

# Live Video Broadcasting Playing Method Product Introduction



### Copyright Notice

©2013-2018 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

### Trademark Notice



All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

### Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

# Contents

## Playing Method

### Mobile Play

### Web Play

#### Web Player TcPlayer

#### Web Player TcPlayer

#### FAQ

#### Web LVB Player 1.0

#### Web LVB Player 1.0

#### Problem solving

#### Web VOD Player 1.0

#### Web VOD Player 1.0

#### Problem solving

# Playing Method

## Mobile Play

Last updated : 2018-06-27 11:08:50

### Quick integration

You can quickly integrate LVB playback feature into your existing App by following the steps below:

- Step 1: [Activate](#) LVB service
- Step 2: [Download](#) RTMP SDK package
- Step 3: Complete interfacing process by referring to relevant documents ([iOS](#) & [Android](#))

### Complete solution

[Mobile LVB](#) is a collection of mobile LVB solutions. Presented in a form of free source code, it aims to show you how to utilize Tencent Cloud services such as LVB, VOD, IM and COS to build an LVB solution suitable for you.

# Web Play

## Web Player TcPlayer

## Web Player TcPlayer

Last updated : 2018-08-02 16:27:28

## Overview

TcPlayer Lite is mainly used to play audio/video streams in mobile phone browsers and PC browsers. With this player, you can share your videos on social platforms such as WeChat Moments and Weibo without installing other Apps. This document is intended for developers with basic knowledge in JavaScript.

## Basics

The following are the basics you must learn before getting started:

- **LVB and VOD**

LVB video source is generated in real time. Once the VJ stops broadcasting, the LVB URL becomes invalid, and since the live streams are played in real time, no progress bar is displayed on the player during the playback.

VOD video source is a file on the server, which can be played at any time as long as it has not been deleted by the provider. Since the entire video file is stored on the server, a progress bar is displayed during the playback.

- **Supported Protocols**

Playing videos using the web player depends on browsers, rather than the codes in webpages. For this reason, the compatibility may be lower than we expected. The fact is that **not all mobile browsers can yield expected performance results**. Some of the mobile browsers don't even support video playback at all. The most common video source URLs used to play videos on webpages are URLs ending with "m3u8". We call them HLS (HTTP Live Streaming), a standard from Apple. At present, this format has the best compatibility with various mobile browser products thanks to Apple's influence. However, this format has a drawback: a big delay of 20-30 seconds. Even so, we don't have any other choices for mobile browsers.

This situation looks better on PC, because PC browsers still use FLASH widgets, which supports various video source formats. Besides, the FLASH widgets for different browsers are all developed by Adobe, so they have good

compatibility.

Video protocol	Application	URL example	PC Browser	Mobile Browser
HLS (m3u8)	LVB	http://2157.liveplay.myqcloud.com/2157_358535a.m3u8	support	support
HLS (m3u8)	VOD	http://200002949.vod.myqcloud.com/200002949_b6ffc.f0.m3u8	support	support
FLV	LVB	http://2157.liveplay.myqcloud.com/2157_358535a.flv	support	not support
FLV	VOD	http://200002949.vod.myqcloud.com/200002949_b6ffc.f0.flv	support	not support
RTMP	LVB only	rtmp://2157.liveplay.myqcloud.com/2157_280d88	support	not support
MP4	VOD only	http://200002949.vod.myqcloud.com/200002949_b6ffc.f0.MP4	support	support

## Interfacing

### Step 1: Prepare the page

Introduce the initialization script to the page on which you want to play videos (including PC or H5)

```
<script src="//imgcache.qq.com/open/qcloud/video/vcplayer/TcPlayer-2.2.1.js" charset="utf-8"></script>;
```

**Note:**

You can't perform debugging directly using the local pages because TcPlayer Lite cannot handle cross-domain issues in such situations.

### Step 2: Place Container into HTML

Place a "container" in the page where you want to display the player. That is, place a div and give it a name such as: id\_test\_video. When this is done, all videos will be rendered in this container. You can control the container size through the div attributes. Example code is as follows:

```
<div id="id_test_video" style="width:100%; height:auto;"></div>
```

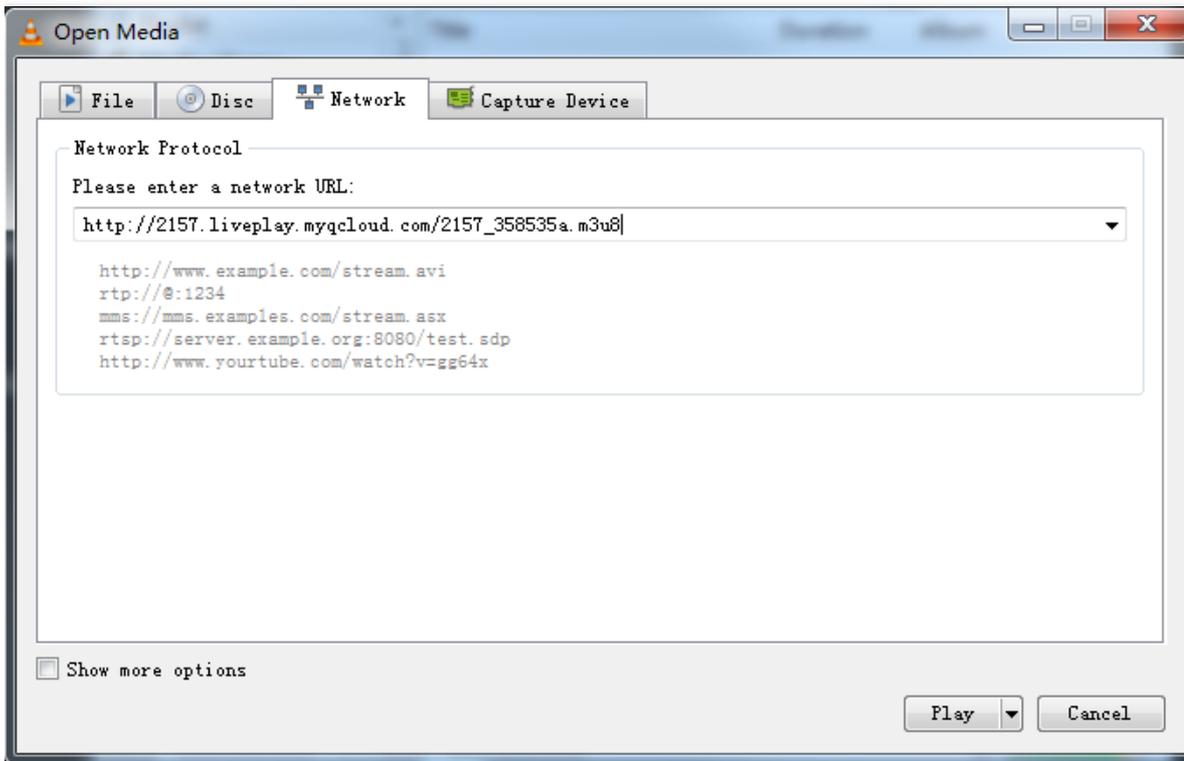
### Step 3: Prepare for Video Playback

The next step is to write the JavaScript code, which is used to pull audio/video streams from the specified URL and display the video in the container added in Step 2.

#### 3.1 A simple playback

A typical LVB URL is shown below, which is based on the HLS (m3u8) protocol. Players such as VLC can directly open this URL to view the video as long as the VJ is still broadcasting:

`http://2157.liveplay.myqcloud.com/2157_358535a.m3u8 //m3u8 playback URL`



Now, we need to play the video of this URL on mobile phone browsers, we can write the JavaScript code like this:

```
var player = new TcPlayer('id_test_video', {  
  "m3u8": "http://2157.liveplay.myqcloud.com/2157_358535a.m3u8", //Replace the URL with a valid playback URL.  
  "autoplay": true, //Most mobile browsers including Safari on iOS do not support auto video playback.  
  "coverpic": "http://www.test.com/myimage.jpg",  
  "width": '480', //Video display width. Use the video definition width if possible.  
  "height": '320', //Video display height. Use the video definition height if possible.  
});
```

This code segment can be used to play LVB videos based on HLS (m3u8) protocol on PC and mobile phone browsers. However, while videos based on HLS (m3u8) protocol have good compatibility (certain Android phones still do not support them), a big delay over 20 seconds is expected on them.

### 3.2 Achieve lower delay on PC

As PC browsers support flash, the JavaScript code is as follows:

```
var player = new TcPlayer('id_test_video', {  
  "m3u8": "http://2157.liveplay.myqcloud.com/2157_358535a.m3u8",  
  "flv": "http://2157.liveplay.myqcloud.com/live/2157_358535a.flv", //Add an FLV playback URL to play videos on PC. Re  
  place the URL with a valid playback URL.  
  "autoplay": true, //Most mobile browsers including Safari on iOS do not support auto video playback.  
  "coverpic": "http://www.test.com/myimage.jpg",  
  "width": '480', //Video display width. Use the video definition width if possible.
```

```
"height" : '320' //Video display height. Use the video definition height if possible.
});
```

Compared with the previous code, we add an FLV playback URL. When the TcPlayer detects that the browser is a PC browser, it will select the FLV linkage to achieve lower delay. The condition is that both FLV and HLS (m3u8) address can stream the video. You don't have to consider this problem if you use Tencent Cloud VOD service because every VOD channel of Tencent Cloud supports three protocols: FLV, RTMP and HLS (m3u8).

### 3.3 Why won't the videos play?

The reasons why videos cannot be played are as follows:

- **Reason 1: Invalid Video Source**

If it's an LVB URL, be sure to check whether the VJ has stopped streaming and isn't in "Broadcasting" status. You can send a notification to viewers via a floating window: "The VJ is offline".

If it's a VOD URL, you need to check whether the file to be played is still on the server. For example, the playback URL may be deleted from the VOD system.

- **Reason 2: Local WebPage Debugging**

TcPlayer Lite does not support debugging in local web page (opening the page where video is played using `file://` protocol). This is because of cross-domain security restrictions of browsers. Simply placing a file such as `test.html` in Windows and testing video playback using the file will definitely end up in failure. You need to upload the file to a server to perform the test. Frontend engineers can achieve local testing by setting up a local proxy for the online web page through reverse proxy. This is a feasible method for local debugging.

- **Reason 3: Mobile Phone Compatibility Problem**

Common mobile browsers do not support FLV and RTMP URLs (The latest QQ Browser can play videos based on FLV protocol, but it is not widely used). You can only use HLS (m3u8) URLs.

- **Reason 4: Cross-Domain Security Problem**

PC browsers achieve video playback based on Flash widgets. Those who did Flash development before would know that **the Flash widget performs cross-domain access check**. You will encounter this problem if cross-domain policy is not deployed on the server where the video you want to play resides in. The workaround can be: Find the cross-domain configuration file `crossdomain.xml` under the root domain name of the video server and add domain name `qq.com` to it:

```
<cross-domain-policy>
<allow-access-from domain="*.qq.com" secure="false"/>
</cross-domain-policy>
```

### Step 4: Configure a Cover Image for the Player

Next, we will explain how to use "coverpic" shown in the above sample code.

**Note: The cover image may become invalid in certain mobile device playback environments because these environments are relatively more complicated compared to PC, and different browsers and App Webviews achieve H5 videos by different means. If you encounter such a situation, feel free to send feedback to our technical support**

(the feedback should include key information such as system and the version of your browser or App) and we will provide assistance as much as possible.

#### 4.1 Easily Configure a Cover Image

You can pass the URL of an image to `coverpic` as the player's cover image. The image will be displayed in the center of the player area, with its actual resolution.

```
"coverpic" : "http://www.test.com/myimage.jpg"
```

#### 4.2 Configure Cover Image Display Style

You can also pass an object to `coverpic` and configure the display style (`"style"`) and image URL (`"src"`) for the cover image in the object.

Three styles are supported:

- default: display the image in the center, with its actual resolution;
- stretch: stretch the image to fill up to entire player area. This may distort the image.
- cover: horizontally stretch the image while keeping its ratio to fill up the player area. The image may not be fully displayed in the area.

```
"coverpic" : {"style":"stretch", "src":"http://www.test.com/myimage.jpg"}
```

#### 4.3 Implementation Case

This is an online sample code segment, which uses `"cover"` to display the cover image. Right-click in a PC browser and select **View Page Source** to view the codes of the page:

```
http://imgcache.qq.com/open/qcloud/video/vcplayer/demo/tcplayer-cover.html
```

**Note:**

The configured cover image may become invalid in certain mobile devices. For more information, please see the FAQ section.

### Step 5: Multi-Definition Support

#### 5.1 Principle

You can choose different definitions for videos on Youku.com, Tudou.com or Tencent. How do we achieve this feature?



First, you should know that **the players cannot change a video's definition**. There is only one definition when the video source is generated, which is called **Original**. The original video has different encoding formats and encapsulation formats, and Web pages do not support all video playback formats. In VOD scenarios, the encoding format for videos must be H.264, and encapsulation format must be MP4 or FLV.

So, how is multi-definition selection implemented? This is where the Video Cloud comes into play:

- In LVB scenarios, the original video from VJ will be transcoded in real time in Tencent Cloud, then videos transcoded with different channels are distributed, such as "High Definition (HD)" and "Standard Definition (SD)". The video from each channel has its own corresponding URL:

```
http://2157.liveplay.myqcloud.com/2157_358535a.m3u8 //Original  
http://2157.liveplay.myqcloud.com/2157_358535a_900.m3u8 //HD  
http://2157.liveplay.myqcloud.com/2157_358535a_550.m3u8 //SD
```

- In VOD scenarios, once a video file is uploaded to Tencent Cloud, you can perform operations to transcode the video and generate videos of different definitions, such as "High Definition (HD)" and "Standard Definition (SD)". Note: The uploaded original video is not transcoded by Tencent Cloud and cannot be played directly.

```
http://200002949.vod.myqcloud.com/200002949_b6ffc.f240.m3u8 //Original, replaced with transcoded video of Ultra High Definition  
http://200002949.vod.myqcloud.com/200002949_b6ffc.f230.av.m3u8 //HD  
http://200002949.vod.myqcloud.com/200002949_b6ffc.f220.av.m3u8 //SD
```

**Note:**

The uploaded original video is not transcoded by Tencent Cloud and cannot be played directly.

## 5.2 Code Implementation

The following code segment allows player to support multiple definitions. In other words, it displays options used to select different definitions on the player's user interface.

```
var player = new TcPlayer('id_test_video', {  
  "m3u8" : "http://200002949.vod.myqcloud.com/200002949_b6ffc.f240.m3u8", //Replace the URL with a valid playback  
  URL.  
  "m3u8_hd" : "http://200002949.vod.myqcloud.com/200002949_b6ffc.f230.av.m3u8",  
  "m3u8_sd" : "http://200002949.vod.myqcloud.com/200002949_b6ffc.f220.av.m3u8",  
  "autoplay" : true, //Most mobile browsers including Safari on iOS do not support auto video playback.  
  "coverpic" : "http://www.test.com/myimage.jpg",  
});
```

## 5.3 Implementation Case

This is an online sample code segment, which uses multi-definition selection and switching features. Right-click in a PC browser and select **View Page Source** to view the code of the page:

```
http://imgcache.qq.com/open/qcloud/video/vcplayer/demo/tcplayer-clarity.html?autoplay=true
```

Generally, you will see the following screen:



PC now supports video playback in multiple definitions as well as definition switching. This is not supported on mobile devices.

## Step 6: Customize Error Messages

The default error messages may not suite your needs. For example, the error messages "network error, check your network configuration or if the playback link is correct" or "video decoding error" can be more appropriate as needed. In this case, you can customize messages in TcPlayer Lite:

## 6.1 Code Implementation

The following code is the core code used to allow the player to support custom messages. The configured messages are mainly placed in the "wording" attribute.

```
var player = new TcPlayer('id_test_video', {
  "m3u8" : "http://200002949.vod.myqcloud.com/200002949_b6ffc.f0.m3u8" ,//Replace the URL with a valid playback URL.
  "autoplay" : true, //Safari browser on iOS do not support this feature.
  "coverpic" : "http://www.test.com/myimage.jpg",
  "wording": {
    2032: "Video request failed, check your network.",
    2048: "m3u8 file request failed, this may be caused by network error or cross-domain issue."
  }
});
```

## 6.2 Implementation Case

This is an online sample code segment, which shows playback failure and uses custom message feature. Right-click in a PC browser and select **View Page Source** to view the code of the page:

```
http://imgcache.qq.com/open/qcloud/video/vcplayer/demo/tcplayer.html?m3u8=http://2527.vod.myqcloud.com/2527_b393eb1.f230.av.m3u8
```

## 6.3 Error Code Reference Table

Code	Message	Description
1	Network error, check if your network configuration or the playback link is correct	(Error prompted by H5)
2	Network error, check if your network configuration or the playback link is correct	Web player cannot decode the video format (error prompted by H5)
3	Video decoding error	(Error prompted by H5)
4	Current system environment does not support this video format	(Error prompted by H5)
5	Current system environment does not support this video format	The player detects that the current browser environment does not support the passed video. The reason may be that the current browser does not support MSE, or Flash plug-in is not enabled
10	Do not use the player under "file" protocol, or video playback may fail	
11	Incorrect parameter used. Check the player calling code	

Code	Message	Description
12	Enter video playback address	
13	The LVB ended, come back later	The (NetConnection.Connect.Closed) event is triggered during a normal RTMP playback (error prompted by Flash)
1001	Network error, check if your network configuration or the playback link is correct	Network disconnected (NetConnection.Connect.Closed) (error prompted by Flash)
1002	Failed to acquire video, check if the playback link is valid	Failed to pull playback file (NetStream.Play.StreamNotFound), this may be caused by a server error or that the video file does not exist (error prompted by Flash)
2032	Failed to acquire video, check if the playback link is valid	(error prompted by Flash)
2048	Failed to load video file, cross-domain access is rejected	Failed to request m3u8 file, this may be caused by network error or cross-domain issue (error prompted by Flash)

**Note:**

Code 1-4 are original H5 events;

Due to the black box feature of Flash and the uncertainty of H5 video playback standards, the error messages will be irregularly updated.

## Source Code Reference

This is an online sample code segment. Right-click in a PC browser and select **View Page Source** to view the codes of the page:

```
http://imgcache.qq.com/open/qcloud/video/vcplayer/demo/tcplayer.html?autoplay=true
```

You can also use it to test your player. Append the URL of the video to be played after the link and refresh to play the video:

```
http://imgcache.qq.com/open/qcloud/video/vcplayer/demo/tcplayer.html?autoplay=true&m3u8=http://1251132611.vod2.myqcloud.com/4126dd3evodtransgzp1251132611/8a592f8b9031868222950257296/f0.f240.m3u8
```

## Parameter List

Here are all parameters supported by the player as well as their detailed description.

Parameter	Type	Default Value	Description
m3u8	String	None	m3u8 playback URL (original) Example: <code>http://2157.liveplay.myqcloud.com/2157_358535a.m3u8</code>
m3u8_hd	String	None	m3u8 playback URL (high definition) Example: <code>http://2157.liveplay.myqcloud.com/2157_358535ahd.m3u8</code>
m3u8_sd	String	None	m3u8 playback URL (standard definition) Example: <code>http://2157.liveplay.myqcloud.com/2157_358535asd.m3u8</code>
flv	String	None	flv playback URL (original) Example: <code>http://2157.liveplay.myqcloud.com/2157_358535a.flv</code>
flv_hd	String	None	flv playback URL (high definition) Example: <code>http://2157.liveplay.myqcloud.com/2157_358535ahd.flv</code>
flv_sd	String	None	flv playback URL (standard definition) Example: <code>http://2157.liveplay.myqcloud.com/2157_358535asd.flv</code>
mp4	String	None	mp4 playback URL (original) Example: <code>http://200002949.vod.myqcloud.com/200002949_b6ffc.f0.mp4</code>
mp4_hd	String	None	mp4 playback URL (high definition) Example: <code>http://200002949.vod.myqcloud.com/200002949_b6ffc.f40.mp4</code>
mp4_sd	String	None	mp4 playback URL (standard definition) Example: <code>http://200002949.vod.myqcloud.com/200002949_b6ffc.f20.mp4</code>
rtmp	String	None	rtmp playback URL (original) Example: <code>rtmp://2157.liveplay.myqcloud.com/live/2157_280d88</code>
rtmp_hd	String	None	rtmp playback URL (high definition) Example: <code>rtmp://2157.liveplay.myqcloud.com/live/2157_280d88hd</code>
rtmp_sd	String	None	rtmp playback URL (standard definition) Example: <code>rtmp://2157.liveplay.myqcloud.com/live/2157_280d88sd</code>
width	Number	None	<b>Required parameter</b> , used to configure player width (in pixels) Example: 640
height	Number	None	<b>Required parameter</b> , used to configure player height (in pixels) Example: 480

Parameter	Type	Default Value	Description
volume	Number	0.5	Used to configure initial volume. Range: 0-1 [v2.2.0+] Example: 0.6
live	Boolean	false	<b>Required parameter</b> , used to configure whether the video is LVB video. This determines whether to render certain controls such as the time axis, and is used to differentiate between VOD and LVB processing logics Example: true
autoplay	Boolean	false	Whether to enable auto-playback <b>Note: This option is only effective for most PC platforms</b> Example: true
coverpic	String/Object	None	Preview cover image. You can either pass an image address, or an object containing image address (src) and display style (style). Available styles: "default": image is displayed 1:1 in the center "stretch": stretch the image to fill up the player area. This may distort the image "cover": horizontally stretch the image while keeping its ratio to fill up the player area. The image may not be fully displayed in the area Example: " http://www.test.com/myimage.jpg " Or { "style": "cover", "src": h ttp://www.test.com/myimage.jpg }
controls	String	"default"	"default": displays default controls. "none": does not display controls. "system": displays system controls on mobile devices. <b>Note: You need to configure this as "system" if you want to use system full screen mode on mobile devices. The default full screen solution uses Fullscreen API + pseudo-full screen</b> <b>Example</b> Example: "system"
systemFullscreen	Boolean	false	In a browser environment that does not support the Fullscreen API, you can try to use the webkitEnterFullScreen method provided by the browser for full screen after you enable it. If Fullscreen API is supported, the system will enter full screen and system control is used Example: true
flash	Boolean	true	Whether to prioritize flash to play videos. <b>Note: this option is only effective on PC platforms [v2.2.0+]</b> Example: true
flashUrl	String	None	You can set flash swf url <b>Note: this option is only effective on PC platforms [v2.2.1+]</b>

Parameter	Type	Default Value	Description
h5_flv	Boolean	false	Whether to enable flv.js to play flv videos. If this is enabled, players will use flv.js to play FLV videos in browsers that support MSE. However, since not all such browsers can use flv.js, players do not enable this attribute by default. [v2.2.0+] Example: true
x5_player	Boolean	false	Whether to enable TBS to play flv videos. If this is enabled, players will directly transmit the flv playback addresses to <code>&lt;video&gt;</code> to play in TBS mode (for example, WeChat on Android or QQ Browser), <a href="#">TBS Video Playback</a> [v2.2.0+] Example: true
x5_type	String	None	Enable the H5 player at the same layer through the video attribute "x5-video-player-type". Available value: H5 (This attribute is an experimental one of TBS kernel and is not supported by non-TBS kernel. <a href="#">Standard for Docking TBS H5 Single-Layer Player</a> ) Example: "h5"
x5_fullscreen	String	None	Declare whether to enter TBS full screen mode when playing a video, by using the "x5-video-player-fullscreen" attribute of "video". Available value: true (this is an experimental attribute of TBS kernel, and is not supported for non-TBS kernels). Example: "true"
x5_orientation	Number	None	Declare the orientation supported by TBS player, using the "x5-video-orientation" attribute of "video". Available values: 0 (landscape), 1: (portrait), 2: (landscape   portrait rotates with mobile phone movement). (This is an experimental attribute of TBS kernel, and is not supported by non-TBS kernels) [v2.2.0+] Example: 0
wording	Object	None	Custom notification Example: { 2032: 'Video request failed, check your network'}
clarity	String	'od'	Default playback definition [v2.2.1+] Example: clarity: 'od'
clarityLabel	Object	{od: 'Ultra High Definition', hd: 'High Definition', sd: 'Standard Definition'}	Custom Definition [v2.2.1+] Example: clarityLabel: {od: 'Blu-ray', hd: 'High Definition', sd: 'Standard Definition'}

Parameter	Type	Default Value	Description
listener	Function	None	Event monitor callback function. A JSON object is passed to this function Example: function(msg){ //Event processing }

## List of Instance Methods

Here's a list of methods supported by player instances:

Method	Parameter	Returned Value	Description	Example
play()	None	None	Start video playback	player.play()
pause()	None	None	Pause video playback	player.pause()
togglePlay()	None	None	Toggle video playback status	player.togglePlay()
mute(muted)	{Boolean} [optional]	true,false {Boolean}	Toggle mute status. The current mute status is returned if no parameter is passed	player.mute(true)
volume(val)	{int} Range: 0-1 [optional]	Range: 0-1	Configure volume. The current volume is returned if no parameter is passed	player.volume(0.3)
playing()	None	true,false {Boolean}	Return whether the video is being played	player.playing()
duration()	None	{int}	Acquire video duration <b>Note: this is only applicable for VOD</b>	player.duration()
currentTime(time)	{int} [optional]	{int}	Configure video playback time point. The current playback time point is returned if no parameter is passed <b>Note: this is only applicable for VOD</b>	player.currentTime()
fullscreen(enter)	{Boolean} [optional]	true,false {Boolean}	Call the full screen API (Fullscreen API). Pseudo-full screen mode is not available when the full screen API is used. The current full screen status is returned if no parameter is passed. <b>Note: mobile devices do not provide APIs for full screen mode, and you cannot acquire their full screen status</b>	player.fullscreen(true)

Method	Parameter	Returned Value	Description	Example
buffered()	None	0-1	Acquire buffer data percentage for video <b>Note: this is only applicable for VOD</b>	player.buffered()
destroy()	None	None	Destroy player instance [v2.2.1+]	player.destroy()
switchClarity()	{String} [Required]	None	Switch clarity, pass values "od", "hd", "sd" [v2.2.1+]	player.switchClarity('od')

**Note:**

The above method is only applicable to the instantiated object of TcPlayer and can only be called after initialization (i.e., after the load event is triggered).

## Advanced Guide

Here, we'll introduce some advanced methods for using video player SDK.

### Use an Advertising SDK

TcPlayer provides a version with integrated IMA SDK. If you need to use the advertisement feature, introduce the following code into the page.

```
<!-- Google IMA SDK -->
<script type="text/javascript" src="//imasdk.googleapis.com/js/sdkloader/ima3.js"></script>
<!-- Use the version with integrated IMA SDK -->
<script type="text/javascript" src="//restcplayer.qcloud.com/sdk/tcplayer-web-1.0.1.js"></script>
```

You can use the advertisement feature by using adTagUrl and auth parameters. Visit <https://tcplayer.qcloud.com> to apply for an account and License information, or send an email to [tcplayer@tencent.com](mailto:tcplayer@tencent.com) for consultation.

```
var player = new TcPlayer('id_test_video', {
  /* Advertisement-related parameter */
  "adTagUrl": "http://ad_tag_url", //Tags of VAST, VMAP and VAPID video ads
  "auth": {
    "user_id": "your_user_id", //Ad account ID
    "app_id": "your_app_id", //Application ID
    "license": "your_license" //Application license
  }
});
```

**Note:**

For TcPlayer 2.2.0 and later versions, this document is not applicable to the version with integrated IMA SDK.

tcplayer-web-1.0.1 is an independent branch.

## ES Module

TcPlayer provides ES Module version, and the module name is TCPlayer. Download URL:

<http://imgcache.qq.com/open/qcloud/video/vcplayer/TcPlayer-module-2.2.1.js>

## Prioritize H5 for Video Playback

TcPlayer combines H5 `<video>` and Flash to play videos. By default, the player chooses the more suitable one for different playback environments.

Although browser providers have already begun to give up their support for Flash widget, most browsers in China still do not support MSE, users cannot switch to H5 `<video>` playback mode when playing FLV HLS (m3u8) videos, and RTMP videos must be played using Flash mode. For this reason, TcPlayer still enables Flash playback mode first by default, and only uses H5 `<video>` to play videos if it detects that the FLASH widget is unavailable.

Using Flash mode by default is because this mode supports the most video formats, while H5 `<video>` only supports MP4 (h.264) by default (other video formats not provided by Tencent Cloud are not discussed here), and it can support HLS (m3u8) and FLV only under certain conditions.

Starting from version 2.2.0, TcPlayer allows users to configure the attribute for playback mode priority. If you prefer H5 `<video>` for playback, you can configure the "Flash" attribute as "false", so the player will enable H5 `<video>` to play videos by default, and use Flash mode if H5 `<video>` is unavailable. If Flash widget is not detected, you will see a message: "Current system environment does not support this video format".

## Listen to Event

TcPlayer combines H5 `<video>` and FLASH to play videos. But the events triggered when playing videos using these two methods are different. In this case, we convert FLASH playback events based on the standards of H5 `<video>` to achieve unified playback event names. TcPlayer captures and transparently transmits the native events triggered by these two playback methods.

[H5 Event Reference List](#)

[Flash Event Reference List](#)

Unified event list:

```
error
timeupdate
load
loadedmetadata
loadeddata
progress
fullscreen
play
playing
pause
ended
```

seeking  
seeked  
**resize**  
volumechange

**Note:**

The fullscreen event cannot be listened to if the full screen mode is enabled by using the control bar.

Events specific to the Flash mode: `netStatus`

**Note:** Due to the black box feature of Flash and that H5 video playback standards are realized in different ways on different platforms, events are triggered differently and can yield different results. Be careful with these differences during the development process.

Different events triggered on mobile device and PC Flash when video is loaded to "ready to play" status while auto-playback is disabled.

**Mobile Device:**

```
Object {type: "load", src: H5Video, ts: 0, detail: Object}
Object {type: "resize", src: H5Video, ts: 1150.5800000000002}
Object {type: "loadedmetadata", src: H5Video, ts: 1150.5850000000003}
Object {type: "volumechange", src: H5Video, ts: 1156.19}
Object {type: "seeking", src: H5Video, ts: 1168.665}
Object {type: "timeupdate", src: H5Video, ts: 1256.8400000000001}
Object {type: "seeked", src: H5Video, ts: 1256.85}
Object {type: "loadeddata", src: H5Video, ts: 1256.865}
Object {type: "timeupdate", src: H5Video, ts: 1256.9}
Object {type: "progress", src: H5Video, ts: 1408.7800000000002}
```

**PC Flash:**

```
Object {type: "load", src: FlashVideo, ts: 27, detail: Object}
Object {type: "loadedmetadata", src: FlashVideo, ts: 166, detail: Object}
Object {type: "volumechange", src: FlashVideo, ts: 184}
Object {type: "progress", src: FlashVideo, ts: 1741}
```

**Note:** The differences mentioned above exist between two platforms. There are also differences between different mobile devices and applications.

Application case: By using event listening, you can implement auto reconnection when the playback fails. [Online Example](#).

## Update Log

The following shows the major version of TcPlayer after continuous update and improvement, to make it easier for users to keep track of the conditions in these versions. Some of the historical bug fixes and minor versions are not listed.

Date	Version	Update
2016.12.28	2.0.0	Initial version
2017.3.4	2.1.0	Till 2017.6.30, TcPlayer has gone through a number of iterative development processes and has gradually become stable. Feature descriptions in current documents are all based on this version, unless specified otherwise.
2017.6.30	2.2.0	<ol style="list-style-type: none"> <li>1. Added parameters for controlling playback environment determination: Flash, h5_flv, x5_player;</li> <li>2. Adjusted the initialization logic of the player and optimized the error prompt effect;</li> <li>3. Added support for flv.js, which can be used to play flv videos if conditions are met;</li> <li>4. Added support for the "x5-video-orientation" attribute;</li> <li>5. Added playback environment determination logic. You can adjust the priority of H5 and Flash with parameter, and decide whether to enable TBS playback;</li> <li>6. Enabled version number release method to avoid affecting the usage of previous versions;</li> <li>7. Optimized timestamps for triggering events (unified as standard time);</li> <li>8. Fixed Bugs.</li> </ol>
2017.12.7	2.2.1	<ol style="list-style-type: none"> <li>1. Added the systemFullscreen parameter;</li> <li>2. Added the flashUrl parameter;</li> <li>3. Fixed the UI problem when users switch to mute after the volume is set to max;</li> <li>4. Fixed the problem that two clicks are needed to play the video on WeChat for iOS 11;</li> <li>5. Fixed the problem that the Safari 11 system style is covered;</li> <li>6. Adapted to the condition that X5 kernel will trigger "seeking", but not "seeked";</li> <li>7. Fixed the problem of currentTime configuration failure when the progress bar is dragged to the starting position;</li> <li>8. Added the capability of switching the clarity while keeping the volume unchanged;</li> <li>9. Fixed the problem that the player width cannot be determined when the page width is 0;</li> <li>10. Added the capability of completely terminating player node for "destroy" method.</li> </ol>
2017.12.20	2.2.1	<ol style="list-style-type: none"> <li>1. Added the capability of setting the message clarity;</li> <li>2. Added the capability of setting the default clarity;</li> <li>3. Supported clarity switch methods.</li> </ol>

# FAQ

Last updated : 2018-09-20 11:27:36

## Which browsers are supported by TcPlayer?

TcPlayer has been tested on and supports the following browsers: PC: IE 10+, Edge, Chrome, Firefox, QQ Browser, MAC Safari. Mobile: Android 4.4+, iOS 8.0+, WeChat, Mobile QQ, QQ Browser, Chrome, Safari.

To support IE 8 and IE 9, you must introduce the Polyfill script before introducing the player script, as shown below:

```
<!--[if lt IE 9]>
<script src="//imgcache.qq.com/open/qcloud/video/vcplayer/libs/es5-shim.js" charset="utf-8"></script>
<script src="//imgcache.qq.com/open/qcloud/video/vcplayer/libs/es5-sham.js" charset="utf-8"></script>
<![endif]-->
<script src="//imgcache.qq.com/open/qcloud/video/vcplayer/TcPlayer-2.2.1.js" charset="utf-8"></script>;
```

## On mobile devices, the video does not go into full screen when TcPlayer is switched to full screen mode, and browser interface is still displayed. Why is that?

First, you should know that the full screen solution provided by TcPlayer is to use Fullscreen API + pseudo-full screen. If the browser supports Fullscreen API, the video container will fill up the entire screen when the player goes into full screen mode, and the control bar is provided by TcPlayer, as shown in the figure:



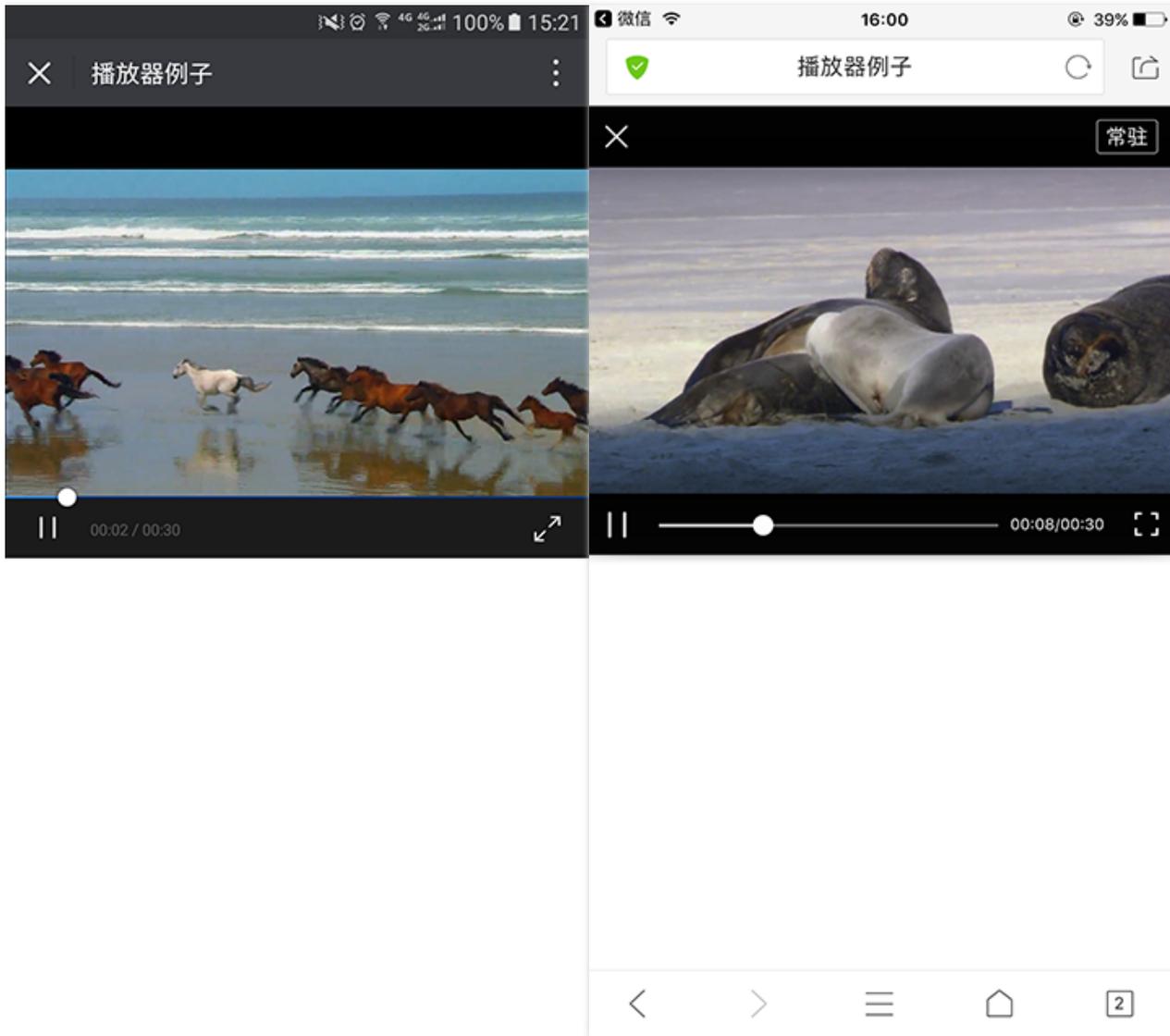
(Android Chrome)

If the browser does not support Fullscreen API, then pseudo-full screen mode will be used, as shown in the figure:



(Left: WeChat on Android. Right: WeChat on iOS)

The control bars displayed in these two full screen modes are all provided by TcPlayer, and you enter these modes by tapping on the full screen button on the control bar or by using the method provided by TcPlayer. However, the control bar provided by TcPlayer may not always be displayed on mobile devices, as their webview hijacks the video playback in most situations and uses the control bar provided by webview. In this case, the TcPlayer control bar will not be displayed, and you cannot use the full screen solution provided by TcPlayer, as shown in the figure:



(Left: WeChat on Android, where video playback is hijacked by TBS. Right: iOS, where video playback is hijacked by QQ Browser)

Upon entering full screen mode:



(Left: WeChat on Android. Right: QQ Browser on iOS)

As you can see, the control bar is different from the TcPlayer one. No APIs are provided for JS in this full screen mode, so TcPlayer cannot realize this mode. We usually refer to the full screen mode where the video covers the whole screen as System Full Screen, and the mode where the video covers the whole browser page area as Pseudo-Full Screen. Therefore, if you can see the browser interface after going into full screen mode, then it is pseudo-full screen. You can only use the control bar provided by the system if you want to go into system full screen mode on mobile devices. You may choose the control bar type by using the "controls" attribute of TcPlayer.

### Why is the screen stretched when the video is played in H5?

Video stretching is not supported when playing video in H5. Check if the player container width/height configuration is correct.

### Why can't I cover my own div on top of the video?

Different browsers implement the `<video>` tag in different ways. For example, if you open a web page in QQ or WeChat (on Android system), then it's very likely that X5 browser kernel is used (the kernel which is bound with QQ or WeChat, i.e. the QQ Browser kernel). For certain reasons, the QQ Browser team implemented such a solution that "the video `<video>`

must be placed in the top layer" (for more information, please see [QQ Browser Documentation](#)). However, with recent coordination among teams within the company, the QQ Browser Team is gradually changing this policy. This issue may have already been solved in the latest X5 browser kernel as you read this document.

### **Why is the configured cover image not working?**

This is the same problem with the previous one ("cannot cover div on top of the video"). The cover image cannot be displayed as long as the browser does not allow elements to cover up the `<video>` tag.

### **Why is video displayed in full screen mode by default in certain mobile browsers?**

If the video is played through inline playback inside the application (that is, your own application encapsulates an iOS webview control which is used to display web pages. In this mode, you can customize the details of webview, which may have different performance from standard Safari browser), you can configure the "webkit-playsinline" attribute for the `<video>` tag in HTML (or configure the "playsinline" attribute if you're using iOS 10), and then configure "allowsInlineMediaPlayback" for webview. In this way, videos are played in non-full screen mode (inline playback) when you open pages in the application.

If you open the page in Safari on iOS 10, you can realize non-full screen playback (inline playback) by using the method above (configure "playsinline" attribute for the `<video>` tag). You cannot disable auto full screen playback for Safari on systems below iOS 10. This attribute is already added for our browser. It only needs to be supported by the devices.

For Android devices, it is known that there are many different customized versions for Android systems, and each system realizes the `<video>` tag in a different way, without a universal standard. For this reason, video playback capabilities on Android have much lower consistency compared to iOS. The "webkit-playsinline playsinline" attribute is already added for the player according to current universal method. It only needs to be supported by the devices.

### **Why can't I realize auto video playback in mobile browsers?**

There are only two ways to realize auto video playback on mobile web: by configuring the "autoplay" attribute for the `<video>` tag, or by calling the "play()" method provided by the `<video>` tag. However, auto video playback has always been prohibited on mobile web, thus the universal method now is to trigger video playback by users actions. For example, monitor on tap events of users and call the "play()" method. It is possible that certain customized webviews support the "autoplay" attribute of the `<video>` tag, or can realize auto video playback by calling other special functions. Videos can be played automatically when you open the pages in these webviews. Our player will add the "autoplay" attribute for the `<video>` tag if "autoplay" is configured as "true". Now we only need the devices to support this attribute.

### **Why does the Flash player have two play buttons in Chrome on PC?**

Flash is no longer be automatically played starting from Chrome 42 (Google has purchased WebRTC and made it open-source for a reason). Chrome only plays major Flash contents automatically, while other Flash contents are paused, unless users enable them manually.

### **Why can the LVB video be played in browsers on PC but cannot on mobile devices?**

Only hls (m3u8) protocol is supported for playing LVB videos in mobile browsers. Thus, you need to confirm whether the LVB stream pulling addresses contain URL for pulling hls (m3u8) stream. The video cannot be played on mobile phones if you only provide an flv or rtmp address for our player.

# Web LVB Player 1.0

## Web LVB Player 1.0

Last updated : 2018-09-20 17:53:42

### Feature Description

Tencent Cloud LVB player Web SDK solution allows users to directly use the proven video playback capability. Through flexible APIs, the SDK can be easily integrated with user's Web Apps to achieve the playing with desktop Apps. At the same time, the Web SDK provides the capability of uploading videos at Web end.

The file formats supported by the SDK are limited by the global hotlink protection feature. For more information, please see FAQs on the official website. This document is intended for developers who want to use Tencent Cloud LVB player Web SDK for development and have basics knowledge in JavaScript.

### Supported Formats

Playback Format: LVB video formats supported by the Web SDK.

Playback Format	PC Browser	Mobile Browser
HLS (m3u8)	Yes	Yes
RTMP	Yes	No
FLV	Yes	No

**Android system compatibility:** Unlike iPhones that only use a Safari browser, a wide range of native browsers may come with Android phones. For this reason, compatibility of Android browsers has become a problem that plagues the industry. Therefore, the above table does not apply to all Android phones.

### Preparations

#### Step 1: Activate the service

Sign up for a Tencent Cloud account at [Tencent Cloud official website](#), and activate the **LVB** service.

#### Step 2: Create a channel

Go to the [LVB console](#) and create an LVB channel.

#### Step 3: Obtain an ID

You can find and manage the LVB channel in the console. You can find its app\_id on the console interface. Click the channel to find its ID in **Basic Settings**.

Console Home Products lglelin@ten... Billing Center Ticket

**Live**

- Overview
- Access Management
  - LVB Code Access
  - Channel Hosting**
- Monitoring
- Screenshot & Porn Detection
- Global Settings
- Statistical Analysis

**LVB Management** appid: [redacted]

New Open Disabled Delete Search by channel ID

Each user has five streaming channels for free. When you are using more than five streaming concurrent channels (excluding non-streamed channels), the exceeding channels will be a difference between the peak number of actual monthly push channels and the five free channels. For billing details and examples, please see [price overview](#).

Filter by:  All  Created manually  Created by API

<input type="checkbox"/>	LVB Name	ID	Status	Watermark	Create time ↑	Operation
<input type="checkbox"/>	11	10905947996257028763	No input stream	N/A	2018-07-18 21:43:32	<a href="#">Close</a> <a href="#">Test</a> <a href="#">Share</a>

Console Home Products

**Live**

- Overview
- Access Management
- Monitoring
- Screenshot & Porn Detection
- Global Settings
- Statistical Analysis

[Back](#) | 11

**Basic settings**

**Basic settings** [Edit](#)

Name	11
Channel ID	[redacted]
Status	No input stream
Description	11
Encoding	Original video
Watermark settings	N/A

#### Step 4: Prepare pages

Introduce the initialization script to the page in which you want to play videos (including PC or H5).

```
<script src="//qzonestyle.gtimg.cn/open/qcloud/video/live/h5/live_connect.js" charset="utf-8"></script>;
```

**Note:**

The playback page should be accessed with IP address or domain name. Playback is not allowed if you directly

open the static page;

Note: The channel ID and APPID can also be obtained through the server API. [Link to the reference document.](#)

## Basic Usage of APIs

### Step 1: Add a player container

Place a player container in the page where you want to display the player. For example, add the following code into index.html:

```
<div id="id_video_container" style="width:100%; height:auto;"></div>
```

The container's ID, width, and height can be customized.

### Step 2: Create a Player object

Create a player object in the JavaScript that is introduced at the bottom of the page. In this case, the player constructor is used:

```
var player = new qcVideo.Player("id_video_container", {  
  "channel_id": "16093104850682282611",  
  "app_id": "1251783441",  
  "width" : 480,  
  "height" : 320  
});
```

Call the constructor to generate a player object and find the LVB stream to play based on the channel\_id and app\_id. If channel\_id and app\_id do not exist, the video can be played with the LVB address. An example is provided in the [Case 3 in API Use Cases](#). You can control the player with the Player object. For more information, please see the Overview of API Methods below the parameter options of player object.

### Complete sample code

```
<!DOCTYPE html>  
<html>  
<head>  
<meta http-equiv="X-UA-Compatible" content="IE=Edge,chrome=1" />  
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>  
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=0, minimum-scale=1.0, maximum-scale=1.0"/>  
<title>LVB</title>  
</head>  
<body>  
<div id="id_video_container" style="width:100%; height:auto;"></div>  
<script src="//qzonestyle.gtimg.cn/open/qcloud/video/live/h5/live_connect.js" charset="utf-8"></script>  
<script type="text/javascript">  
(function () {  
  var player = new qcVideo.Player("id_video_container", {
```

```
"channel_id": "16093104850682282611",
"app_id": "1251783441",
"width" : 480,
"height" : 320
});
})()
</script>
</body>
</html>
```

## API Use Cases

Complete [Add a player container](#) as described in Basic Usage of APIs before using API.

### Case 1: Play LVB videos on PC or mobile device (H5)

LVB SDK works in the same way both on PC and H5 devices. The SDK detects the platform and selects the best playback scheme. For example, on PC platform, SDK prefers a Flash player to adapt to different video formats (A Flash player version above 10 is required. Otherwise you are prompted to upgrade the Flash player). On a mobile device (H5), SDK uses a video label to play videos (If the browser does not support H5, you are prompted to use QQ browser). The SDK supports playing videos using both channel ID and video file address.

#### Note:

The two playback methods cannot be used at the same time.

### Case 2: Play videos using channel ID

```
var option = {
"channel_id": "16093425727656143421",
"app_id": "1251132611",
"width" : 480,
"height" : 320

//...You may use other custom attributes
};

var player = new qcVideo.Player("id_video_container", option);
```

#### Note:

LVB key mode is not supported when playing videos using channel ID.

### Case 3: Play videos using LVB video address

If neither app\_id nor channel\_id exists, the player can alternatively use LVB video address to play the video.

```
var option = {
  "live_url" : "http://2157.liveplay.myqcloud.com/2157_358535a.m3u8",
  "live_url2" : "http://2000.liveplay.myqcloud.com/live/2000_2a1.flv",
  "width" : 480,
  "height" : 320

  //...You may use other custom attributes
};

var player = new qcVideo.Player("id_video_container", option);
```

**Note:**

A maximum of two playback addresses (live\_url and live\_url2) are supported. If both of them are passed, the address for which the platform can provide the best support is selected for playback. For example, on PC, RTMP or FLV format is preferred, while on a mobile H5, hls format is preferred.

**Case 4: How to use "On-screen Comment"**

After the initialization of player, you can add on-screen comments for videos by calling the addBarrage(barrage) method of player object. For more information on parameters, please see [Overview of API Methods](#). For example, add two on-screen comments for the video that is being played:

```
var barrage = [
  {"type":"content", "content":"hello world", "time":"1"},
  {"type":"content", "content":"Center", "time":"1", "style":"C64B03;30", "position":"center"}
];
player.addBarrage(barrage);
```

**Note:**

On-screen comment is only implemented at the frontend. Backend support should be self-developed. This feature only applies to Flash players on PC, but not supported in H5.

## Overview of API Methods

### Constructor

```
qcVideo.Player(id, option, listener);
```

**ID:** String; **Required;** This parameter indicates the ID of the container where the player is located in the page and can be customized.

**option:** Object; **Required.** This parameter indicates the options that can be set for player's parameters. The options are as follows:

Parameter	Type	Default Value	Description
channel_id	String	None	This is <b>Required</b> when playing video by video ID
app_id	String	None	The parameter is <b>Required</b> if the LVB video is played using video ID. For the videos under the same account, this parameter remains the same.
width	Number	None	<b>Required</b> , used to configure player width (in pixel). Example: 640
height	Number	None	<b>Required</b> , used to configure player height (in pixel). Example: 480
cache_time	Number	0.3	The maximum cache time before the LVB video playback starts. This parameter can help avoid stutter in RTMP video caused by insufficient downstream bandwidth (Optional). <b>Note: This parameter only applies to Flash players on PC.</b>
h5_start_patch	Object	None	Starts to play pre-video ads on H5 (Optional) { url : image address, stretch: false //Indicates whether the image is stretched to fit the full screen of player. Default value is false }

Parameter	Type	Default Value	Description
wording	Object	None	<p>LVB messages can be customized (Optional; For details, refer to <a href="#">Error Code</a>)</p> <pre>{ '1' : 'Database error', '2' : 'Cannot connect to LVB source; the LVB source has not pushed video content (hls)', '3' : 'LVB has finished (hls)', '113' : 'Connection timeout; please try again later', '114' : 'Connection timeout; please try again later', '1000' : 'Incorrect channelID or APPID', '1001' : 'Invalid parameter; failed to obtain the bizid', '1009' : 'LVB source has expired', '10000' : 'Connection timeout; Please check network settings', '10008' : 'Wrong password; please enter again', //Invalid password '10020' : 'Insufficient balance in LVB account; please top up it in time', '11044' : 'Invalid request', '11045' : 'channelID is missing in the request parameter', '11046' : 'Wrong password, please enter again', '20110' : 'Wrong password, please enter again', '20113' : 'LVB has finished; please try again later' , //'downstream type does not exist' Pulling stream does not exist '20201' : 'LVB has finished; please try again later', //get upstream info error' Error while querying pushing stream '20301' : 'LVB channel does not exist; please verify channel ID', 'TipReconnect' : 'Reconnecting' 'TipRequireSafari' : 'Current browser does not support video play; please use safari to watch the video' 'TipRequireFlash' : 'Current browser does not support video play; you can play the video by downloading the latest QQ browser or installing FLASH player' 'TipVideoinfoResolveError' : 'Error while parsing video information; please check whether the parameter is correct, //No json data is returned from API; cannot parse the data 'TipVideoinfoError' : 'Video information error; please check whether the parameter is correct', 'TipConnectError' : 'Failed to connect to the service; please check the network settings', 'TipConnectDeny' : 'Connection to the service is denied', //Flash request triggered a security exception 'TipLiveEnd' : 'LVB has finished; please try again later', // NetStream.Play.Stop event, 'TipStreamNotFound' : 'LVB has finished; please try again later' //Failed to connect to LVB source; the LVB source has not pushed video content }</pre>
live_url	String	None	LVB address. hls/rtmp/flv formats are supported. It is <b>required</b> when video is played using video address
live_url2	String	None	LVB address. hls/rtmp/flv formats are supported (Optional)

Parameter	Type	Default Value	Description
volume	Number	0.5	Set the initial volume value ranging from 0 to 1. Default value is 0.5. (Optional) <b>Note: This parameter only applies to Flash players.</b>
https	Number	0	Enable or disable HTTPS for playback pages. 0: Disable; 1: Enable.
hide_volume_tips	Number	0	Whether the volume tip is hidden. 0: Displayed; 1: Hidden <b>Note: This parameter only applies to Flash players.</b>
x5_type	String	None	Enable the H5 player at the same layer through the video attribute "x5-video-player-type". Value: H5. (This attribute is an experimental one of TBS kernel and is not supported by non-TBS kernel. For more information, please see <a href="#">TBS Document</a> )
x5_fullscreen	String	None	Enter the full screen mode during video play. Supported value: true: Enable (This attribute is an experimental one of TBS kernel and is not supported by non-TBS kernel. For details, refer to <a href="#">TBS Document</a> )
WMode	String	window	When in window mode, you cannot put other page elements over the Flash player. You can change this into opaque or parameter values for other flash wmode if required. Other elements can be placed over the flash player. <b>Note: This parameter only applies to Flash players on PC.</b>

**listener:** Object; Optional; List of callback functions in case of playing status change.

Function Name	Type	Description
playStatus	function	Triggered when playback status changes. Callback function parameter "status": String returned values: ready: "Player is ready", playing: "Playback in progress", playEnd: "Playback ended", error: "Play has finished with an exception or error" //Since the playback event trigger conditions vary with mobile devices, it is recommended to listen the error when listening the playEnd event to prevent the callback from being executed incorrectly. type: String The type of error is returned when an error occurs. Example: function(status, type){ ... }

## Settings and actions

The player objects returned by the constructors can be set using the following methods:

Method	Description
resize(width,height)	Parameter: width: int; height: int Function: Set the width and height for current player. Returned value: none

Method	Description
play()	Function: Start playback. Returned value: int (common error code) <b>Note: This function only applies to Flash players on PC.</b>
stop()	Function: Stop playback. Returned value: int (common error code) <b>Note: This function only applies to Flash players on PC.</b>
pause()	Function: Pause the playback of current video. Returned value: int (common error code) <b>Note: This function only applies to Flash players on PC.</b>
resume()	Function: Resume the playback of current video. Returned value: int (common error code) <b>Note: This function only applies to Flash players on PC.</b>
addBarrage(barrage)	Parameter: barrage // Array barrage information [ "type":"content", // Message type, content: plain text (Required) "content":"hello world", // Text message (Required) "time":"1.101" ,//The time length (in sec) between the moment when the current method is called for adding a caption and the moment when the caption is displayed. (Required) "style": "C64B03;35" ,// Separated by semicolons; the first is color value, and the second is font size (optional) "postion":"center" // Location center: center, bottom: bottom, up: top (optional) }, ... ] Function: Add on-screen comments Returned value: int <a href="#">Error Codes</a> <b>Note: On-screen comment is only implemented at frontend, and backend functions should be self-developed. This function only applies to Flash players on PC.</b>
closeBarrage()	Function: disable on-screen comment. Call addBarrage again to re-enable the feature. Returned value: int <a href="#">Error codes</a> <b>Note: On-screen comment is only implemented at frontend, and backend functions should be self-developed. This function only applies to Flash players on PC.</b>

The common error codes of the above methods are as follows:

Error Code	Description
200	Operation successful
0	Player not fully initialized
-2	Unknown operation command

# Problem solving

Last updated : 2018-07-10 18:12:05

## Error Codes

List of error codes during the use of SDK

- Error messages returned from the frontend

Message	Description
An error occurred while parsing video information. Check whether the parameter is correct.	No JSON data is returned from API. Data cannot be parsed.
Video information error. Check whether the parameter is correct.	An error occurred while parsing video information
Failed to connect the service. Check the network settings.	Failed to obtain the video channel information.
Connection to service is denied.	Flash request triggered a security exception.
Video data is missing.	Video source data is missing.
LVB has finished. Try again later.	NetStream.Play.Stop event
LVB has finished. Try again later.	Failed to connect to the LVB source. The LVB source has not pushed video content

- Error messages returned from the backend

Code	Message	Description
1	Database error	Database connection timed out or an error occurred during query.
2	Failed to connect to LVB source. LVB source has not pushed video content (hls).	M3U8 file cannot be obtained due to LVB source connection failure.
3	LVB has finished (hls).	Manifest is not a valid M3U8 file, or LVB source has finished.
113	Connection timeout. Try again later.	The parameter is incorrect.
1000	Incorrect channelID or APPID	The app_id is incorrect (length error).
1001	Invalid parameter. Failed to obtain the bizid.	The app_id is incorrect (with a correct length but wrong content).
1009	LVB source has expired.	The broadcast address is invalid. LVB source has expired.
10000	Request timeout	Timeout when pulling player configuration and video information. Check your network and try again. Timeout is 10 seconds.

Code	Message	Description
10001	Failed to parse data	Failed to parse the data obtained by pulling player configuration and video information. It may be caused by a network problem or server exception
10002	Connection timeout. Try again later	Failed to pull player configuration and video information. It may be caused by a network problem or server exception
10008	Wrong password. Enter again.	Invalid password
10020	Insufficient balance in LVB account. Top up it in time	The balance is insufficient.
11044	Invalid request	The APPID is missing
11045	Channel ID is missing in the request parameter.	Channel ID is missing
11046	Wrong password. Enter again.	Password is missing
20110	Wrong password. Enter again.	Invalid password
20113	LVB has finished. Try again later.	Pulling stream does not exist (downstream type is not exist).
20201	LVB has finished. Try again later.	An error occurred while querying pushing stream (get upstream info error).
20301	LVB channel does not exist. Check the channel ID.	Incorrect channel_id (app_id is correct).

**Note:**

Note: In case of any error code that is not listed in this table, contact customer service. Our engineers can help you solve the problem.

**FAQs**

- **Why is the screen distortedly stretched when the video is played in H5?**

Video stretching is not supported when playing video in H5. Check if the player container has right width/height configuration.

- **Why did I receive the message that "The LVB has finished. Try again later"?**

If you get no response from the LVB address when attempting to connect to this address and fail to connect to it even after five attempts, this message appears. In this case, you need to verify whether the video stream is being pushed and whether the network connectivity is normal.

- **The video cannot be hidden on the QQ browser.**

QQ browser takes over the video playback feature from H5, and the X5 kernel uses self-developed player to play

videos. QQ browsers use a unified playback interface to ensure a good user experience. For more information, please see [QQ Browser Documentation](#).

- **Video is automatically played in full screen mode on iOS.**

By default, video is played in full screen mode on iOS system due to the webkit setting. To achieve inline playback within an App, you can set the webkit-playsinline attribute. Any Safari browser on iOS below 10 is unable to disable the automatic use of full screen mode.

- **Why does the Flash player have two play buttons in Chrome on PC?**

Flash is no longer automatically played starting from Chrome 42. Chrome only plays major Flash contents automatically, while other Flash contents are paused, unless users enable them manually.

- **Why can I play LVB videos in browsers on PC, but not on mobile devices?**

Only HLS videos are supported in the browsers on mobile devices. Check whether the LVB pulling address contains an HLS pulling URL.

# Web VOD Player 1.0

## Web VOD Player 1.0

Last updated : 2018-09-20 18:00:47

### Feature Description

This document describes how to use the Web Player (Web SDK) of Tencent Cloud VOD service, which allows users to directly use the proven video playback capability. Through flexible APIs, the SDK can be easily integrated with self-built Web Apps to achieve the playing with desktop Apps. At the same time, the Web SDK provides the capability of uploading videos at Web end.

The file formats supported by the SDK are limited by the global hotlink protection feature. For more information, please see FAQs on the official website. This document is intended for developers who want to use Tencent Cloud VOD player Web SDK for development and have basics knowledge in JavaScript.

### Supported Features

#### Playback Format

The following table lists the video formats supported by Web SDK:

Playback Format	PC Browser	Mobile Browser
HLS (m3u8)	Yes	Yes
MP4	Yes	Yes
FLV	No	No

**Android system compatibility:** Unlike iPhones that only use a Safari browser, a wide range of native browsers may come with Android phones. For this reason, compatibility of Android browsers has become a problem that plagues the industry. Therefore, the above table does not apply to all Android phones.

#### Upload format

Video formats supported by the SDK for upload are as follows:

Standard	Detailed Format
Microsoft	WMV, WM, ASF, ASX
REAL	RM, RMVB, RA, RAM
MPEG	MPG, MPEG, MPE, VOB, DAT
Others	MOV, 3GP, MP4, MP4V, M4V, MKV, AVI, FLV, F4V

**Transcode service of the VOD platform:** Since MP4 and HLS (m3u8) formats are relatively widely supported for both PC and mobile browsers, Tencent Cloud VOD platform always publishes the uploaded videos in these formats.

### Platform compatibility

It requires excessive effort to create two sets of codes for smartphone browser and PC browser. However, by using this player, the same code will adapt itself to PC browser/smartphone browser, which means the player will automatically choose the optimal playback method according to the platform it runs on. For example: On PC, the service will use Flash player as first priority in order to cope with various video formats. On smartphone, the service will use HTML5 technology to play videos instead.

## Preparations

### Step 1: Activate the service

Sign up for a Tencent Cloud account at [Tencent Cloud official website](#), and activate the **VOD** service.

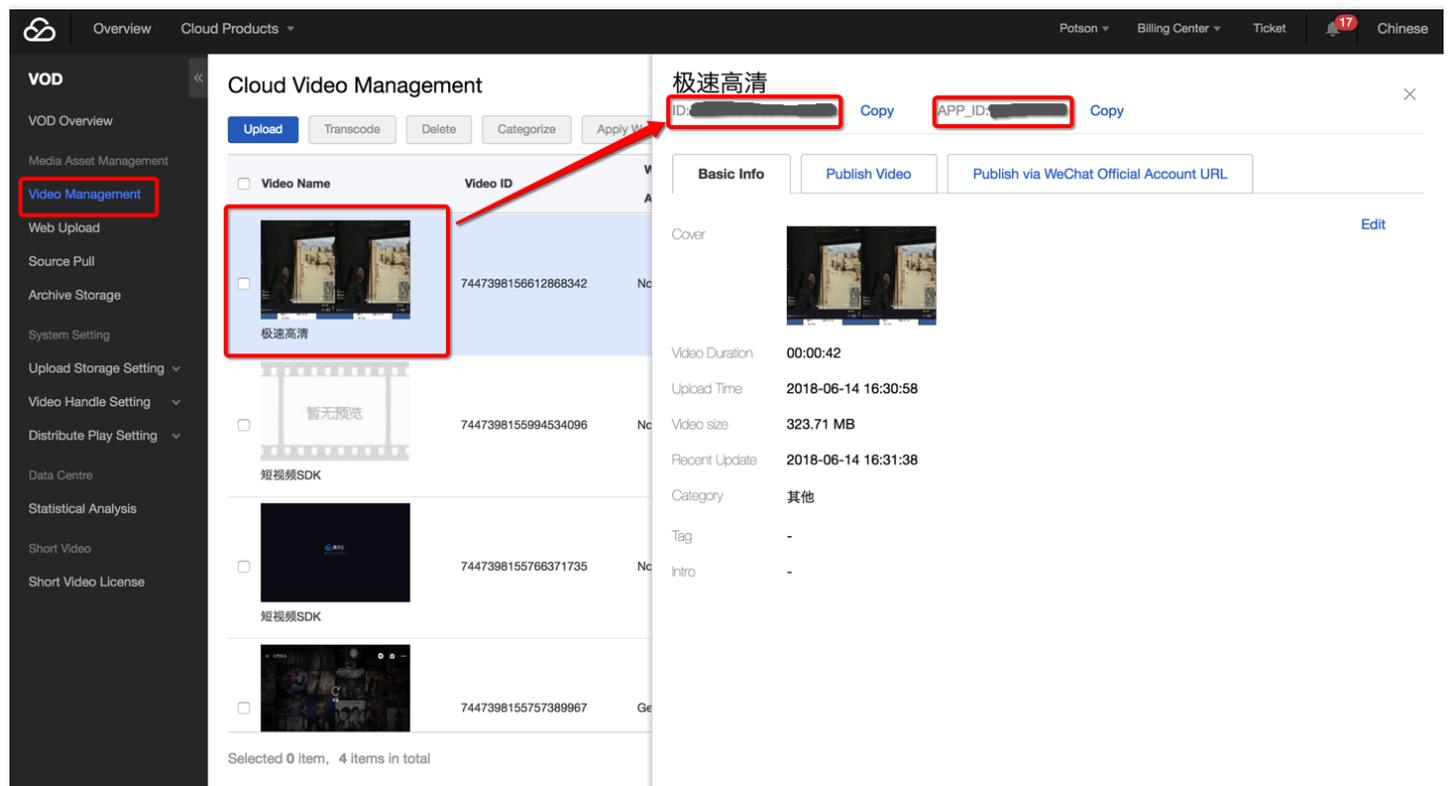
### Step 2: Upload files

After the VOD service is activated, go to [VOD Video Management](#) to upload new video files. Since the document is meant to introduce how to use the player, this operation can help you obtain your own online video address. You are unable to enter this page if you haven't activated VOD service.

### Step 3: Obtain an ID

After the video is uploaded, you can find the file ID (the most basic information for the player to play videos) in the Video Management page. The player also includes Quality Statistics feature. You need to verify your APPID which can be queried in the Video Management page before using the player. In the figure below, the ID on the left is the video file ID,

while the other one is your APPID.



#### Step 4: Prepare pages

Introduce the initialization script to the page in which you want to play videos (including PC or H5):

```
<script src="//qzonestyle.gtimg.cn/open/qcloud/video/h5/h5connect.js" charset="utf-8"></script>;
```

**Local file restriction:** You cannot play local files using the player due to cross-domain issues. You must access the player through an IP or domain--this is why we asked you to upload a video file first and acquire its online playback address.

## Adding Player

You can add a video player into your page by following the two simple steps below.

#### Step 1: Add a player container

Place a player container in the page where you want to display the player. For example, add the following code into index.html. The container ID, width and height can be customized.

```
<div id="id_video_container" style="width:100%; height:auto;"></div>;
```

#### Step 2: Create a Player object

Next, create a player object in the introduced JavaScript using a player constructor.

```
var player = new qcVideo.Player("id_video_container", {
  "file_id": "1465197896261041838",
  "app_id": "125132611",
  "width":640,
  "height":480
});
```

The constructor is used to generate a player object and find the corresponding video to play based on file\_id and app\_id. You can control the player with the player object. For more information, please see [Overview of API Methods](#) for the parameter options of player object.

## Complete sample code

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="X-UA-Compatible" content="IE=Edge,chrome=1" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=0, minimum-scale=1.0, maximum-scale=1.0"/>
<title>VOD</title>
</head>
<body>
<div id="id_video_container" style="width:100%; height:auto;"></div>
<script src="//qzonestyle.gtimg.cn/open/qcloud/video/h5/h5connect.js" charset="utf-8"></script>
<script type="text/javascript">
(function () {
var player = new qcVideo.Player("id_video_container", {
  "file_id": "1465197896261041838",
  "app_id": "125132611",
  "width":640,
  "height":480
});
})();
</script>
</body>
</html>
```

## Other Cases

### case 1: How do I play a video if I have the video address but not the file\_id and app\_id?

Video playback address need to be passed, in which case file\_id and app\_id are not required. JS example:

```
var option = {
  "width": 640,
  "height": 480,
  //...You may use other custom attributes
  "third_video": {
```

```
"urls":{
  20 : "http://208.vod.myqcloud.com/204.mp4"//This is only an example. Please replace this with actual video address
}
};
var player = new qcVideo.Player("id_video_container", option);
```

The "urls" attribute of the "third\_video" parameter is an Object, where you can pass multiple video addresses with different video definitions. Detailed parameter descriptions can be found in [Overview of API Methods](#).

**Note:**

"urls" should include at least one video address

### case 2: How to use "On-screen Comment"?

After the initialization of player, you can add on-screen comments for videos by calling the `addBarrage(barrage)` method of player object. For more information on parameters, please see [Overview of API Methods](#). For example, add two on-screen comments for the video that is being played:

```
var barrage = [
  {"type":"content", "content":"hello world", "time":"1"},
  {"type":"content", "content":"Center", "time":"1", "style":"C64B03;30", "position":"center"}
];
player.addBarrage(barrage);
```

**Note:**

On-screen comment is only implemented at the frontend. Backend support should be self-developed. This feature only applies to Flash players on PC, but not supported in H5.

### Case 3: How do I perform some operations at the end of the video, such as video recommendation?

Use the listener parameter and pass a callback function for the `playStatus` event. This function will be called when the playback status changes. For description on the specific callback function parameters, please see [API Overview](#).

For example:

```
var option ={
  "file_id":"1465197896261041838",
  "app_id":"125132611",
  "width":800,
  "height":720
  //...You may use other custom attributes
};
var listener = {
  playStatus: function (status){
    //TODO
```

```
console.log(status);
}
};
var player = new qcVideo.Player("id_video_container", option, listener);
```

#### Case 4: How to make the player remember video playback progress and start playing the video from the same point the next time?

In "option", set the parameter "remember" to 1. The player records the time point when the video was stopped in the previous playback, and continues from this point upon the next playback. For example:

```
var option = {
  "file_id": "1465197896261041838",
  "app_id": "125132611",
  "width": 800,
  "height": 720,
  "remember": 1
  //...You may use other custom attributes
};
var player = new qcVideo.Player("id_video_container", option);
```

#### Case 5: How to make the player adjust its size automatically along with the web page?

Use the player object method "resize(width, height)" to modify player size dynamically.

```
player.resize(640, 480);
```

#### Case 6: How to play videos that have been configured with passwords from Cloud Video Management?

Just like playing normal videos, SDK automatically displays a password input window. Playback starts after the password is entered.

##### Note:

Password feature is only available when playing videos by passing video IDs.

#### Case 7: How to generate a link that is shared by using a QR code or a link?

Example (please replace the appid and fileid in the link with actual IDs):

```
http://play.video.qcloud.com/qrplayer.html?appid=1251769111&fileid=14651978969211156176147&autoplay=0&sw=640&sh=426&$def=20&wmode=transparent
```

```
http://play.video.qcloud.com/iplayer.html?appid=1251769111&fileid=14651978969211156176147&autoplay=0&sw=1800&sh=1200&def=20&wmode=transparent
```

#### Case 8: How to specify video playback definition?

Use the "definition" parameter to specify the definition with which the video is played (on condition that the definition is available for this video). This is available for two playback methods: play with video ID and play with address. See [Parameter Description](#) link for more information. For example:

```
var option = {
  "file_id": "14651978969261415426",
  "app_id": "1251606588",
  "definition": 30,
  "width": 800,
  "height": 700
};

var player = new qcVideo.Player("id_video_container", option);
```

## Overview of API Methods

### Constructor

```
qcVideo.Player(id, option, listener);
```

**ID:** String; **Required;** This parameter indicates the ID of the container where the player is located in the page and can be customized.

**option:** Object ; **Required.**

This parameter indicates the options that can be set for player's parameters. The options are as follows:

Parameter	Type	Default Value	Description
file_id	String	None	Unique ID of the VOD file. This is <b>Required</b> when playing video by video ID
app_id	String	None	The parameter is <b>Required</b> if the LVB video is played using video ID. For the videos under the same account, this parameter remains the same.
width	Number	None	<b>Required</b> , used to configure player width (in pixel). Example: 640
height	Number	None	<b>Required</b> , used to configure player height (in pixel). Example: 480
auto_play	Number	0	Whether auto playback is allowed. 0: Disable; 1: Enable <b>Note: This parameter only applies to Flash players on PC.</b>
disable_full_screen	Number	0	Whether full-screen mode is allowed. 0: Enable; 1: Disable <b>Note: This parameter only applies to Flash players on PC.</b>
disable_drag	Number	0	Whether users are allowed to drag the video progress bar. 0: Allow; 1: Disallow <b>Note: This parameter only applies to Flash players on PC.</b>

Parameter	Type	Default Value	Description
stretch_full	Number	0	Whether the video is scaled up to the same size as the player. 0: Do not scale up. 1: Scale up to full screen <b>Note: This parameter only applies to Flash players on PC.</b>
stop_time	Number	None	Trial mode. For example, if you set it to "60", the video playback stops in 60 seconds, and the "playStatus" event is triggered at the same time
remember	Number	0	Whether to remember last played position. 0: No. 1: Yes. If enabled, the time point when the video was stopped in the previous playback will be recorded and the next playback will start from this position. <b>Note: This parameter only applies to Flash players on PC.</b>
playbackRate	Number	1	Playback speed. For example, "2" means the video will be played at double speed, while "0.5" means half speed. <b>Note: This option is only applies to H5 players</b>
hide_h5_setting	Boolean	false	Whether to hide the H5 Settings button. true: Hide. false: Do not hide
hide_h5_error	Boolean	false	Whether to hide error messages prompted by H5. <b>Note: This parameter only applies to H5 players.</b>
WMode	String	window	When in window mode, you cannot put other page elements over the Flash player. You can change this into opaque or parameter values for other flash wmode if required. <b>Note: This parameter only applies to Flash players on PC.</b>
stretch_patch	Boolean	false	If configured as "true", the video ad that is displayed when the video starts, ends or pauses will be enlarged to cover the whole player.
definition	Number	None	Used to specify the definition with which the video will be played. The configured definition must be available for the video. Available values: 10, 20, 30, 40, 210, 220, 230, 240. For more information about the relation between these values and video types, see the parameter descriptions for <a href="#">third_video</a> .
videos	Array	None	When hotlink protection is enabled, you can achieve player playback by configuring an accessible video address for "videos". The definition type is obtained by matching the URL with the URL prefix queried through the backend. For more information, please see <a href="#">User Guide on Hotlink Protection</a> . For example: [ <code>http://xxx.myqcloud.com/xxxyy\_f220.m3u8? **sign**=xxx</code> , ... ]

Parameter	Type	Default Value	Description
third_video	Object	None	<p>This option is only used when playing videos by using video addresses.</p> <p>Parameter Example:  <code>{'duration': 20, //Video duration (in sec). Optional parameter. If not passed, the video duration will be automatically updated when MetaData is loaded.</code></p> <p><b>Note: This parameter is required in case of MP4 playback.</b>  <code>'urls': { // (At least one address must be included. Be careful about the video format)</code>  <code>10: "address for mp4 mobile videos",</code>  <code>20: "address for mp4 SD videos",</code>  <code>30: "address for mp4 HD videos",</code>  <code>40: "address for mp4 ultra high definition videos",</code>  <code>210: "address for hls mobile videos",</code>  <code>220: "address for hls SD videos",</code>  <code>230: "address for hls HD videos",</code>  <code>240: "address for hls ultra high definition videos"</code>  <code>}</code>  <code>}</code></p> <p><b>Note: If you simulate a mobile device in Chrome or other PC browsers, please use an address for MP4 videos.</b></p>

**listener:** Object; Optional; List of callback functions in case of playing status change.

Function Name	Type	Description
fullScreen	function	<p>Triggered when entering/exiting full-screen mode. Callback function parameter: isFullScreen: Boolean</p> <p>Returned value: true: enter full screen, false: exit full screen</p> <p>Example: <code>function(isFullScreen){ ... }</code></p> <p><b>Note: This event only applies to PC Platform Flash Player</b></p>
playStatus	function	<p>Triggered when playback status changes. Callback function parameter "status": String</p> <p>Returned value: ready: "Player is ready"; seeking: "Search"; suspended: "Pause"; playing: "Playback in progress"; playEnd: "Playback ended"; stop: "Triggered by the end of trial duration"; error: "Triggered in case of H5 playback error"</p> <p>Example: <code>function(status, msg){ ... }</code></p>
dragPlay	function	<p>Triggered when you drag and change the progress bar; second: Number</p> <p>Returned value: Final progress bar position (in sec)</p> <p>Example: <code>function(second){ ... }</code></p> <p><b>Note: This event only applies to PC Platform Flash Player</b></p>

## Obtain parameter and status

Here are the methods for obtaining parameters and statuses for player object that is returned by the constructor

Method	Returned Value	Description
getVolume	Number. Value range: 0-1	Obtain the current volume
getDuration	Number (in sec)	Obtain the total duration of the current video
getCurrentTime	Number (in sec)	Obtain the current playback position
isSeeking	Boolean ; true indicates "loading"	Whether the current playback status is "loading"
isSuspended	Boolean ; true indicates "paused"	Whether the current playback status is "paused"
isPlaying	Boolean ; true indicates "playing"	Whether the current playback status is "playing"
isPlayEnd	Boolean ; true indicates "playback ended"	Whether the current playback status is "playback ended"
getWidth	Number(int)	Get the width of current player
getHeight	Number(int)	Get the height of current player
getClarity	Number(int) (1: "Mobile", 2: "Standard definition", 3: "High definition", 4: "Ultra high definition")	Get the current video definition
getAllClaritys	Array<int> ( 1: "Mobile", 2: "Standard definition", 3: "High definition", 4: "Ultra high definition")	Get all available definitions for the current video

## Settings and actions

The player objects returned by the constructors can be set using the following methods:

Method	Description
resize(width,height)	Parameter: width: int; height: int Function: Set the width and height for current player. Returned value: none
play(second)	Parameter: second: int (in sec) Function: Start playback. You can specify a time point at which the video starts to play Returns: int <a href="#">Error Codes</a> Note: "second" can only be a null value or 0 when you play a video by using a video address
pause()	Function: Pause the playback of current video Returned value: int <a href="#">Error codes</a>
resume()	Funtion: Resume playback. Returned value: int <a href="#">Error codes</a>

Method	Description
setClarity(clarity)	<p>Parameter: clarity, int definition. Value range: (1: "Mobile", 2: "Standard definition", 3: "High definition", 4: "Ultra high definition")</p> <p>Function: Change video definition</p> <p>Returned value: Int <a href="#">Error Codes</a></p> <p>Note: Make sure that the definition is available for the current video before configuring it using "clarity", otherwise the player may choose a definition based on the default player rules</p>
changeVideo(opt)	<p>Parameter: opt Object; Contains the basic information of the video to be played. This is nearly identical to the "second" parameter of the constructor. For more information, please see <a href="#">Constructor Instruction</a></p> <p>Function: Change video dynamically</p> <p>Returned value: int <a href="#">Error Codes</a></p>
addBarrage(barrage)	<p>Parameter: barrage, array barrage information</p> <pre>[{   "type":"content", // Message type, content: plain text (<b>Required</b>)   "content":"hello world", // Text message (<b>required</b>)   "time":"1.101" ,//The time length (in sec) between the moment when the current method is called for adding a caption and the moment when the caption is displayed. (<b>Required</b>)   "style": "C64B03;35" ,// Separated by semicolons; the first is color value, and the second is font size (optional)   "postion":"center" // Location   center: center, bottom: bottom, up: top (optional) }, ... ]</pre> <p>Function: Add on-screen comments</p> <p>Returned value: int <a href="#">Error Codes</a></p> <p>Note: <b>On-screen comment is only implemented at frontend, and backend functions should be self-developed. This function only applies to Flash players on PC.</b></p>
closeBarrage()	<p>Function: disable on-screen comment. Call addBarrage again to re-enable the feature.</p> <p>Returned value: int <a href="#">Error codes</a></p> <p>Note: <b>On-screen comment is only implemented at frontend, and backend functions should be self-developed. This function only applies to Flash players on PC.</b></p>

The common error codes of the above methods are as follows:

Error Code	Description
200	Operation successful
0	Player not fully initialized
-1	Failed to change video dynamically. Required parameter is missing
-2	Unknown operation command
-3	Playback time is beyond valid playback range

## Video File Upload

Users can use VOD Web SDK to upload videos. Tencent Cloud Video users can therefore upload video files using Web. The SDK supports uploading via HTML5, but it's not possible for browsers that do not support HTML5.

For more information on how to proceed, see [Cloud VOD Web Upload SDK](#).

# Problem solving

Last updated : 2018-07-10 18:10:30

## Error Codes

The table contains error codes that may occur when using the SDK. In case of any error code that is not listed in this table, contact customer service. Our engineers can help you solve the problem.

Code	Description	
1003	Incorrect password	
10000	Request timeout (timeout when pulling player configuration and video information. Check your network and try again. Timeout is 10 seconds)	
10001	Failed to parse data (Failed to parse the data obtained by pulling player configuration and video information. It may be caused by a network problem or server exception)	
10002	Connection timeout. Try again later	(Failed to pull player configuration and video information. It may be caused by a network problem or server exception)
10008	Incorrect APPID or File ID	
11044	APPID missing	
11045	File ID missing	
11046	Password missing	

## FAQs

- **Why is the screen distortedly stretched when the video is played in H5?**

Video stretching is not supported when playing video in H5. Check if the player container has right width/height configuration.

- **My QQ browser is downloading the video. How to stop it?**

JS cannot intervene such actions due to the restrictions of the QQ mobile browser kernel. Similarly, the kernels of some browsers, such as UC, also provide auto video detection/download feature. Contact your browser developer to disable this feature.

- **The video cannot be hidden on the QQ browser.**

QQ browser takes over the video playback feature from H5, and the X5 kernel uses self-developed player to play videos. QQ browsers use a unified playback interface to ensure a good user experience. For more information, please see [QQ Browser Documentation](#).

- **I didn't get the correct status information when calling relevant methods such as `isPlaying()`.**

In some mobile browsers and webviews, video playback will be taken over by the browser's kernel, which means the SDK will not be able to acquire the correct playback status.

- **Video cannot be automatically played on mobile device after auto playback is set.**

Most mobile browsers cannot automatically load media files due to reasons like data traffic. Users need to trigger this action manually when playing videos.

- **Video is automatically played in full screen mode on iOS.**

By default, video is played in full screen mode on iOS system due to the webkit setting. To achieve inline playback within an App, you can set the `webkit-playsinline` attribute. Any Safari browser on iOS below 10 is unable to disable the automatic use of full screen mode.

- **Why does the Flash player have two play buttons in Chrome on PC?**

Flash is no longer automatically played starting from Chrome 42. Chrome only plays major Flash contents automatically, while other Flash contents are paused, unless users enable them manually.