天翼云云点播

SDK/API 接口文档

V 2.0

天翼云科技有限公司

目录

1. 简介

天翼云云点播提供了点播、转码能力，您可以将上传的视频转码成为适合在互联网进行传播的格式，这些格式主要是 H.264 编码的 MP4, HLS 和 FLV。您可以在用户控制台或者相关的后台 SDK 中取得这些视频的链接。

为了更加便利地使用点播功能，我们也提供基于 videojs 封装的视频播放器，以下是 Web 播放器的相关接口使用方法以及相关的接口。

请注意，本文档中的播放器使用的是传统的通过 `<script>` 标签引入的。通过 Node.js 模块化引入的我们后续推出。

2. 阅读对象

本文档的读者，我们假设您已经具备一些基本的 HTML 以及 Javascript 基础。同时您已经在天翼云开通云点播服务，并且已经知道如何从该服务中取得视频的 URL。

3. 初始化 Web 播放器

Step 1: 在页面中引入文件

请将下载好的播放器文件解压缩，并且放在一个 Web 服务器中。然后在 HTML 文件中引入以下的 css 和 js 文件。

```
<link href="//path/to/ctyunplayer.css" rel="stylesheet">
<script src="//path/to/ctyunplayer.js"></script>
```

Step 2: 在页面中放置播放器容器

播放器使用的是 HTML 5 的 `<video>` 标签，请在需要展示视频的地方引入以下容器，其中容器 id，高宽都可以自定义。

`<source>` 中的 `src` 是播放视频的 url，可以在天翼云云点播服务中取得。而 `type` 字段则是有以下对应规则：

- 如果视频格式是 mp4，则该字段应为 `video/mp4`
- 如果视频格式为 hls，则该字段应为 `application/x-mpegURL`

```
<video id="ctyunplayer-container" width="414" height="270" preload="tru
e" playsinline>
    <source src="//path/to/ctyun/xstore/transcoded/video.hls" type="app
```

```
lication/x-mpegURL">
</video>
```

Step 3: 通过 JS 代码进行初始化

在页面加载完之后（通常会在 windows 的 load 事件中 或者在 HTML 的页面尾部的 <script> 标签中）执行以下 javascript 初始化代码

```
var player = ctyunplayer("ctyunplayer-container");
player.play();
```

4. Player 对象接口

在初始化了 player 之后，除了播放器原本的提供的按钮之外，您还可以通过以下的接口来实现相关的功能：

4.1. 禁止使用右键

```
player.disableContextMenu();
```

4.2. 恢复使用右键

```
player.enableContextMenu();
```

4.3. 禁止使用画中画

Chrome 以及同内核的浏览器适用。

```
player.disablePictureInPicture();
```

4.4. 恢复使用画中画

Chrome 以及同内核的浏览器适用。

```
player.enablePictureInPicture();
```

4.5. 设置 URL

设置播放器播放的 url，该 url 的地址为视频的地址，支持 MP4 以及 HLS。

```
player.setUrl("//path/to/video");
```

4.6. 设置 Poster

设置一个封面，该封面在视频未开始播放时显示，该封面应该是一个图片的 URL。

```
player.setPoster("//path/to/poster");
```

4.7. 设置允许播放视频的时长

设置当前视频可以播放的时长，该功能通常用在试看的场景，在播放到视频的对应时长之后，播放器就不会继续播放后续视频了。参数的单位为秒。

```
player.setPlayableVideoDuration(300); // 最多播放到5 分钟
```

4.8. 设置片头广告

设置片头广告，片头广告播放时，播放器将不会显示控制栏，用户不能跳过广告直接观看视频。

```
player.setStartAd("//path/to/start/ad/video");
```

4.9. 设置片尾广告

设置片尾广告，在播放完视频之后继续播放该广告视频，用户不能暂停和跳过这个广告。

```
player.setEndAd("//path/to/end/ad/video");
```

4.10. 设置暂停广告

设置暂停广告，在视频暂停时会展示这个广告，该广告仅支持图片类型。

```
player.setPauseAd("//path/to/pause/ad/picture");
```

4.11. 加载视频内容

开始下载视频信息，如果您的 `<video>` 标签没有设置 `preload="true"`，可以通过这个方法开始来加载您的视频。

```
player.load();
```

4.12. 播放视频

播放视频，请注意，在您重新设置了 URL 之后，该方法会从片头广告开始播放而不是设置的视频。

```
player.play();
```

4.13. 暂停视频

调用该方法之后，如果设置了暂停广告，播放器会显示暂停广告。

```
player.pause();
```

4.14. 停止视频

停止播放视频，视频指针直接指导视频最后。

```
player.stop();
```

4.15. 从指定时间播放视频

让视频从指定时间开始播放。其中的参数单位为毫秒。

```
player.seekTo(3000); // 从第 3 秒开始播放
```

4.16. 获取视频的时长

获取当前播放视频的时长。

```
var duration = player.getDuration();
```

4.17. 获取当前播放的位置

获取当前播放到视频的哪个时间，返回的单位为毫秒。

```
var currentPosition = player.getCurrentPosition();
```

4.18. 获取当前视频预加载的时间范围

返回一个对象，该对象描述了当前播放器预加载的视频的时间范围，该对象包含了 from 和 to 两个字段。

```
var range = player.getBufferedRange();
var from = range.from;
var to = range.to;
```

4.19. 获取当前加载的时间的结束时间

如果您的视频是从头开始加载的话，您可以调用以下方法，该方法直接返回最终的加载的最终时间点。

```
var bufferedEnd = player.getBufferedEnd();
```

4.20. 获取当前加载了视频的百分比

获取当前已经加载了视频的百分比，返回 0 ~ 1 之间的浮点数。

```
var percentage = player.getBufferedPercentage();
```

4.21. 设置音量

设置音量，从 0～1 之间的浮点数，精确到小数点后 2 位。

```
player.setVolume(0.30);
```

4.22. 设置静音

设置为静音

```
player.mute();
```

4.23. 恢复音量

恢复到静音之前的音量，仅在当前为静音状态下有效。

```
player.resumeVolume();
```

4.24. 查询当前是否是静音状态

```
var ismuted = player.isMuted();
```

4.25. 进入全屏

```
player.enterFullScreen();
```

4.26. 退出全屏

```
player.existFullScreen();
```

4.27. 设置播放倍数

传入值为浮点数，可以为 0.5 或者 2.0 等均可。

```
player.setSeed(4.0);
```

4.28. 检查视频是否正在播放

```
var isplaying = player.isPlaying();
```

4.29. 检查是否暂停播放

```
var ispaused = player.isPaused();
```

4.30. 检查是否停止

```
var isstop = player.isStop();
```

4.31. 检查片头广告是否在播放中

```
var isAdPlaying = player.isStartAdPlaying();
```

4.32. 检查片尾广告是否在播放中

```
var isEndAdPlaying = player.isEndAdPlaying();
```

4.33. 暂停广告是否显示

```
var isShown = player.isPausedAdShow();
```

4.34. 设置水印

```
player.setWatermark({
    /**
     * 水印文字
     */
    "text": "水印文字",
    /**
     * 水印样式，可选，可以不填写。
     */
    "className": "",
    /**
     * 水印内连样式
     */
    "style": {
        "top": "20px",
        "left": "3%",
        // "right": "",
        // "bottom": "",
        "fontSize": "26px",
        "color": "red"
    }
});
```

4.35. 设置文字水印

不建议使用，请使用 setWatermark 方法代替！

```
player.setWatermarkText("watermark", {
    "top": "20px",
    "left": "3%",
    // "right": "",
    // "bottom": "",
    "fontSize": "26px",
    "color": "red"
});
```

4.36. 设置弹幕

设置弹幕，弹幕的数据结构如下：

```
var floatingTextArr = [{
    "value": "弹幕内容", // 弹幕内容
    "time": 1, // 显示弹幕的时间，单位为秒
    "color": 'red', // 弹幕的颜色
    "speed": 1, // 弹幕的速度
    "fontSize": 22, // 弹幕的字号
    "position_min": 0, // 弹幕浮动展示在屏幕的位置高度范围，小值，0~1，默认为 0
    "position_max": 1 // 弹幕浮动展示在屏幕的位置高度范围，大值，0~1，默认为 1
}]; // 这是一个列表，可以有多条弹幕数据
player.setFloatingComments(floatingTextArr);
```

4.37. 增加弹幕

为当前的弹幕列表增加弹幕内容。

```
var floatingTextArr = [{
    "value": "弹幕内容", // 弹幕内容
    "time": 1, // 显示弹幕的时间，单位为秒
    "color": 'red', // 弹幕的颜色
    "speed": 1, // 弹幕的速度
    "fontSize": 22, // 弹幕的字号
    "position_min": 0, // 弹幕浮动展示在屏幕的位置高度范围，小值，0~1，默认为 0
    "position_max": 1 // 弹幕浮动展示在屏幕的位置高度范围，大值，0~1，默认为 1
}]; // 这是一个列表，可以有多条弹幕数据
player.addFloatingComments(floatingTextArr);
```

4.38. 停止播放片头广告

```
player.stopStartAd();
```

4.39. 停止播放片尾广告

```
player.stopEndtAd();
```

4.40. 隐藏当前的暂停广告

```
player.hidePausedAd();
```

4.41. 销毁当前播放器

```
player.dispose();
```

4.42. 事件

播放器在执行指定动作时会触发事件，通过以下事件注册机制可监听到相应动作时执行对应函数。请注意，这些监听事件并不不像 JS 自带的 `on("type", function())` 那样会不断叠加您的函数，也没有提供 `off` 接口，因此，您后续设置的函数会覆盖前面设置的。

4.43. 在内容开始加载时

```
player.onLoadStart(function(){

});
```

4.44. 在开始播放时

开始播放开始时执行，该注册的方法会在播放片头广告之前执行。

```
player.onPlay(function(){

});
```

4.45. 在开始播放正片时

```
player.onPlayMain(function(){

});
```

4.46. 在暂停时

```
player.onPause(function(){

});
```

4.47. 在指定播放事件时

```
player.onSeekedTo(function(){

});
```

4.48. 在停止时

```
player.onStop(function(){

});
```

4.49. 在正片停止时

```
player.onStopMain(function(){

});
```

4.50. 在进入全屏时

```
player.onEnterFullScreen(function(){

});
```

4.51. 在退出全屏时

```
player.onExistFullScreen(function(){

});
```

4.52. 在错误时

```
player.onError(function(){

});
```