`V2TXLivePixelFormat` enumerated values

Value	Description
V2TXLivePixelFormatUnknown	Unknown
V2TXLivePixelFormatI420	YUV420P (I420)
V2TXLivePixelFormatTexture2D	OpenGL 2D texture

`V2TXLiveBufferType` enumerated values

Value	Description
V2TXLiveBufferTypeUnknown	Unknown
V2TXLiveBufferTypeByteBuffer	`DirectBuffer` , which carries buffers in the format of I420 and others and is used at the native layer.

V2TXLiveBufferTypeByteArray	`byte[]` , which carries buffers in the format of I420 and others and is used at the Java layer.
V2TXLiveBufferTypeTexture	Texture ID, which allows direct operation. It delivers the best performance and has the smallest impact on video quality.

sendSeiMessage

This API is used to send SEI messages. The player V2TXLivePlayer can receive SEI messages via the `onReceiveSeiMessage` callback in V2TXLivePlayerObserver.

```
public abstract int sendSeiMessage(int payloadType, byte[] data);
```

Parameters

Parameter	Type	Description
payloadType	int	Data type. Valid values: `5` , `242` (recommended)
data	byte[]	Data to send

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

Beauty Filter APIs

getBeautyManager

This API is used to get the beauty filter management object TXBeautyManager.
You can do the following using `TXBeautyManager` :
Set the beauty filter style and apply effects including skin brightening, rosy skin, eye enlarging, face slimming, chin slimming, chin lengthening/shortening, face shortening, nose narrowing, eye brightening, teeth whitening, eye bag removal, wrinkle removal, and smile line removal.
Adjust the hairline, eye spacing, eye corners, lip shape, nose wings, nose position, lip thickness, and face shape.
Apply animated effects such as face widgets (materials).
Add makeup effects.
Recognize gestures.

```
public TXBeautyManager getBeautyManager()
```

Audio APIs

startMicrophone

This API is used to turn the mic on.

```
public abstract int startMicrophone();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopMicrophone

This API is used to turn the mic off.

```
public abstract int stopMicrophone();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

setAudioQuality

This API is used to set audio quality.

```
public abstract int setAudioQuality(V2TXLiveAudioQuality quality);
```

Parameters

Parameter	Type	Description
quality	V2TXLiveAudioQuality	Audio quality

`V2TXLiveAudioQuality` enumerated values

Value	Description
V2TXLiveAudioQualitySpeech	Speech. Audio sample rate: 16 kHz; sound channel: mono; bitrate: 16 Kbps This is designed for speech-based call scenarios such as online conferencing and audio call.
V2TXLiveAudioQualityDefault	Default. Audio sample rate: 48 kHz; sound channel: mono; bitrate: 50 Kbps This is the default audio quality of the SDK and is recommended.

V2TXLiveAudioQualityMusic	Music. Audio sample rate: 48 kHz; sound channel: dual; bitrate: 128 Kbps This is designed for music-oriented scenarios with hi-fi requirements, such as karaoke and live music streaming.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_REFUSED` : you cannot change audio quality settings when streams are being published.

enableVolumeEvaluation

This API is used to enable the volume reminder for audio capturing.

Note:

After enabling the volume reminder, you can get the volume measured by the SDK in the `V2TXLivePusherObserver#onMicrophoneVolumeUpdate(int)` callback.

```
public abstract int enableVolumeEvaluation(int intervalMs);
```

Parameters

Parameter	Type	Description
intervalMs	int	Interval (ms) for volume callbacks. The minimum interval allowed is 100 ms. If the value is `0` (default) or smaller, the callback is disabled. `300` is recommended.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

Audio Effect APIs

getAudioEffectManager

This API is used to get the audio effect management object TXAudioEffectManager.

You can do the following using `TXAudioEffectManager` :

Set the volume of audio captured by the mic.

Set reverb and voice changing effects.

Enable in-ear monitoring and set its volume.

Add background music and specify how to play it.

```
public TXAudioEffectManager getAudioEffectManager()
```

Device Management APIs

getDeviceManager

This API is used to get the device management object TXDeviceManager.

You can do the following using `TXDeviceManager` :

Switch between the front and rear cameras.

Enable autofocus.

Set camera zoom level.

Enable/Disable flashlight.

```
public abstract TXDeviceManager getDeviceManager();
```

Other APIs

setProperty

This API is used to call the advanced APIs of `V2TXLivePusher` .

```
public abstract int setProperty(String key, Object value);
```

Parameters

Parameter	Type	Description
key	String	Key of the advanced API to call
value	Object	Parameters required by the advanced API

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because `key` is empty.

setMixTranscodingConfig

This API is used to set On-Cloud MixTranscoding parameters. If you have enabled relayed push on the **Function Configuration** page of the TRTC console, then each channel of image will have a default CDN live streaming address. There may be multiple hosts 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 MixTranscoding.

```
public abstract int setMixTranscodingConfig(V2TXLiveTranscodingConfig config);
```

Parameters

Parameter	Type	Description
config	V2TXLiveTranscodingConfig	On-Cloud MixTranscoding configuration

V2TXLiveTranscodingConfig

Value	Description
V2TXLiveBufferTypeUnknown	Unknown
V2TXLiveBufferTypeByteBuffer	`DirectBuffer` , which carries buffers in the format of I420 and others and is used at the native layer.
V2TXLiveBufferTypeByteArray	`byte[]` , which carries buffers in the format of I420 and others and is used at the Java layer.
V2TXLiveBufferTypeTexture	Texture ID, which allows direct operation. It delivers the best performance and has the smallest impact on video quality.

showDebugView

This API is used to set whether to display the dashboard.

```
public abstract void showDebugView(boolean isShow);
```

Parameters

Parameter	Type	Description
isShow	boolean	Whether to display the dashboard. Default value: `false`

V2TXLivePusherObserver

Last updated：2022-10-13 11:40:07

Overview

Callback notifications for live stream publishing

Features

You can use `V2TXLivePusherObserver` to receive notifications about V2TXLivePusher, including publisher connection status, first audio/video frame, statistics, and warning and error messages.

Basic Callback APIs

onError

Callback for error. This callback is triggered when the publisher encounters an error.

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

Parameters

Parameter	Type	Description
code	int	Error code
msg	String	Error message
extraInfo	Bundle	Extra information

onWarning

Callback for warning.

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

Parameters

Parameter	Type	Description
code	int	Warning code
msg	String	Warning message
extraInfo	Bundle	Extra information

Video Callback APIs

onPushStatusUpdate

Callback of the publisher's connection status.

```
public void onPushStatusUpdate(V2TXLivePushStatus status, String msg, Bundle extraI
```

Parameters

Parameter	Type	Description
status	V2TXLivePushStatus	Status code
msg	String	Status message
extraInfo	Bundle	Extra information

`V2TXLivePushStatus` enumerated values

Value	Description
V2TXLivePushStatusDisconnected	Disconnected from the server
V2TXLivePushStatusConnecting	Connecting to the server
V2TXLivePushStatusConnectSuccess	Connected to the server
V2TXLivePushStatusReconnecting	Reconnecting to the server

onSnapshotComplete

Callback for a screenshot taken.

```
public void onSnapshotComplete(Bitmap image)
```

Parameters

Parameter	Type	Description
image	Bitmap *	The video image captured

onProcessVideoFrame

Callback for custom video processing.

Note:

You will receive this callback after you call `V2TXLivePusher#enableCustomVideoProcess(boolean, V2TXLiveDef.V2TXLivePixelFormat, V2TXLiveDef.V2TXLiveBufferType)` to enable custom video processing.

```
public void onProcessVideoFrame(V2TXLiveVideoFrame srcFrame, V2TXLiveVideoFrame dst
```

Parameters

Parameter	Type	Description
srcFrame	V2TXLiveVideoFrame	Images before processing
dstFrame	V2TXLiveVideoFrame	Images after processing

onGLContextCreated

Callback for a GL context for custom video processing being created.

```
public void onGLContextCreated()
```

onGLContextDestroyed

Callback for a GL context for custom video processing being destroyed.

```
public void onGLContextDestroyed()
```

onCaptureFirstVideoFrame

Callback for capturing the first video frame.

```
public void onCaptureFirstVideoFrame()
```

Audio Callback APIs

onCaptureFirstAudioFrame

Callback for capturing the first audio frame.

```
public void onCaptureFirstAudioFrame()
```

onMicrophoneVolumeUpdate

Callback of mic capturing volume.

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

Statistics Callback API

onStatisticsUpdate

Callback of publisher statistics.

```
public void onStatisticsUpdate(V2TXLivePusherStatistics statistics)
```

Parameters

Parameter	Type	Description
statistics	V2TXLivePusherStatistics	Publisher statistics

MixTranscoding Callback API

onSetMixTranscodingConfig

Callback for setting On-Cloud MixTranscoding parameters.

Note:

You will receive this callback after you call `V2TXLivePusher#setMixTranscodingConfig(V2TXLiveDef.V2TXLiveTranscodingConfig)` to set On-Cloud MixTranscoding parameters.

```
public void onSetMixTranscodingConfig(int code, String msg)
```

Parameter	Type	Description
code	int	`0` : successful; other values: failed
msg	String	Error message

TXBeautyManager

Last updated：2022-10-13 11:40:07

Basic APIs for beauty filter, makeup filter, and animated widgets.

setBeautyStyle

This API is used to set the beauty filter type.

```
void setBeautyStyle(int beautyStyle)
```

Parameters

Parameter	Type	Description
beautyStyle	int	Beauty filter style. 0: smooth; 1: natural; 2: misty.

setFilter

This API is used to specify the material filter effect.

```
void setFilter(Bitmap bmp)
```

Parameters

Parameter	Type	Description
bmp	Bitmap	Filter image.

Note:

Filter images must be in the PNG format. The filter lookup table image used in the demo is located in `app/src/main/res/drawable-xxhdpi/` .

setFilterStrength

This API is used to set the strength of a filter.

```
void setFilterStrength(float strength)
```

Parameters

Parameter	Type	Description
strength	float	Value range: 0-1. The larger the value, the more obvious the effect. Default value: 0.5.

Introduction

In application scenarios such as shows, a high strength is required to highlight the characteristics of hosts. The default strength is 0.5, and if it is not sufficient, it can be adjusted with the following APIs.

setGreenScreenFile

This API is used to set the green screen effect for videos. It works only in the Enterprise Edition.

```
boolean setGreenScreenFile(String path)
```

Parameters

Parameter	Type	Description
path	String	Path to the video file, which can be in MP4 format. null: disable the effect.

Introduction

The green screen feature here is not intelligent keying. It requires that a green screen exists behind the recorded person or object for further chroma keying.

setBeautyLevel

This API is used to set the strength of the beauty filter.

```
void setBeautyLevel(int beautyLevel)
```

Parameters

Parameter	Type	Description
beautyLevel	int	Strength of the beauty filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setWhitenessLevel

This API is used to set the strength of the skin brightening filter.

```
void setWhitenessLevel(int whitenessLevel)
```

Parameters

Parameter	Type	Description
whitenessLevel	int	Strength of the skin brightening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setRuddyLevel

This API is used to set the strength of the rosy skin filter.

```
void setRuddyLevel(int ruddyLevel)
```

Parameters

Parameter	Type	Description
ruddyLevel	int	Strength of the rosy skin filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setEyeScaleLevel

This API is used to set the strength of the eye enlarging filter. It works only in the Enterprise Edition.

```
void setEyeScaleLevel(int eyeScaleLevel)
```

Parameters

Parameter	Type	Description
eyeScaleLevel	int	Strength of the eye enlarging filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setFaceSlimLevel

This API is used to set the strength of the face slimming filter. It works only in the Enterprise Edition.

```
void setFaceSlimLevel(int faceSlimLevel)
```

Parameters

Parameter	Type	Description
faceSlimLevel	int	Strength of the face slimming filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setFaceVLevel

This API is used to set the strength of the chin slimming filter. It works only in the Enterprise Edition.

```
void setFaceVLevel(int faceVLevel)
```

Parameters

Parameter	Type	Description
faceVLevel	int	Strength of the chin slimming filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setChinLevel

This API is used to set the strength of the jaw shrinking or expanding filter. It works only in the Enterprise Edition.

```
void setChinLevel(int chinLevel)
```

Parameters

Parameter	Type	Description
chinLevel	int	Strength of the jaw shrinking or expanding filter. Value range: -9 to 9. 0 means disabling the filter. Values less than 0 mean shrinking the jaw, and values greater than 0 mean expanding the jaw.

setFaceShortLevel

This API is used to set the strength of the face shortening filter. It works only in the Enterprise Edition.

```
void setFaceShortLevel(int faceShortlevel)
```

Parameters

Parameter	Type	Description
faceShortlevel	int	Strength of the face slimming filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setNoseSlimLevel

This API is used to set the strength of the nose slimming filter. It works only in the Enterprise Edition.

```
void setNoseSlimLevel(int noseSlimLevel)
```

Parameters

Parameter	Type	Description
noseSlimLevel	int	Strength of the face slimming filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setEyeLightenLevel

This API is used to set the strength of the eye brightening filter. It works only in the Enterprise Edition.

```
void setEyeLightenLevel(int eyeLightenLevel)
```

Parameters

Parameter	Type	Description
eyeLightenLevel	int	Strength of the eye brightening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setToothWhitenLevel

This API is used to set the strength of the teeth whitening filter. It works only in the Enterprise Edition.

```
void setToothWhitenLevel(int toothWhitenLevel)
```

Parameters

Parameter	Type	Description
toothWhitenLevel	int	Strength of the teeth whitening filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setWrinkleRemoveLevel

This API is used to set the strength of the wrinkle removal filter. It works only in the Enterprise Edition.

```
void setWrinkleRemoveLevel(int wrinkleRemoveLevel)
```

Parameters

Parameter	Type	Description
wrinkleRemoveLevel	int	Strength of the wrinkle removal filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setPounchRemoveLevel

This API is used to set the strength of the eye bag removal filter. It works only in the Enterprise Edition.

```
void setPounchRemoveLevel(int pounchRemoveLevel)
```

Parameters

Parameter	Type	Description
pounchRemoveLevel	int	Strength of the eye bag removal filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setSmileLinesRemoveLevel

This API is used to set the strength of the smile line removal filter. It works only in the Enterprise Edition.

```
void setSmileLinesRemoveLevel(int smileLinesRemoveLevel)
```

Parameters

Parameter	Type	Description
smileLinesRemoveLevel	int	Strength of the wrinkle removal filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setForeheadLevel

This API is used to set the strength of the hairline lowering filter. It works only in the Enterprise Edition.

```
void setForeheadLevel(int foreheadLevel)
```

Parameters

Parameter	Type	Description
foreheadLevel	int	Strength of the hairline lowering filter. Value range: 0-9. 0 means disabling the filter. The greater the value of 1-9, the lower the hairline.

setEyeDistanceLevel

This API is used to set the strength of the eye distance shortening filter. It works only in the Enterprise Edition.

```
void setEyeDistanceLevel(int eyeDistanceLevel)
```

Parameters

Parameter	Type	Description
eyeDistanceLevel	int	Strength of the eye distance shortening filter. Value range: 0-9. 0 means disabling the filter. The greater the value of 1-9, the shorter the eye distance.

setEyeAngleLevel

This API is used to set the strength of the eye tilting filter. It works only in the Enterprise Edition.

```
void setEyeAngleLevel(int eyeAngleLevel)
```

Parameters

Parameter	Type	Description
eyeAngleLevel	int	Strength of the eye tilting filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more upward the outer eye corners, and the more downward the inner eye corners.

setMouthShapeLevel

This API is used to set the strength of the mouth narrowing filter. It works only in the Enterprise Edition.

```
void setMouthShapeLevel(int mouthShapeLevel)
```

Parameters

Parameter	Type	Description
mouthShapeLevel	int	Strength of the mouth narrowing filter. Value range: 0-9. 0 means disabling the filter. The greater the value of 1-9, the smaller the mouth.

setNoseWingLevel

This API is used to set the strength of the nose wing narrowing filter. It works only in the Enterprise Edition.

```
void setNoseWingLevel(int noseWingLevel)
```

Parameters

Parameter	Type	Description
noseWingLevel	int	Strength of the nose wing narrowing filter. Value range: 0-9. 0 means disabling the filter. The greater the value of 1-9, the smaller the nose wing.

setNosePositionLevel

This API is used to set the nose position. It works only in the Enterprise Edition.

```
void setNosePositionLevel(int nosePositionLevel)
```

Parameters

Parameter	Type	Description
nosePositionLevel	int	Nose position level. Value range: 0-9. 0 means disabling the filter. The greater the value of 1-9, the lower the nose position.

setLipsThicknessLevel

This API is used to set the strength of the lip thickening filter. It works only in the Enterprise Edition.

```
void setLipsThicknessLevel(int lipsThicknessLevel)
```

Parameters

Parameter	Type	Description
lipsThicknessLevel	int	Strength of the lip thickening filter. Value range: 0-9. 0 means disabling the filter. The greater the value of 1-9, the thicker the lips.

setFaceBeautyLevel

This API is used to set the strength of the face shape filter. It works only in the Enterprise Edition.

```
void setFaceBeautyLevel(int faceBeautyLevel)
```

Parameters

Parameter	Type	Description
faceBeautyLevel	int	Strength of the face shape filter. Value range: 0-9. 0 means disabling the filter. The larger the value, the more obvious the effect.

setMotionTmpl

This API is used to select AI animated effect widgets. It works only in the Enterprise Edition.

```
void setMotionTmpl(String motionPath)
```

Parameters

Parameter	Type	Description
motionPath	String	Path to the animated effect.

setMotionMute

This API is used to mute or unmute animated effects. It works only in the Enterprise Edition. Some animated effects have audio effects, which can be disabled through this API when they are played back.

```
void setMotionMute(boolean motionMute)
```

Parameters

Parameter	Type	Description
motionMute	boolean	true: mute. false: unmute.

Playback
V2TXLivePlayer

Last updated：2022-10-20 15:57:07

Overview

Tencent Cloud's live stream player

The player pulls audio/video data from the specified live streaming URL and plays the data after decoding and local rendering.

Features

The player has the following capabilities:

Playing over protocols including RTMP, HTTP-FLV, TRTC, and WebRTC.

Taking screenshots of streamed video.

Delay adjustment. You can set the minimum and maximum cache time for auto adjustment by the player.

Custom video processing. You can process live video based on your project requirements before rendering and playback.

Basic SDK APIs

setObserver

This API is used to set the callbacks of the player. After setting, you can listen for callback events of `V2TXLivePlayer`, including player status, playback volume, first audio/video frame, statistics, and warning and error messages.

```
public abstract void setObserver(V2TXLivePlayerObserver observer);
```

Parameters

Parameter	Type	Description
observer	V2TXLivePlayerObserver	Target object for the player's callbacks. For details, see V2TXLivePlayerObserver.

Basic Playback APIs

setRenderView

This API is used to set the rendering view to display video.

```
public abstract int setRenderView(TXCloudVideoView view);
```

Parameters

Parameter	Type	Description
view	TXCloudVideoView	The player's rendering view

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

setRenderView

This API is used to set the rendering view to display video.

```
public abstract int setRenderView(SurfaceView view);
```

Parameters

Parameter	Type	Description
view	SurfaceView	The player's rendering view

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

setRenderView

This API is used to set the rendering view to display video.

```
public abstract int setRenderView(TextureView view);
```

Parameters

Parameter	Type	Description
view	TextureView	The player's rendering view

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

startLivePlay

This API is used to start playing audio/video streams.

```
public abstract int startLivePlay(String url);
```

Note:
Since v10.7, `startPlay` has been replaced by `startLivePlay`, and you need to call `V2TXLivePremier#setLicence` or `TXLiveBase#setLicence` to set the license to use the live playback feature (you only need to set the license once). Otherwise, playback will fail (black screen). You can use a live stream publishing license, UGSV license, or video playback license to activate the live playback feature. If you don't have any of the licenses, you can buy one or apply for a trial license for free.

Parameters

Parameter	Type	Description
url	String	Playback URL of audio/video streams, which supports RTMP, HTTP-FLV, TRTC, and WebRTC

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful
`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the URL is invalid.
`V2TXLIVE_ERROR_REFUSED` : RTC does not support using the same stream ID for publishing and playback on the same device.

stopPlay

This API is used to stop playing audio/video streams.

```
public abstract int stopPlay();
```

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

isPlaying

This API is used to get whether the player is playing streams.

```
public abstract int isPlaying();
```

Response

Whether streams are being played

`1` : yes

`0` : no

Video APIs

setRenderRotation

This API is used to set the rotation of images displayed by the player.

```
public abstract int setRenderRotation(V2TXLiveRotation rotation);
```

Parameters

Parameter	Type	Description
rotation	V2TXLiveRotation	The degrees by which the image is rotated. Default value: `V2TXLiveRotation0` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveRotation` enumerated values

Value	Description
V2TXLiveRotation0	No rotation
V2TXLiveRotation90	Rotate 90 degrees clockwise
V2TXLiveRotation180	Rotate 180 degrees clockwise
V2TXLiveRotation270	Rotate 270 degrees clockwise

setRenderFillMode

This API is used to set the image fill mode.

```
public abstract int setRenderFillMode(V2TXLiveFillMode mode);
```

Parameters

Parameter	Type	Description
mode	V2TXLiveFillMode	The image fill mode. Default value: `V2TXLiveFillModeFit` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveFillMode` enumerated values

Value	Description
V2TXLiveFillModeFit	The image fits the screen without cropping. If the aspect ratio of the image and the screen do not match, there will be black bars.
V2TXLiveFillModeFill	The image fills the entire screen, without black bars. If the aspect ratio of the image and screen do not match, the parts of the image that don't fit will be cropped.

pauseVideo

This API is used to pause playing video streams.

```
public abstract int pauseVideo();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

resumeVideo

This API is used to resume playing video streams.

```
public abstract int resumeVideo();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

snapshot

This API is used to take a screenshot of streamed video.

Note:

After `V2TXLIVE_OK` is returned, you can get the screenshot taken in the `V2TXLivePlayerObserver.onSnapshotComplete` callback.

```
public abstract int snapshot();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_REFUSED` : failed to take a screenshot because playback has stopped.

enableObserveVideoFrame

This API is used to set custom video rendering. You can use this API to obtain each frame of decoded video for custom rendering.

Note:

After custom rendering is enabled, you can get video frames in the `V2TXLivePlayerObserver.onRenderVideoFrame` callback.

```
public abstract int enableObserveVideoFrame(
        boolean enable,
        V2TXLivePixelFormat pixelFormat,
        V2TXLiveBufferType bufferType);
```

Parameters

Parameter	Type	Description
enable	Boolean	Whether to enable custom rendering. Default value: `false` .
pixelFormat	V2TXLivePixelFormat	Pixel format of the video called back for custom rendering
bufferType	V2TXLiveBufferType	Buffer type of the video called back for custom rendering

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_NOT_SUPPORTED` : unsupported pixel format or data format.

`V2TXLivePixelFormat` enumerated values

Value	Description
V2TXLivePixelFormatUnknown	Unknown
V2TXLivePixelFormatI420	YUV420P (I420)
V2TXLivePixelFormatTexture2D	OpenGL 2D texture

`V2TXLiveBufferType` enumerated values

Value	Description
V2TXLiveBufferTypeUnknown	Unknown
V2TXLiveBufferTypeByteBuffer	Direct buffers. This type is for I420 and other buffers and is used at the native layer.
V2TXLiveBufferTypeByteArray	Byte arrays. This type is for I420 and other buffers and is used at the Java layer.
V2TXLiveBufferTypeTexture	Texture ID, which allows direct operation. It delivers the best performance and has the smallest impact on video quality.

Audio APIs

pauseAudio

This API is used to pause playing audio streams.

```
public abstract int pauseAudio();
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

resumeAudio

This API is used to resume playing audio streams.

```
public abstract int resumeAudio();
```

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

setPlayoutVolume

This API is used to set volume.

```
public abstract int setPlayoutVolume(int volume);
```

Parameters

Parameter	Type	Description
volume	int	Volume. Value range: 0-100. Default value: `100` .

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

enableVolumeEvaluation

This API is used to enable the volume reminder for playback.

Note:

After enabling the volume reminder, you can get the SDK's volume evaluation through the `V2TXLivePlayerObserver.onPlayoutVolumeUpdate` callback.

```
public abstract int enableVolumeEvaluation(int intervalMs);
```

Parameters

Parameter	Type	Description
intervalMs	int	Interval (ms) for triggering the `onPlayoutVolumeUpdate` callback. The minimum interval allowed is 100 ms. If the value is `0` (default) or smaller, the callback is disabled. `300` is recommended.

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

Other APIs

setCacheParams

This API is used to set the minimum and maximum cache time (seconds) for auto adjustment by the player.

```
public abstract int setCacheParams(float minTime, float maxTime);
```

Parameters

Parameter	Type	Description
minTime	float	The minimum cache time for auto adjustment by the player. The value must be greater than `0` . Default value: `1`
maxTime	float	The maximum cache time for auto adjustment by the player. The value must be greater than `0` . Default value: `5`

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

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

`V2TXLIVE_ERROR_REFUSED` : you cannot change the cache policy while streams are being played.

showDebugView

This API is used to set whether to show the debug view for the player status information.

```
public abstract void showDebugView(boolean isShow);
```

Parameters

Parameter	Type	Description
isShow	boolean	Whether to show the debug view. Default value: `NO` .

V2TXLivePlayerObserver

Last updated：2022-10-13 11:40:07

Overview

Callbacks of Tencent Cloud's live stream player.

Features

You can use `V2TXLivePlayerObserver` to receive callbacks of V2TXLivePlayer including player status, playback volume, first audio/video frame, statistics, warning and error messages, etc.

Basic Callback APIs

onError

Callback for error

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

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
code	int	Error code
msg	String	Error message
extraInfo	Bundle	Extra information

onWarning

Callback for warning

```
public void onWarning(V2TXLivePlayer player, int code, String msg, Bundle extraInfo
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
code	int	Warning code

msg	String	Warning message
extraInfo	Bundle	Extra information

onConnected

Callback for successfully connecting to the server

```
public void onConnected(V2TXLivePlayer player, Bundle extraInfo)
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
extraInfo	Bundle	Extra information

Video Callback APIs

onVideoResolutionChanged

Callback for change of player resolution

```
public void onVideoResolutionChanged(V2TXLivePlayer player, int width, int height)
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
width	int	Video width
height	int	Video height

onVideoLoading

Callback for loading video

```
public void onVideoLoading(V2TXLivePlayer player, Bundle extraInfo)
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
extraInfo	Bundle	Extra information

onVideoPlaying

Callback for video playback

```
public void onVideoPlaying(V2TXLivePlayer player, boolean firstPlay, Bundle extraIn
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
firstPlay	boolean	Whether it is the first playback
extraInfo	Bundle	Extra information

onSnapshotComplete

Callback for a screenshot taken

```
public void onSnapshotComplete(V2TXLivePlayer player, Bitmap image)
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
image	Bitmap *	The video image captured

onRenderVideoFrame

Callback for custom video rendering

Note:

You will receive this callback after calling `[V2TXLivePlayer enableCustomRendering:pixelFormat:bufferType:]` to enable custom video rendering.

```
public void onRenderVideoFrame(V2TXLivePlayer player, V2TXLiveVideoFrame videoFrame
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
videoFrame	V2TXLiveVideoFrame	Video frame

Audio Callback APIs

onAudioLoading

Callback for loading audio

```
public void onAudioLoading(V2TXLivePlayer player, Bundle extraInfo)
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
extraInfo	Bundle	Extra information

onAudioPlaying

Callback for audio playback

```
public void onAudioPlaying(V2TXLivePlayer player, boolean firstPlay, Bundle extraIn
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
firstPlay	boolean	Whether it is the first playback
extraInfo	Bundle	Extra information

onPlayoutVolumeUpdate

Callback of the player's volume

```
public void onPlayoutVolumeUpdate(V2TXLivePlayer player, int volume)
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
volume	int	Volume. Value range: 0-100

Statistics Callback API

onStatisticsUpdate

Callback of the player's statistics

```
public void onStatisticsUpdate(V2TXLivePlayer player, V2TXLivePlayerStatistics stat
```

Parameters

Parameter	Type	Description
player	V2TXLivePlayer	The player object sending the callback
statistics	V2TXLivePlayerStatistics	Player statistics

Demo

Last updated：2022-10-20 15:57:07

Tencent Cloud offers a straightforward and easy-to-understand API example project regarding frequently asked questions among developers. You can use it to quickly learn how to use different APIs.

Download Address

Platform	GitHub Address
Android	GitHub

Contents

The demo project covers the following features:

Basic Features:

Publishing from Camera

Publishing from Screen

Playback

Same-Room Communication

Cross-Room Communication

Advanced Features:

Dynamically Switching Rendering Control

Custom Video Capturing

Third-Party Beauty Filters

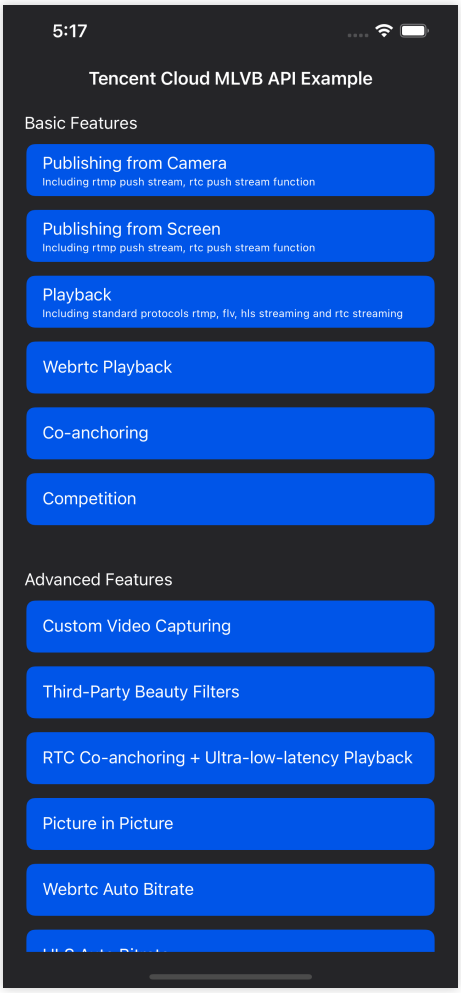
RTC-Based Co-anchoring + Ultra-Low-Latency Playback

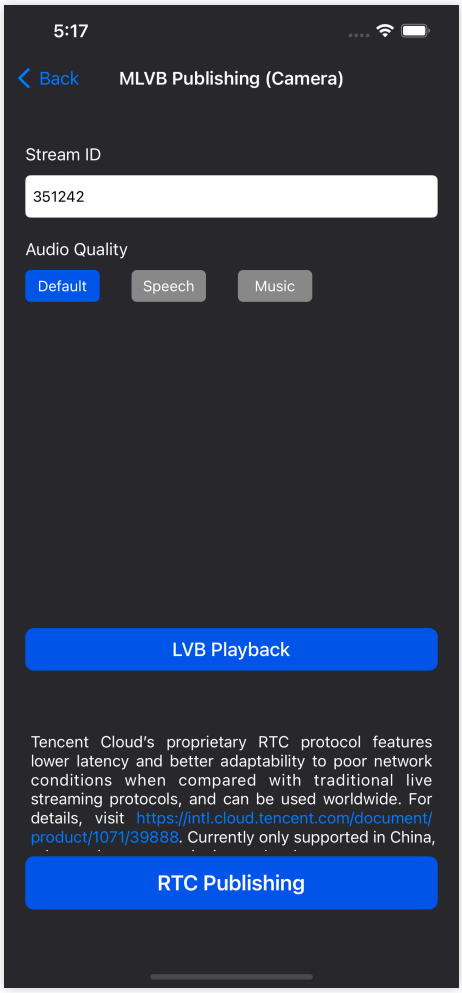
Note:

For clarity purposes, the naming of folders in the project may differ slightly from a standard Android Studio project in terms of letter case.

UI Screenshots

Main View	Publishing View	Playb





Web
TXLivePusherObserver

Last updated：2022-10-13 11:40:07

Callbacks for publishing, including publisher status, statistics, warnings, and errors.

onError

Callback for error. This callback is triggered when the publisher encounters an error.

```
onError(code: number, msg: string, extraInfo: Object): void;
```

Parameters

`code` : error code. For details, please see Error codes for publishing.

`msg` : error message

`extraInfo` : extra information. This parameter is not used currently.

onWarning

Callback for warning.

```
onWarning(code: number, msg: string, extraInfo: Object): void;
```

Parameters

`code` : error code. For details, please see Error codes for publishing.

`msg` : error message

`extraInfo` : extra information. This parameter is not used currently.

Note

For the error information returned for failure to turn the camera or mic on or record the screen, please see getUserMedia exceptions.

onCaptureFirstAudioFrame

Callback for capturing the first audio frame.

```
onCaptureFirstAudioFrame(): void;
```

onCaptureFirstVideoFrame

Callback for capturing the first video frame.

```
onCaptureFirstVideoFrame(): void;
```

onPushStatusUpdate

Callback of the publisher's connection status.

```
onPushStatusUpdate(status: number, msg: string, extraInfo: Object): void;
```

Parameters

`status` : connection status code. For details, please see Status codes for publishing.

`msg` : connection status message

`extraInfo` : extra information. This parameter is not used currently.

onStatisticsUpdate

Callback of publisher statistics, primarily WebRTC-related statistics, which may vary from browser to browser.

```
onStatisticsUpdate(statistics: Object): void;
```

Parameters

`statistics` : publishing statistics. For details, please see Publishing statistics.

Error codes for publishing

Below are the error codes returned by onError and onWarning:

Enumerated Value	Code	Description
TXLIVE_ERROR_WEBRTC_FAILED	-1	Failed to call a WebRTC API.
TXLIVE_ERROR_REQUEST_FAILED	-2	Error when requesting the server to publish streams.
TXLIVE_WARNING_CAMERA_START_FAILED	-1001	Failed to turn the camera on.
TXLIVE_WARNING_MICROPHONE_START_FAILED	-1002	Failed to turn the mic on.
TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	-1003	Failed to record the screen.
TXLIVE_WARNING_MEDIA_FILE_START_FAILED	-1004	Failed to open a local media file.
TXLIVE_WARNING_CAMERA_INTERRUPTED	-1005	Capturing from the camera was interrupted (because the device was disconnected or the user denied the access).
TXLIVE_WARNING_MICROPHONE_INTERRUPTED	-1006	Capturing from the mic was interrupted (because the device was disconnected or the user denied the access).

TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	-1007	Screen recording was interrupted (because the user clicked Chrome's built-in stop-sharing button).

Status codes for publishing

Below are the status codes returned by onPushStatusUpdate:

Enumerated Value	Code	Description
TXLIVE_PUSH_STATUS_DISCONNECTED	0	Disconnected from the server.
TXLIVE_PUSH_STATUS_CONNECTING	1	Connecting to the server.
TXLIVE_PUSH_STATUS_CONNECT_SUCCESS	2	Connected to the server.
TXLIVE_PUSH_STATUS_RECONNECTING	3	Reconnecting to the server.

Publishing statistics

The onStatisticsUpdate callback API keeps you up to date on WebRTC-related statistics during publishing. The SDK collects data once every second, and the structure of the object it returns is as follows:

Parameter	Description
timestamp	Timestamp of data collection, i.e., seconds that have elapsed since 00:00:00 UTC, 1 January 1970 till the time of data collection
video	Video statistics
audio	Audio statistics

`video`

Parameter	Description
bitrate	Video bitrate (bit/s)
framesPerSecond	Video frame rate
frameWidth	Video width, invalid for Firefox
frameHeight	Video height, invalid for Firefox
framesEncoded	Number of encoded frames
framesSent	Number of published frames, invalid for Firefox

packetsSent	Number of data packets sent
nackCount	Number of negative acknowledgements (NACK)
firCount	Number of Full Intra Requests (FIR)
pliCount	Number of Picture Loss Indications (PLI)
frameEncodeAvgTime	Average encoding time (ms), invalid for Safari and Firefox
packetSendDelay	Average time (ms) data packets are cached locally before being sent, invalid for Firefox

`audio`

Parameter	Description
bitrate	Audio bitrate (bit/s)
packetsSent	Number of data packets sent

Flutter
Overview

Last updated：2022-10-20 15:57:08

V2TXLivePlayer

Video player

For details, see V2TXLivePlayer.
The player pulls audio/video data from the specified live streaming URL and plays the data after decoding and local rendering.
The player has the following capabilities:
Playing over protocols including RTMP, HTTP-FLV, TRTC, and WebRTC.
Taking screenshots of streamed video
Delay adjustment. You can set the minimum and maximum cache time for auto adjustment by the player.
Custom video processing. You can process live video based on your project requirements before rendering and playback.

Basic SDK APIs

API	Description
addListener	Sets player callbacks.

Basic playback APIs

API	Description
setRenderViewID	Sets the player's rendering view (`ViewID`).
startLivePlay	Since v10.7, `startPlay` has been replaced by `startLivePlay` , and you need to call `V2TXLivePremier#setLicence` to set the license to use the live playback feature (you only need to set the license once). Otherwise, playback will fail (black screen). You can use a live stream publishing license, UGSV license, or video playback license to activate the live playback feature. If you don't have any of the licenses, you can buy one or apply for a trial license for free to use the feature.
stopPlay	Stops playback.
isPlaying	Gets whether playback is ongoing.

Video APIs

API	Description
setRenderRotation	Sets video rotation.
setRenderFillMode	Sets the fill mode.
pauseVideo	Pauses the player's video.
resumeVideo	Resumes the player's video.
snapshot	Takes a screenshot of the played video.

Audio APIs

API	Description
pauseAudio	Pauses the player's audio.
resumeAudio	Resumes the player's audio.
setPlayoutVolume	Sets the volume.
enableVolumeEvaluation	Enables the volume reminder for playback.

Other APIs

API	Description
setCacheParams	Sets the minimum and maximum cache time (seconds) for auto adjustment by the player.
showDebugView	Sets whether to show the debug view of player status information.

V2TXLivePlayerObserver

Player callbacks

Basic SDK callback APIs

API	Description
onError	Callback for error. This callback is returned when the player encounters an error.

onWarning	Callback for warning
onConnected	Callback for successfully connecting to the server

Video callback APIs

API	Description
onVideoPlaying	Callback for video playback
onVideoLoading	Callback for loading video
onVideoResolutionChanged	Callback for change of player resolution
onSnapshotComplete	Callback for a screenshot taken
onRenderVideoFrame	Callback for custom video rendering

Audio callback APIs

API	Description
onAudioPlaying	Callback for audio playback
onAudioLoading	Callback for loading audio
onPlayoutVolumeUpdate	Callback of the player's volume

Statistics callback APIs

API	Description
onStatisticsUpdate	Callback of player statistics

V2TXLivePusher

Stream publisher

For details, see V2TXLivePusher.

`V2TXLivePusher` encodes local audio/video and publishes the encoded data to a specified URL. It supports any publishing server.

It has the following capabilities:

Custom video capturing. You can customize audio/video data sources based on your project requirements.

Retouching, filters, and stickers. `V2TXLivePusher` integrates multiple retouching algorithms (natural & smooth) and color space filters (custom filters are supported).
QoS control technology. `V2TXLivePusher` can adapt automatically to different upstream network conditions by controlling audio/video traffic in real time based on the network conditions of hosts.
Facial feature adjustment and animated widgets. Powered by YouTu's AI facial recognition technology, `V2TXLivePusher` supports animated widgets and fine-tuning of facial features, such as eye enlarging, face slimming, and nose reshaping. You need to purchase a **YouTu license** to use these live streaming effects.

Basic SDK APIs

API	Description
addListener	Sets publisher callbacks.

Basic publishing APIs

API	Description
setRenderViewID	Sets the rendering view for local camera preview.
startPush	Starts publishing audio/video data.
stopPush	Stops publishing audio/video data.
isPushing	Gets whether publishing is ongoing.

Video APIs

API	Description
setVideoQuality	Sets video encoding parameters for publishing.
setRenderRotation	Sets video rotation for local camera preview.
setRenderMirror	Sets the mirror mode for local camera preview.
startCamera	Turns the local camera on.
stopCamera	Turns the local camera off.
startVirtualCamera	Starts image publishing.
stopVirtualCamera	Stops image publishing.
startScreenCapture	Starts screen capturing.

stopScreenCapture	Stops screen capturing.
snapshot	Takes a screenshot of the published video.
setWatermark	Sets watermarks for the publisher. Watermarking is disabled by default.
setEncoderMirror	Sets the mirror mode for encoded video.
enableCustomVideoCapture	Enables/Disables custom video capturing.
sendCustomVideoFrame	Sends the captured video data to the SDK in the custom video capturing mode.
enableCustomVideoProcess	Enables/Disables custom video processing.
sendSeiMessage	Sends an SEI message.

Beauty filter APIs

API	Description
getBeautyManager	Gets the beauty filter management object `TXBeautyManager`, which is used to set beauty filters.

Audio APIs

API	Description
startMicrophone	Turns the mic on.
stopMicrophone	Turns the mic off.
setAudioQuality	Sets the quality of published audio.
enableVolumeEvaluation	Enables the volume reminder for capturing.

Audio effect APIs

API	Description
getAudioEffectManager	Gets the audio effect management object.

Device management APIs

API	Description
getDeviceManager	Gets the device management object.

Other APIs

API	Description
setProperty	Calls an advanced API of `V2TXLivePusher`.
setMixTranscodingConfig	Sets On-Cloud MixTranscoding parameters.
showDebugView	Sets whether to display the dashboard.

V2TXLivePusherObserver

Basic SDK callback APIs

API	Description
onError	Callback for error. This callback is returned when the publisher encounters an error.
onWarning	Callback for warning

Video callback APIs

API	Description
onPushStatusUpdate	Callback of the publisher's connection status
onSnapshotComplete	Callback for a screenshot taken
onProcessVideoFrame	Callback for custom video processing
onGLContextDestroyed	Callback for destroying the OpenGL context in the SDK
onCaptureFirstVideoFrame	Callback for capturing the first video frame

Audio callback APIs

API	Description
onCaptureFirstAudioFrame	Callback for capturing the first audio frame
onMicrophoneVolumeUpdate	Callback of mic capturing volume

Mixtranscoding callback APIs

API	Description
onSetMixTranscodingConfig	Callback for setting On-Cloud MixTranscoding parameters

Statistics callback APIs

API	Description
onStatisticsUpdate	Callback of publisher statistics

Publishing

V2TXLivePusher

Last updated：2022-10-20 15:57:08

Overview

Tencent Cloud's live stream publisher

Features

`V2TXLivePusher` encodes local audio/video and publishes the encoded data to a specified URL. It supports any publishing server.

It has the following capabilities:

Custom video capturing. You can customize audio/video data sources based on your project requirements.

Retouching, filters, and stickers. `V2TXLivePusher` integrates multiple retouching algorithms (natural & smooth) and color space filters (custom filters are supported).

QoS control technology. `V2TXLivePusher` can adapt automatically to different upstream network conditions by controlling audio/video traffic in real time based on the network conditions of hosts.

Facial feature adjustment and animated widgets. Powered by YouTu's AI facial recognition technology, `V2TXLivePusher` supports animated widgets and fine-tuning of facial features, such as eye enlarging, face slimming, and nose reshaping. You need to purchase a **YouTu license** to use these live streaming effects.

Basic SDK APIs

initWithLiveMode

This API is used to initialize the publisher.

```
V2TXLivePusher(V2TXLiveMode liveMode)
```

Parameters

Parameter	Type	Description
liveMode	V2TXLiveMode	Publishing protocol: RTMP (default) or ROOM

addListener

This API is used to set the callbacks of the publisher. After the setting, you can listen for callback events of `V2TXLivePusher`, including publisher status, volume, statistics, warning and error messages, etc.

```
void addListener(V2TXLivePusherObserver func)
```

Parameters

Parameter	Type	Description
observer	V2TXLivePusherObserver	Target object for the publisher's callbacks. For details, see V2TXLivePusherObserver.

Basic Publishing APIs

setRenderViewID

This API is used to set the view for local camera preview. Images captured by the local camera are displayed on the view passed in after retouching, facial feature adjustments, and filter application.

```
Future<V2TXLiveCode> setRenderViewID(int viewID)
```

Parameters

Parameter	Type	Description
viewID	int	The view ID, which is returned by `V2TXLiveVideoWidget.onViewCreated(int viewID)` .

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

startPush

This API is used to start publishing audio/video data.

```
Future<V2TXLiveCode> startPush(String url)
```

Parameters

Parameter	Type	Description
url	String	The URL to which data is published. Any publishing server is supported.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the URL is invalid.

`V2TXLIVE_ERROR_REFUSED` : you cannot use the same stream ID for publishing and playback on the same device.

stopPush

This API is used to stop publishing audio/video data.

```
Future<V2TXLiveCode> stopPush()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

isPushing

This API is used to get whether the publisher is publishing streams.

```
Future<V2TXLiveCode> isPushing()
```

Response

Whether streams are being played

`1` : yes

`0` : no

Video APIs

setVideoQuality

This API is used to set the video encoding parameters for publishing.

```
Future<V2TXLiveCode> setVideoQuality(V2TXLiveVideoEncoderParam param)
```

Parameters

Parameter	Type	Description
param	V2TXLiveVideoEncoderParam	Video encoding parameters

setRenderRotation

This API is used to set the rotation of local camera preview.

```
Future<V2TXLiveCode> setRenderRotation(V2TXLiveRotation rotation)
```

Parameters

Parameter	Type	Description
rotation	V2TXLiveRotation	The degrees by which the image is rotated. Default value: `v2TXLiveRotation0` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveRotation` enumerated values

Value	Description
v2TXLiveRotation0	No rotation
v2TXLiveRotation90	Rotate 90 degrees clockwise
v2TXLiveRotation180	Rotate 180 degrees clockwise
v2TXLiveRotation270	Rotate 270 degrees clockwise

setRenderMirror

This API is used to set the mirror mode of local camera preview.

```
Future<V2TXLiveCode> setRenderMirror(V2TXLiveMirrorType mirrorType)
```

Parameters

Parameter	Type	Description
mirrorType	V2TXLiveMirrorType	The mirror mode of the camera. Default value: `v2TXLiveMirrorTypeAuto` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLiveMirrorType` enumerated values

Value	Description
v2TXLiveMirrorTypeAuto	The default value. In this mode, the front camera is mirrored, but the rear camera is not.
v2TXLiveMirrorTypeEnable	Both the front and rear cameras are mirrored.
v2TXLiveMirrorTypeDisable	Neither the front nor rear camera is mirrored.

startCamera

This API is used to turn the local camera on.

Note:

The `startVirtualCamera` , `startCamera` , and `startScreenCapture` APIs are mutually exclusive when you use them with the same pusher instance. For example, if you call `startCamera` first and then call `startVirtualCamera` , the SDK will stop publishing camera data and publish an image instead.

```
Future<V2TXLiveCode> startCamera(bool frontCamera)
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopCamera

This API is used to turn the local camera off.

```
Future<void> stopCamera()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

startVirtualCamera

This API is used to publish an image.

Note:

The `startVirtualCamera`, `startCamera`, and `startScreenCapture` APIs are mutually exclusive when you use them with the same pusher instance. For example, if you call `startCamera` first and then call `startVirtualCamera`, the SDK will stop publishing camera data and publish an image instead.

```
Future<V2TXLiveCode> startVirtualCamera(String type, String imageUrl)
```

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

Parameter	Type	Description
type	String	Valid values: `network` (an image from the internet), `file` (a local image).
imageUrl	String	Image URL

stopVirtualCamera

This API is used to stop publishing images.

```
Future<V2TXLiveCode> stopVirtualCamera()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

startScreenCapture

This API is used to start screen recording.

Note:

To start screen recording on iOS, instead of using this API, you need to do the following.

The `startVirtualCamera`, `startCamera`, and `startScreenCapture` APIs are mutually exclusive when you use them with the same pusher instance. For example, if you call `startCamera` first and then call `startVirtualCamera`, the SDK will stop publishing camera data and publish an image instead.

1. First, use Broadcast Upload Extension to start screen recording.
2. Then, call `enableCustomVideoCapture` to enable custom capturing.
3. At last, call `sendCustomVideoFrame` to send the screen images captured by Broadcast Upload Extension.

```
Future<V2TXLiveCode> startScreenCapture(String appGroup)
```

Parameters

Parameter	Type	Description
appGroup	String	The App Group ID shared by the host app and Broadcast Upload Extension. You can set it to `nil`, but for better reliability, we recommend you set it as instructed in our documentation.

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

stopScreenCapture

This API is used to stop screen recording.

```
Future<V2TXLiveCode> stopScreenCapture()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

snapshot

This API is used to take a screenshot of the local video while it is being published.

Note:

After `V2TXLIVE_OK` is returned, you can get the screenshot taken in the `V2TXLivePusherObserver.onSnapshotComplete` callback.

```
Future<V2TXLiveCode> snapshot()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK`: successful

`V2TXLIVE_ERROR_REFUSED`: failed to take a screenshot because publishing has stopped.

setWatermark

This API is used to set a watermark for the publisher. Watermarking is disabled by default.

```
Future<V2TXLiveCode> setWatermark(String type, String image, double x,
    double y, double scale)
```

Parameters

Parameter	Type	Description
type	String	Valid values: `network` (an image from the internet), `file` (a local image).
image	String	Image URL
x	double	Watermark coordinate
y	double	Watermark coordinate
scale	double	Percentage by which the image is resized

setEncoderMirror

This API is used to set the mirror mode of encoded images.

```
Future<V2TXLiveCode> setEncoderMirror(bool mirror)
```

Parameters

Parameter	Type	Description
mirror	Boolean	Whether to mirror encoded images. Default value: `false`.

enableCustomVideoCapture

This API is used to enable/disable custom video capturing. In the custom video capturing mode, the SDK stops capturing images from the camera, but continues to encode and send images.

```
Future<V2TXLiveCode> enableCustomVideoCapture(bool enable)
```

Parameters

Parameter	Type	Description
enable	bool	Whether to enable custom video capturing. Default value: `false`.

sendCustomVideoFrame

This API is used to send the captured custom video data to the SDK.

```
Future<V2TXLiveCode> sendCustomVideoFrame(V2TXLiveVideoFrame videoFrame)
```

Parameters

Parameter	Type	Description
videoFrame	V2TXLiveVideoFrame	Video frames sent to the SDK

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because the video data is invalid.

`V2TXLIVE_ERROR_REFUSED` : failed. You must call `enableCustomVideoCapture` to enable custom video capturing first.

enableCustomVideoProcess

This API is used to enable/disable custom video processing.

```
Future<V2TXLiveCode> enableCustomVideoProcess(bool enable,
    V2TXLivePixelFormat pixelFormat, V2TXLiveBufferType bufferType)
```

Parameters

Parameter	Type	Description
enable	bool	Whether to enable custom video processing. Default value: `false` .
pixelFormat	V2TXLivePixelFormat	Pixel format of video frames
bufferType	V2TXLiveBufferType	Buffer type of video data

`V2TXLivePixelFormat` enumerated values

Value	Description
v2TXLivePixelFormatUnknown	Unknown
v2TXLivePixelFormatI420	YUV420P (I420)
v2TXLivePixelFormatTexture2D	OpenGL 2D texture

`V2TXLiveBufferType` enumerated values

Value	Description
v2TXLiveBufferTypeUnknown	Unknown

v2TXLiveBufferTypeByteBuffer	Direct buffers. This type is for I420 and other buffers and is used at the native layer.
v2TXLiveBufferTypeByteArray	Byte arrays. This type is for I420 and other buffers and is used at the Java layer.
v2TXLiveBufferTypeTexture	Texture ID, which allows direct operation. It delivers the best performance and has the smallest impact on video quality.

sendSeiMessage

This API is used to send an SEI message. V2TXLivePlayer can receive SEI messages via the `onReceiveSeiMessage` callback of V2TXLivePlayerObserver.

```
Future<V2TXLiveCode> sendSeiMessage(int payloadType, Uint8List data)
```

Parameters

Parameter	Type	Description
payloadType	int	Data type. Valid values: `5` , `242` (recommended).
data	Uint8List	Data to be sent

Response

V2TXLiveCode:
`V2TXLIVE_OK` : successful

Beauty Filter APIs

getBeautyManager

This API is used to get the beauty filter manager TXBeautyManager.
You can do the following using the beauty filter manger:
Set the beauty filter style and apply effects including skin brightening, rosy skin, eye enlarging, face slimming, chin slimming, chin lengthening/shortening, face shortening, nose narrowing, eye brightening, teeth whitening, eye bag removal, wrinkle removal, and smile line removal.
Adjust the hairline, eye spacing, eye corners, lip shape, nose wings, nose position, lip thickness, and face shape.
Apply animated effects such as face widgets (resources).
Add makeup effects.
Recognize gestures.

```
TXBeautyManager getBeautyManager()
```

Audio APIs

startMicrophone

This API is used to turn the mic on.

```
Future<V2TXLiveCode> startMicrophone()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

stopMicrophone

This API is used to turn the mic off.

```
Future<V2TXLiveCode> stopMicrophone()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

setAudioQuality

This API is used to set audio quality.

```
Future<V2TXLiveCode> setAudioQuality(V2TXLiveAudioQuality quality)
```

Parameters

Parameter	Type	Description
quality	V2TXLiveAudioQuality	Audio quality

`V2TXLiveAudioQuality` enumerated values

Value	Description
v2TXLiveAudioQualitySpeech	Speech. Audio sample rate: 16 kHz; sound channel: mono; bitrate: 16 Kbps This is designed for call scenarios such as online conferencing and

	audio calls.
v2TXLiveAudioQualityDefault	Default. Audio sample rate: 48 kHz; sound channel: mono; bitrate: 50 Kbps This is the default audio quality of the SDK and is recommended.
v2TXLiveAudioQualityMusic	Music. Audio sample rate: 48 kHz; sound channel: dual; bitrate: 128 Kbps This is designed for music-oriented scenarios with hi-fi requirements, such as karaoke and live music streaming.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_REFUSED` : you cannot change audio quality settings when streams are being published.

enableVolumeEvaluation

This API is used to enable the volume reminder for audio capturing.

Note:

After enabling the volume reminder, you can get the volume measured by the SDK in the `V2TXLivePusherObserver.onMicrophoneVolumeUpdate` callback.

```
Future<V2TXLiveCode> enableVolumeEvaluation(int intervalMs)
```

Parameters

Parameter	Type	Description
intervalMs	int	Interval (ms) for volume callbacks. The minimum interval allowed is 100 ms. If the value is `0` (default) or smaller, the callback is disabled. `300` is recommended.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

Audio Effect APIs

getAudioEffectManager

This API is used to get the audio effect manager TXAudioEffectManager.

You can do the following using `TXAudioEffectManager` :

Set the volume of audio captured by the mic.

Set reverb and voice changing effects.

Enable in-ear monitoring and set its volume.

Add background music and specify how to play it.

```
TXAudioEffectManager getAudioEffectManager()
```

Device Management APIs

getDeviceManager

This API is used to get the device manager TXDeviceManager.

You can do the following using `TXDeviceManager` :

Switch between the front and rear cameras.

Enable autofocus.

Set camera zoom level.

Enable/Disable flashlight.

```
TXDeviceManager getDeviceManager()
```

Other APIs

setProperty

This API is used to call the advanced APIs of `V2TXLivePusher` .

```
Future<V2TXLiveCode> setProperty(String key, String value)
```

Parameters

Parameter	Type	Description
key	String	Key of the advanced API to call
value	String	Parameters required by the advanced API

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : operation failed because `key` is empty.

setMixTranscodingConfig

This API is used to set On-Cloud MixTranscoding parameters. If you have enabled relay to CDN on the **Function Configuration** page of the TRTC console, each anchor will have a default CDN address. There may be multiple anchors in a room, each sending their own video and audio, but the audience needs only one live stream. Therefore, you need to mix multiple audio/video streams into one standard live stream. This is MixTranscoding.

```
Future<V2TXLiveCode> setMixTranscodingConfig(V2TXLiveTranscodingConfig? config)
```

Parameters

Parameter	Type	Description
config	V2TXLiveTranscodingConfig	On-Cloud MixTranscoding configuration

V2TXLiveTranscodingConfig

Value	Description
v2TXLiveBufferTypeUnknown	Unknown
v2TXLiveBufferTypeByteBuffer	Direct buffers. This type is for I420 and other buffers and is used at the native layer.
v2TXLiveBufferTypeByteArray	Byte arrays. This type is for I420 and other buffers and is used at the Java layer.
v2TXLiveBufferTypeTexture	Texture ID, which allows direct operation. It delivers the best performance and has the smallest impact on video quality.

showDebugView

This API is used to set whether to display the dashboard.

```
Future<V2TXLiveCode> showDebugView(bool isShow)
```

Parameters

Parameter	Type	Description
isShow	bool	Whether to display the dashboard. Default value: `false` .

V2TXLivePusherObserver

Last updated：2022-10-20 15:57:08

Overview

Callbacks of Tencent Cloud's live stream publisher

Features

You can use `V2TXLivePusherObserver` to receive callbacks of V2TXLivePusher, including the publisher status, first audio/video frame, statistics, and warning and error messages.

Basic Callback APIs

onError

Callback for error. This callback is triggered when the publisher encounters an error.

```
V2TXLivePusherListenerType.onError
```

Parameters

Parameter	Type	Description
code	V2TXLiveCode	Error code
msg	String	Error message
extraInfo	Map	Extra information

onWarning

Callback for warning.

```
V2TXLivePusherListenerType.onWarning
```

Parameters

Parameter	Type	Description
code	V2TXLiveCode	Warning code
msg	String	Warning message
extraInfo	Map	Extra information

Video Callback APIs

onPushStatusUpdate

Callback of the publisher's connection status.

```
V2TXLivePusherListenerType.onPushStatusUpdate
```

Parameters

Parameter	Type	Description
status	V2TXLivePushStatus	Status code
msg	String	Status message
extraInfo	Map	Extra information

`V2TXLivePushStatus` enumerated values

Value	Description
V2TXLivePushStatusDisconnected: 0	Disconnected from the server
V2TXLivePushStatusConnecting: 1	Connecting to the server
V2TXLivePushStatusConnectSuccess: 2	Connected to the server
V2TXLivePushStatusReconnecting: 3	Reconnecting to the server

onSnapshotComplete

Callback for a screenshot taken

```
V2TXLivePusherListenerType.onSnapshotComplete
```

Parameters

Parameter	Type	Description
image	Uint8List	Screenshot taken

onProcessVideoFrame

Callback for custom video processing.

Note:

You will receive this callback after you call `V2TXLivePusher.enableCustomVideoProcess(bool enable, V2TXLivePixelFormat pixelFormat, V2TXLiveBufferType bufferType)` to enable custom video processing.

```
V2TXLivePusherListenerType.onProcessVideoFrame
```

Parameters

Parameter	Type	Description
srcFrame	Map	For images before processing
dstFrame	Map	For images after processing

onGLContextDestroyed

Callback for a GL context for custom video processing being destroyed.

```
V2TXLivePusherListenerType.onGLContextDestroyed
```

onCaptureFirstVideoFrame

Callback for capturing the first video frame.

```
V2TXLivePusherListenerType.onCaptureFirstVideoFrame
```

Audio Callback APIs

onCaptureFirstAudioFrame

Callback for capturing the first audio frame.

```
V2TXLivePusherListenerType.onCaptureFirstAudioFrame
```

onMicrophoneVolumeUpdate

Callback of mic capturing volume.

```
V2TXLivePusherListenerType.onMicrophoneVolumeUpdate
```

Statistics Callback API

onStatisticsUpdate

Callback of publisher statistics.

```
V2TXLivePusherListenerType.onStatisticsUpdate
```

Parameters

Parameter	Type	Description
statistics	Map	Publisher statistics

MixTranscoding Callback API

onSetMixTranscodingConfig

Callback for setting On-Cloud MixTranscoding parameters.

Note:

You will receive this callback after you call `V2TXLivePusher.setMixTranscodingConfig(V2TXLiveTranscodingConfig config)` to set On-Cloud MixTranscoding parameters.

```
V2TXLivePusherListenerType.onSetMixTranscodingConfig
```

Parameter	Type	Description
code	V2TXLiveCode	`0` : successful; other values: failed
msg	String	Error message

Playback
V2TXLivePlayer

Last updated：2022-10-20 15:57:08

Overview

Tencent Cloud's live stream player

The player pulls audio/video data from the specified live streaming URL and plays the data after decoding and local rendering.

Features

The player has the following capabilities:

Playing over protocols including RTMP, HTTP-FLV, TRTC, and WebRTC.

Taking screenshots of streamed video

Delay adjustment. You can set the minimum and maximum cache time for auto adjustment by the player.

Custom video processing. You can process live video based on your project requirements before rendering and playback.

Basic SDK APIs

addListener

This API is used to set the callbacks of the player. After setting, you can listen for callback events of `V2TXLivePlayer`, including player status, playback volume, first audio/video frame, statistics, warning and error messages, etc.

```
void addListener(V2TXLivePlayerObserver func)
```

Parameters

Parameter	Type	Description
observer	V2TXLivePlayerObserver	The target object for the player's callbacks. For details, see V2TXLivePlayerObserver.

Basic Playback APIs

setRenderViewID

This API is used to set the rendering view to display video.

```
Future<V2TXLiveCode> setRenderViewID(int viewID)
```

Parameters

Parameter	Type	Description
viewID	int	The rendering view ID, which is returned by `V2TXLiveVideoWidget.onViewCreated`.

Response

V2TXLiveCode:

`V2TXLIVE_OK`: Successful

startLivePlay

This API is used to start playing audio/video streams.

```
Future<V2TXLiveCode> startLivePlay(String url)
```

Note:

Since v10.7, `startPlay` has been replaced by `startLivePlay`, and you need to call `V2TXLivePremier#setLicence` or `TXLiveBase#setLicence` to set the license to use the live playback feature (you only need to set the license once). Otherwise, playback will fail (black screen). You can use a live stream publishing license, UGSV license, or video playback license to activate the live playback feature. If you don't have any of the licenses, you can buy one or apply for a trial license for free to use the feature.

Parameters

Parameter	Type	Description
url	String	The playback URL, which supports RTMP, HTTP-FLV, TRTC, and WebRTC.

Response

V2TXLiveCode:

`V2TXLIVE_OK`: Successful

`V2TXLIVE_ERROR_INVALID_PARAMETER`: Operation failed because the URL is invalid.

`V2TXLIVE_ERROR_REFUSED`: RTC does not support using the same stream ID for publishing and playback on the same device.

stopPlay

This API is used to stop playing audio/video streams.

```
Future<V2TXLiveCode> stopPlay()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : successful

isPlaying

This API is used to get whether the player is playing streams.

```
Future<int> isPlaying()
```

Response

Whether streams are being played

`1` : yes

`0` : no

Video APIs

setRenderRotation

This API is used to set the rotation of images displayed by the player.

```
Future<V2TXLiveCode> setRenderRotation(V2TXLiveRotation rotation)
```

Parameters

Parameter	Type	Description
rotation	V2TXLiveRotation	The degrees by which images are rotated. Default value: 0.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

`V2TXLiveRotation` enumerated values

Value	Description

v2TXLiveRotation0	No rotation
v2TXLiveRotation90	Rotate 90 degrees clockwise
v2TXLiveRotation180	Rotate 180 degrees clockwise
v2TXLiveRotation270	Rotate 270 degrees clockwise

setRenderFillMode

This API is used to set the image fill mode.

```
Future<V2TXLiveCode> setRenderFillMode(V2TXLiveFillMode mode)
```

Parameters

Parameter	Type	Description
mode	V2TXLiveFillMode	The image fill mode. Default value: `V2TXLiveFillModeFit` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

`V2TXLiveFillMode` enumerated values

Value	Description
v2TXLiveFillModeFit	The image fits the screen without cropping. If the aspect ratio of the image and the screen do not match, there will be black bars.
v2TXLiveFillModeFill	The image fills the entire screen, without black bars. If the aspect ratio of the image and screen do not match, the parts of the image that don't fit will be cropped.

pauseVideo

This API is used to pause playing video streams.

```
Future<V2TXLiveCode> pauseVideo()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

resumeVideo

This API is used to resume playing video streams.

```
Future<V2TXLiveCode> resumeVideo()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

snapshot

This API is used to take a screenshot of streamed video.

Note:

After `V2TXLIVE_OK` is returned, you can get the screenshot taken in the `V2TXLivePlayerObserver.onSnapshotComplete` callback.

```
Future<V2TXLiveCode> snapshot()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

`V2TXLIVE_ERROR_REFUSED` : Failed to take a screenshot because playback has stopped.

Audio APIs

pauseAudio

This API is used to pause playing audio streams.

```
Future<V2TXLiveCode> pauseAudio()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

resumeAudio

This API is used to resume playing audio streams.

```
Future<V2TXLiveCode> resumeAudio()
```

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

setPlayoutVolume

This API is used to set the volume.

```
Future<V2TXLiveCode> setPlayoutVolume(int volume)
```

Parameters

Parameter	Type	Description
volume	int	The volume. Value range: 0-100. Default value: `100` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

enableVolumeEvaluation

This API is used to enable the volume reminder for playback.

Note:

After enabling the volume reminder, you can get the SDK's volume evaluation through the `V2TXLivePlayerObserver.onPlayoutVolumeUpdate` callback.

```
Future<V2TXLiveCode> enableVolumeEvaluation(int intervalMs)
```

Parameters

Parameter	Type	Description
intervalMs	int	The interval (ms) for triggering the `onPlayoutVolumeUpdate` callback. The minimum interval allowed is 100 ms. If the value is `0` (default) or smaller, the callback is disabled. `300` is recommended.

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

Other APIs

setCacheParams

This API is used to set the minimum and maximum cache time (seconds) for auto adjustment by the player.

```
Future<V2TXLiveCode> setCacheParams(double minTime, double maxTime)
```

Parameters

Parameter	Type	Description
minTime	double	The minimum cache time for auto adjustment by the player. The value must be greater than `0` . Default value: `1` .
maxTime	double	The maximum cache time for auto adjustment by the player. The value must be greater than `0` . Default value: `5` .

Response

V2TXLiveCode:

`V2TXLIVE_OK` : Successful

`V2TXLIVE_ERROR_INVALID_PARAMETER` : Operation failed. `minTime` and `maxTime` must be greater than `0` .

`V2TXLIVE_ERROR_REFUSED` : You cannot change the cache policy while streams are being played.

showDebugView

This API is used to set whether to show the debug view for the player status information.

```
Future<void> showDebugView(bool isShow)
```

Parameters

Parameter	Type	Description
isShow	bool	Whether to show the debug view. Default value: `false` .

V2TXLivePlayerObserver

Last updated：2022-10-20 15:57:08

Overview

Callbacks of Tencent Cloud's live stream player.

Features

You can use `V2TXLivePlayerObserver` to receive callbacks of V2TXLivePlayer, including the player status, playback volume, first audio/video frame, statistics, and warning and error messages.

Basic Callback APIs

onError

Callback for error

```
V2TXLivePlayerListenerType.onError
```

Parameters

Parameter	Type	Description
code	V2TXLiveCode	Error code
msg	String	Error message
extraInfo	Map	Extra information

onWarning

Callback for warning

```
V2TXLivePlayerListenerType.onWarning
```

Parameters

Parameter	Type	Description
code	V2TXLiveCode	Warning code
msg	String	Warning message
extraInfo	Map	Extra information

onConnected

Callback for successfully connecting to the server

```
V2TXLivePlayerListenerType.onConnected
```

Parameters

Parameter	Type	Description
extraInfo	Map	Extra information

Video Callback APIs

onVideoPlaying

Callback for video playback

```
V2TXLivePlayerListenerType.onVideoPlaying
```

Parameters

Parameter	Type	Description
firstPlay	bool	Whether it is the first playback
extraInfo	Map	Extra information

onVideoLoading

Callback for loading video

```
V2TXLivePlayerListenerType.onVideoLoading
```

Parameters

Parameter	Type	Description
extraInfo	Map	Extra information

onVideoResolutionChanged

Callback for change of player resolution

```
V2TXLivePlayerListenerType.onVideoResolutionChanged
```

Parameters

Parameter	Type	Description
width	int	Video width
height	int	Video height

onSnapshotComplete

Callback for a screenshot taken

```
V2TXLivePlayerListenerType.onSnapshotComplete
```

Parameters

Parameter	Type	Description
image	Uint8List	Screenshot taken

onRenderVideoFrame

Callback for custom video rendering

Note:

You will receive this callback after calling `V2TXLivePlayer.enableCustomRendering(pixelFormat:bufferType:)` to enable custom video rendering.

```
V2TXLivePlayerListenerType.onRenderVideoFrame
```

Parameters

Parameter	Type	Description
videoFrame	Map	Video frames

Audio Callback APIs

onAudioPlaying

Callback for audio playback

```
V2TXLivePlayerListenerType.onAudioPlaying
```

Parameters

Parameter	Type	Description

firstPlay	bool	Whether it is the first playback
extraInfo	Map	Extra information

onAudioLoading

Callback for loading audio

```
V2TXLivePlayerListenerType.onAudioLoading
```

Parameters

Parameter	Type	Description
extraInfo	Map	Extra information

onPlayoutVolumeUpdate

Callback of the player's volume

```
V2TXLivePlayerListenerType.onPlayoutVolumeUpdate
```

Parameters

Parameter	Type	Description
volume	int	Volume. Value range: 0-100.

Statistics Callback API

onStatisticsUpdate

Callback of the player's statistics

```
V2TXLivePlayerListenerType.onStatisticsUpdate
```

Parameters

Parameter	Type	Description
statistics	Map	Player statistics

Demo

Last updated：2022-10-20 15:57:08

Tencent Cloud offers a straightforward and easy-to-understand API example project regarding frequently asked questions among developers. You can use it to quickly learn how to use different APIs.

Download Address

Platform	GitHub Address
Flutter	GitHub

Contents

The demo project covers the following features:

Publishing from Camera

Publishing from Screen

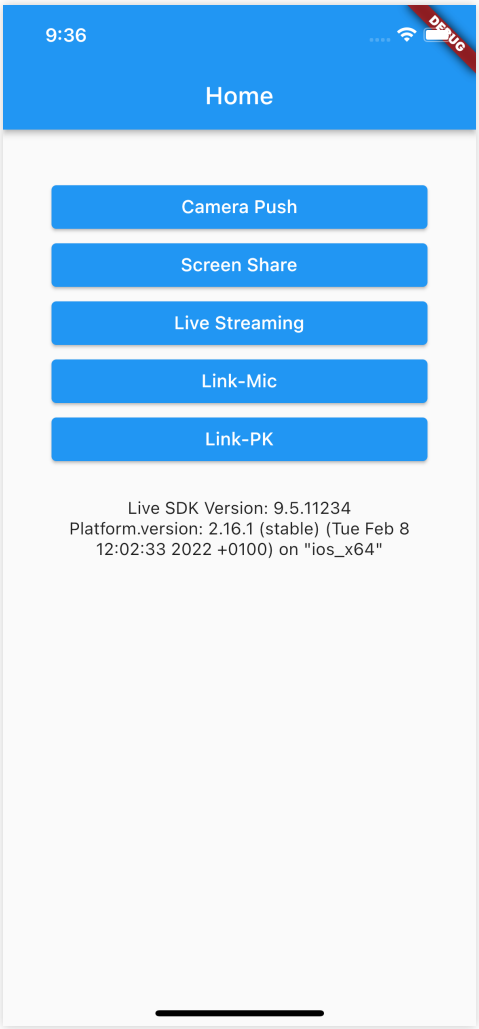
Playback

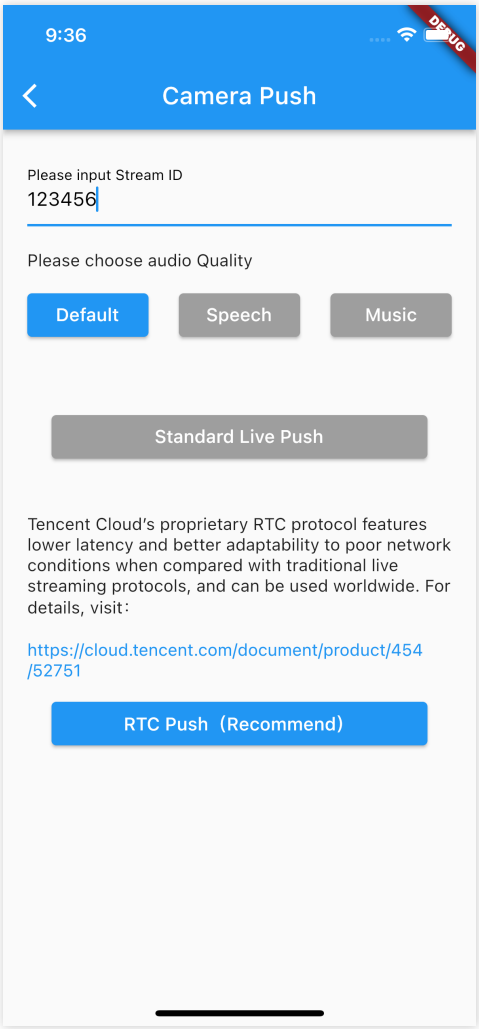
Same-Room Communication

Cross-Room Communication

UI Screenshots

Main View	Publishing View	Pl





Error Codes

Last updated：2022-10-20 15:57:08

onError

The SDK encountered a severe error (for example, invalid license or API call failure) that makes it impossible for stream publishing to continue.

Note:

Video encoding failure does not affect publishing. The SDK will handle the issue automatically to ensure successful encoding of subsequent frames.

Error Code	Event	Description
0	V2TXLIVE_OK	No error
-1	V2TXLIVE_ERROR_FAILED	A common error not yet classified
-2	V2TXLIVE_ERROR_INVALID_PARAMETER	An invalid parameter was passed to the API.
-3	V2TXLIVE_ERROR_REFUSED	The API call was rejected.
-4	V2TXLIVE_ERROR_NOT_SUPPORTED	The API cannot be called.
-5	V2TXLIVE_ERROR_INVALID_LICENSE	Failed to call the API because the license is invalid. Note: Before calling `startLivePlay` , you need to call `V2TXLivePremier#setLicence` to configure the license (you only need to configure the license once). Otherwise, playback will fail (black screen). You can use a live stream publishing license, UGSV license, or video playback license to activate the playback feature. If you don’t have any of the licenses, you can buy one or apply for a trial license for free.
-6	V2TXLIVE_ERROR_REQUEST_TIMEOUT	The request timed out.
-7	V2TXLIVE_ERROR_SERVER_PROCESS_FAILED	The server denied the request.

-8	V2TXLIVE_ERROR_DISCONNECTED	The SDK was disconnected.
-2304	V2TXLIVE_ERROR_NO_AVAILABLE_HEVC_DECODERS	Could not find an available HEVC decoder. This error code indicates that the current device does not support H.265 decoding. Please switch to H.264 for playback.

onWarning

Most warning events trigger protection or recovery logic that involves retries. In most cases, the problems can be fixed by the SDK itself. However, some warning events still need to be handled manually. Warnings serve the purpose of reminding you or your end users that an error occurred.

Error Code	Event	Description
1101	V2TXLIVE_WARNING_NETWORK_BUSY	Bad network connection: Data upload blocked due to limited upstream bandwidth.
2105	V2TXLIVE_WARNING_VIDEO_BLOCK	Stuttering during video playback.
-1301	V2TXLIVE_WARNING_CAMERA_START_FAILED	Failed to turn the camera on.
-1316	V2TXLIVE_WARNING_CAMERA_OCCUPIED	The camera is occupied. Try a different camera.
-1314	V2TXLIVE_WARNING_CAMERA_NO_PERMISSION	No access to the camera. This usually occurs on mobile devices and may be because the user denied the access.
-1302	V2TXLIVE_WARNING_MICROPHONE_START_FAILED	Failed to turn the mic on.
-1319	V2TXLIVE_WARNING_MICROPHONE_OCCUPIED	The mic is occupied. This occurs when, for example, the user is having a call on the mobile device.
-1317	V2TXLIVE_WARNING_MICROPHONE_NO_PERMISSION	No access to the mic. This

		usually occurs on mobile devices and may be because the user denied the access.
-1309	V2TXLIVE_WARNING_SCREEN_CAPTURE_NOT_SUPPORTED	The system does not support screen sharing.
-1308	V2TXLIVE_WARNING_SCREEN_CAPTURE_START_FAILED	Failed to start screen recording. If this occurs on a mobile device, it may be because the user denied the access.
-7001	V2TXLIVE_WARNING_SCREEN_CAPTURE_INTERRUPTED	Screen recording was stopped by the system.