uniapp 安卓视频播放器UTS原生插件

目　录

使用方法

介绍

安卓视频播放器UTS原生插件支持播放本地视频，网络视频，直播等多种视频，支持多种滤镜，支持切换音轨，支持弹幕，支持生成gif图片，视频截图等多种功能

相关链接

安卓YYEVAPlayer MP4礼物播放器原生插件
安卓视频播放器原生插件
安卓VLC视频播放器UTS原生插件

插件使用注意事项

1. 示例文件不包含插件，需要您在插件首页点击“试用”导入插件到项目中，插件导入后打自定义基座，运行项目的时候选择运行到基座即可
2. 组件只能在nvue/uvue页面中使用，不支持vue页面

联系作者

关注微信公众号可联系作者

插件地址

https://ext.dcloud.net.cn/plugin?id=22159

示例文件下载

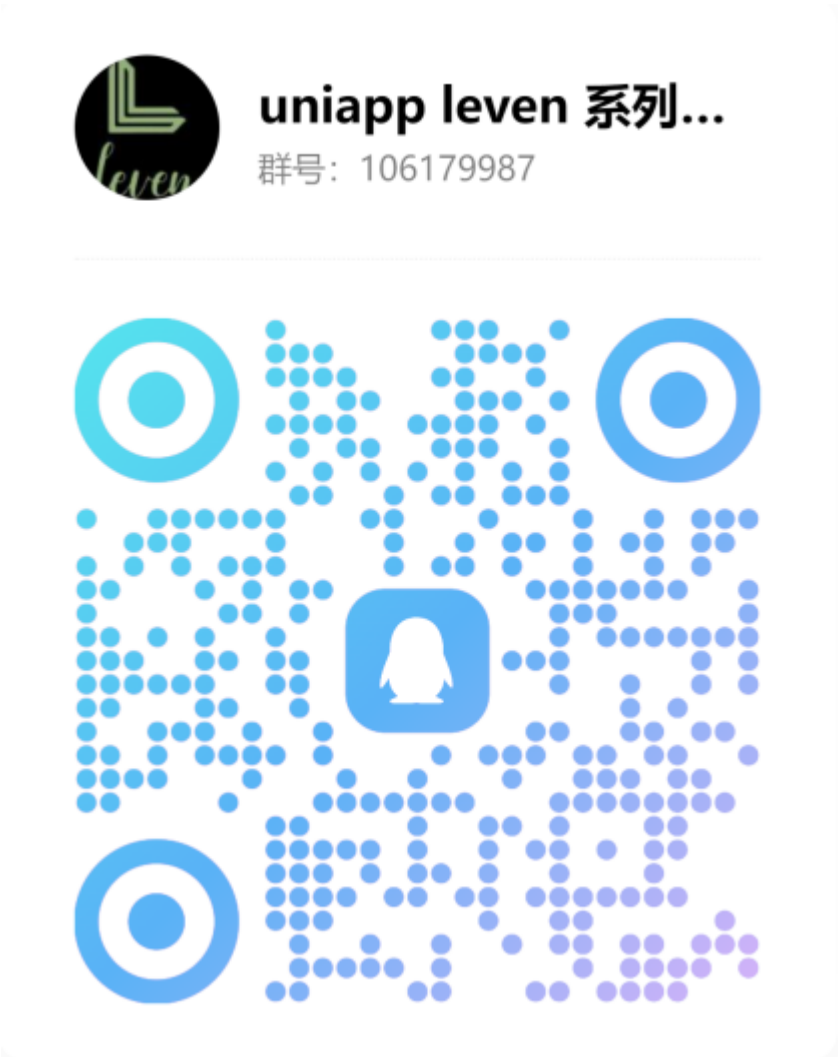
1. 插件首页下载（推荐）

   > 通过“使用 HBuilderX 导入示例项目”可直接创建uniapp项目

2. 本地下载，点击这里即可下载

演示程序下载

加群下载演示程序



权限

1. android.permission.READ_EXTERNAL_STORAGE
2. android.permission.WRITE_EXTERNAL_STORAGE

API使用方法

使用方法

在需要使用插件的页面加载以下代码

```
import * as module from "@/uni_modules/leven-uts-videoPlayer"
```

组件使用方法

在需要使用插件的页面加载以下代码

```
<leven-uts-videoPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :confi
          @onStartPrepared="onStartPrepared" @onPrepared="onPrepared" @onPlayErro
        </leven-uts-videoPlayer>
```

> 具体的使用示例可参考示例文件

更新日志

2025-02-12

首次发布

弹幕文件配置说明

弹幕内容示例

```
<?xml version="1.0" encoding="UTF-8"?>
<i>
    <chatserver>chat.bilibili.com</chatserver>
    <chatid>2962351</chatid>
    <mission>0</mission>
    <maxlimit>1500</maxlimit>
    <source>k-v</source>
    <d p="145.91299438477,1,25,16777215,1422201001,0,D6673695,757075520">233333333
    <d p="140.26400756836,1,25,16777215,1422200985,0,D4c73761,757075551">这.....</
    <d p="136.92900085449,1,25,16777215,1422201038,0,D159c8e9,757076123">233333333
</i>
```

> 弹幕内容为xml格式，解析内容为d标签，如：

```
<d p="23.826000213623,1,25,16777215,1422201084,0,057075e9,757076900">我从未见过如此厚
```

> p属性为弹幕的出现时间以及字号等，逗号分割
> 0:时间(弹幕出现时间)
> 1:类型(1从右至左滚动弹幕|6从左至右滚动弹幕|5顶端固定弹幕|4底端固定弹幕|7高级弹幕|8脚本弹幕)
> 2:字号
> 3:颜色
> 4:时间戳
> 5:弹幕池id
> 6:用户hash
> 7:弹幕id

申请插件所需权限

方法名

`requestPermissions`

用法

- 用法如下：

```
module.requestPermissions(res => {
  console.log(JSON.stringify(res))
});
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "grantedList": [
      "android.permission.READ_EXTERNAL_STORAGE",
      "android.permission.WRITE_EXTERNAL_STORAGE"
    ]
  },
  "message": "权限被拒绝",
  "code": -1
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

申请插件所需权限

参数名	参数类型	参数描述
data.grantedList	Array	申请的权限列表，**被拒绝的时候会返回**
code	Integer	返回类型，0.成功，其他：失败

使用方法

组件使用

在需要使用插件的页面加载以下代码

```
<leven-uts-videoPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config
          @onStartPrepared="onStartPrepared" @onPrepared="onPrepared" @onPlayError
        </leven-uts-videoPlayer>
```

> 组件只能在nvue/uvue页面中使用，不支持vue页面，config参数必传，具体请参考【初始化配置】

初始化配置

config

> 组件初始化配置，必传参数

参数说明

参数名	参数类型	是否必填	默认值	参数描述
url	String	是	无	视频播放地址或文件、文件路径，如果是内部存储文件需要传绝对路径
backgroundImage	String	否	无	播放器预览背景图片，支持网络地址
frame	Integer	否	无	视频帧背景图片，以视频帧作为图片，单位：微秒
error	String	否	无	视频图片加载失败显示的图片，需放置到res/mipmap中
placeholder	String	否	无	图片预加载资源图片，需放置到res/mipmap中
cacheWithPlay	Boolean	否	true	是否边播放边缓存
titleVisible	Boolean	否	true	标题是否显示
title	String	否	无	视频标题
backButtonVisible	Boolean	否	true	返回按钮是否显示
isTouchWidget	Boolean	否	true	是否可以滑动界面改变进度，声音等
needShowWifiTip	Boolean	否	true	是否需要流量提示

初始化配置

参数名	参数类型	是否必填	默认值	参数描述
loop	Boolean	否	false	是否循环播放
bottomContainer	Boolean	否	true	是否显示底部控制条
startButtonVisible	Boolean	否	false	是否显示开始播放按钮
startPosition	Integer	否	0	开始播放位置，目前有时候前几秒有跳动问题，毫秒
header	Object	否	无	请求头信息
enableMediaCodec	Boolean	否	false	是否开启硬解码
isMediaCodecTexture	Boolean	否	false	是否开启硬解码渲染优化
scaleType	String	否	SCREEN_TYPE_DEFAULT	视频播放比例，可选值：SCREEN_TYPE_DEFAULT、SCREEN_TYPE_16_9、SCREEN_TYPE_4_3、SCREEN_TYPE_18_9、SCREEN_TYPE_FULL、SCREEN_MATCH_FULL

参数名	参数类型	是否必填	默认值	参数描述
renderType	String	否	TEXTURE	播放模式，可选值： GLSURFACE：GLSurfaceView 主要用于OpenGL 渲染的，如果支持滤镜，必须选择此模式， SURFACE：SurfaceView，与动画全屏的效果不是很兼容， TEXTURE：TextureView
waterImage	Object	否	无	水印图片，只支持本地图片，需放置到res/mipmap下
waterImage.image	String	否	无	图片名称，不需要加扩展名
waterImage.width	Integer	否	无	图片宽度
waterImage.height	Integer	否	无	图片高度
waterImage.alpha	float	否	无	图片透明度
waterImage.position	Integer	否	无	水印位置，0：右上角，1.右下角，2.左上角,3.左下角,4.居中,5.右中,6.上中,7.下中,8.左中
waterImage.offset	float	否	1.0	位置偏移量
kernel	Integer	否	1	播放器内核，1.IJK内核（默认），2.EXO内核，3.系统内核

初始化配置

参数名	参数类型	是否必填	默认值	参数描述
isMute	Boolean	否	false	是否静音播放
delay	Integer	否	0	生成gif的帧之间延时
sampleSize	Integer	否	1	生成gif采样率
scaleSize	Integer	否	5	生成gif缩放比例
frequencyCount	Integer	否	50	生成gif截图频率，毫秒
danmakuParams	Object	否	无	弹幕配置
danmakuParams.danmakuShow	Boolean	否	true	是否开启弹幕
danmakuParams.danmakuFile	String	否	无	初始化弹幕文件，xml文件，如果是raw文件的话不需要加扩展名，其他需要加上扩展名，具体文件格式请参考【弹幕文件配置说明】
danmakuParams.danmakuFileType	String	否	无	弹幕文件类型，url：网络地址，path：本地路径，raw：raw文件
danmakuParams.maxLines	Integer	否	5	最大显示行数
danmakuParams.preventOverlapping	Boolean	否	true	防弹幕重叠
danmakuParams.danmakuStyle	Object	否	无	描边样式

参数名	参数类型	是否必填	默认值	参数描述
danmakuParams.danmakuStyle.style	String	否	DANMAKU_STYLE_STROKEN	样式，可选值：DANMAKU_STYLE_DEFAULT：自动、DANMAKU_STYLE_NONE：无、DANMAKU_STYLE_SHADOW：阴影、DANMAKU_STYLE_STROKEN：描边、DANMAKU_STYLE_PROJECTION：投影
danmakuParams.danmakuStyle.values	Array[float]	否	[3]	描边值，DANMAKU_STYLE_SHADOW 阴影模式下，values传入阴影半径，DANMAKU_STYLE_STROKEN 描边模式下，values传入描边宽度，DANMAKU_STYLE_PROJECTION 投影模式下，values传入 offsetX, offsetY, alpha offsetX/offsetY: x/y 方向上的偏移量 alpha: 投影透明度 [0...255]
danmakuParams.duplicateMergingEnabled	Boolean	否	true	是否启用合并重复弹幕

初始化配置

参数名	参数类型	是否必填	默认值	参数描述
danmakuParams.scrollSpeedFactor	float	否	1.0	设置弹幕滚动速度系数,只对滚动弹幕有效
danmakuParams.scaleTextSize	float	否	1.0	设置弹幕文本大小
isAutoPlay	Boolean	否	true	是否自动播放

组件方法

开始播放

方法名

`play`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.play({
    //播放地址，rtmp://liteavapp.qcloud.com/live/liteavdemoplayerstreamid
    //http://devimages.apple.com.edgekey.net/streaming/examples/bipbop_4x3/g
    //rtsp://rtspstream:effd2f46af6aef62a77b62104ceafbc0@zephyr.rtsp.stream/
    //http://alvideo.ippzone.com/zyvd/98/90/b753-55fe-11e9-b0d8-00163e0c0248
    // url: "http://vod.ulunix.cn/media_hls/X6-1-wAD-1yQ7R2q-bCJbio780yeu41i
    //https://res.exexm.com/cw_145225549855002
    //https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%
    url: "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5
    //播放器预览背景图片，支持网络地址
    //http://www.yeyuboke.com/svga/image_default.jpg
    backgroundImage: "",
    //视频帧背景图片，以视频帧作为图片，单位：微秒
    frame: 1000000,
    //视频帧图片加载失败显示的图片，需放置到res/mipmap中
    error: "error",
    //图片预加载资源图片，需放置到res/mipmap中
    placeholder: "placeholder",
    //是否边播放边缓存
    cacheWithPlay: true,
    //标题是否显示
    titleVisible: false,
    //视频标题
    title: "安卓高质量视频播放器原生插件",
    //返回按钮是否显示
    backButtonVisible: false,
    //是否可以滑动界面改变进度，声音等
    isTouchWidget: true,
    //是否需要流量提示
    needShowWifiTip: true,
    //是否循环播放
    loop: false,
    //是否显示底部控制条
    bottomContainer: true,
    //是否显示开始播放按钮
```

```
        startButtonVisible: false,
        //开始播放位置，目前有时候前几秒有跳动问题，毫秒
        startPosition: 0,
        //请求头信息
        header: {
          ee: "333",
          allowCrossProtocolRedirects: "true",
          "User-Agent": "LEVEN"
        },
        //是否开启硬解码
        enableMediaCodec: false,
        //是否开启硬解码渲染优化
        isMediaCodecTexture: false,
        //视频播放比例，可选值：
        //SCREEN_TYPE_DEFAULT，SCREEN_TYPE_16_9，SCREEN_TYPE_4_3
        //SCREEN_TYPE_18_9，SCREEN_TYPE_FULL，SCREEN_MATCH_FULL
        scaleType: "SCREEN_TYPE_DEFAULT",
        //播放模式，可选值：GLSURFACE：GLSurfaceView 主要用于OpenGL渲染的，如果支持滤镜，
        renderType: "GLSURFACE",
        //水印图片，只支持本地图片，需放置到res/mipmap下
        waterImage: {
          //图片名称，不需要加扩展名
          image: "uni_icon",
          //图片宽度
          width: 100,
          //图片高度
          height: 100,
          //透明度
          alpha: 0.6,
          //水印位置，0：右上角，1.右下角，2.左上角,3.左下角,4.居中,5.右中,6.上中,7.下中,8
          position: 3,
          //位置偏移量，默认：1.0
          offset: 1.5
        },
        //默认播放内核
        kernel: 2,
        //是否静音播放
        isMute: false,
        //gif的帧之间延时,默认：0
        delay: 0,
        //生成gif采样率，默认：1
        sampleSize: 1,
        //生成gif缩放比例，默认：5
        scaleSize: 5,
        //生成gif截图频率，毫秒，默认：50
        frequencyCount: 50,
        //弹幕配置
        danmakuParams: {
          //是否开启弹幕
          danmakuShow: false,
```

开始播放

```
        //初始化弹幕文件，xml文件，如果是raw文件的话不需要加扩展名，其他需要加上扩展名
        //http://www.yeyuboke.com/uniplugin/player/danmaku.xml
        ///storage/emulated/0/Download/WeiXin/danmaku.xml
        danmakuFile: "comments",
        //弹幕文件类型，url：网络地址，path：本地路径，raw：raw文件
        danmakuFileType: "raw",
        //最大显示行数
        maxLines: 3,
        //防弹幕重叠
        preventOverlapping: true,
        //描边样式
        danmakuStyle: {
          //样式，可选值：
          //DANMAKU_STYLE_DEFAULT：自动
          //DANMAKU_STYLE_NONE：无
          //DANMAKU_STYLE_SHADOW：阴影
          //DANMAKU_STYLE_STROKEN：描边（默认）
          //DANMAKU_STYLE_PROJECTION：投影
          style: "DANMAKU_STYLE_STROKEN",
          //描边值，DANMAKU_STYLE_SHADOW 阴影模式下，values传入阴影半径
          //DANMAKU_STYLE_STROKEN 描边模式下，values传入描边宽度
          //DANMAKU_STYLE_PROJECTION 投影模式下，values传入offsetX, offsetY, alph
          values: [3]
        },
        //是否启用合并重复弹幕，默认：true
        duplicateMergingEnabled: true,
        //设置弹幕滚动速度系数,只对滚动弹幕有效，默认：1.0
        scrollSpeedFactor: 1.2,
        //设置弹幕文本大小，默认：1.0
        scaleTextSize: 1.2
      }
    }, res => {
    this.writeLog("开始播放:" + JSON.stringify(res))
  });
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
url	String	是	无	视频播放地址或文件、文件路径，如果是内部存储文件需要传绝对路径
backgroundImage	String	否	无	播放器预览背景图片，支持网络地址

开始播放

参数名	参数类型	是否必填	默认值	参数描述
frame	Integer	否	无	视频帧背景图片，以视频帧作为图片，单位：微秒
error	String	否	无	视频图片加载失败显示的图片，需放置到res/mipmap中
placeholder	String	否	无	图片预加载资源图片，需放置到res/mipmap中
cacheWithPlay	Boolean	否	true	是否边播放边缓存
titleVisible	Boolean	否	true	标题是否显示
title	String	否	无	视频标题
backButtonVisible	Boolean	否	true	返回按钮是否显示
isTouchWidget	Boolean	否	true	是否可以滑动界面改变进度，声音等
needShowWifiTip	Boolean	否	true	是否需要流量提示
loop	Boolean	否	false	是否循环播放
bottomContainer	Boolean	否	true	是否显示底部控制条
startButtonVisible	Boolean	否	false	是否显示开始播放按钮
startPosition	Integer	否	0	开始播放位置，目前有时候前几秒有跳动问题，毫秒
header	Object	否	无	请求头信息
enableMediaCodec	Boolean	否	false	是否开启硬解码

参数名	参数类型	是否必填	默认值	参数描述
isMediaCodecTexture	Boolean	否	false	是否开启硬解码渲染优化
scaleType	String	否	SCREEN_TYPE_DEFAULT	视频播放比例，可选值：SCREEN_TYPE_DEFAULT、SCREEN_TYPE_16_9、SCREEN_TYPE_4_3、SCREEN_TYPE_18_9、SCREEN_TYPE_FULL、SCREEN_MATCH_FULL
renderType	String	否	TEXTURE	播放模式，可选值：GLSURFACE：GLSurfaceView主要用于OpenGL渲染的，如果支持滤镜，必须选择此模式，SURFACE：SurfaceView，与动画全屏的效果不是很兼容，TEXTURE：TextureView
waterImage	Object	否	无	水印图片，只支持本地图片，需放置到res/mipmap下
waterImage.image	String	否	无	图片名称，不需要加扩展名

开始播放

参数名	参数类型	是否必填	默认值	参数描述
waterImage.width	Integer	否	无	图片宽度
waterImage.height	Integer	否	无	图片高度
waterImage.alpha	float	否	无	图片透明度
waterImage.position	Integer	否	无	水印位置，0：右上角，1.右下角，2.左上角,3.左下角,4.居中,5.右中,6.上中,7.下中,8.左中
waterImage.offset	float	否	1.0	位置偏移量
kernel	Integer	否	1	播放器内核，1.IJK内核（默认），2.EXO内核，3.系统内核
isMute	Boolean	否	false	是否静音播放
delay	Integer	否	0	生成gif的帧之间延时
sampleSize	Integer	否	1	生成gif采样率
scaleSize	Integer	否	5	生成gif缩放比例
frequencyCount	Integer	否	50	生成gif截图频率，毫秒
danmakuParams	Object	否	无	弹幕配置
danmakuParams.danmakuShow	Boolean	否	true	是否开启弹幕

开始播放

参数名	参数类型	是否必填	默认值	参数描述
danmakuParams.danmakuFile	String	否	无	初始化弹幕文件，xml文件，如果是raw文件的话不需要加扩展名，其他需要加上扩展名，具体文件格式请参考【弹幕文件配置说明】
danmakuParams.danmakuFileType	String	否	无	弹幕文件类型，url：网络地址，path：本地路径，raw：raw文件
danmakuParams.maxLines	Integer	否	5	最大显示行数
danmakuParams.preventOverlapping	Boolean	否	true	防弹幕重叠
danmakuParams.danmakuStyle	Object	否	无	描边样式
danmakuParams.danmakuStyle.style	String	否	DANMAKU_STYLE_STROKEN	样式，可选值：DANMAKU_STYLE_DEFAULT：自动、DANMAKU_STYLE_NONE：无、DANMAKU_STYLE_SHADOW：阴影、DANMAKU_STYLE_STROKEN：描边、DANMAKU_STYLE_PROJECTION：投影

开始播放

参数名	参数类型	是否必填	默认值	参数描述
danmakuParams.danmakuStyle.values	Array[float]	否	[3]	描边值，DANMAKU_STYLE_SHADOW 阴影模式下，values传入阴影半径，DANMAKU_STYLE_STROKEN 描边模式下，values传入描边宽度，DANMAKU_STYLE_PROJECTION 投影模式下，values传入 offsetX, offsetY, alpha offsetX/offsetY: x/y 方向上的偏移量 alpha: 投影透明度 [0...255]
danmakuParams.duplicateMergingEnabled	Boolean	否	true	是否启用合并重复弹幕
danmakuParams.scrollSpeedFactor	float	否	1.0	设置弹幕滚动速度系数,只对滚动弹幕有效
danmakuParams.scaleTextSize	float	否	1.0	设置弹幕文本大小

回调

- 示例

```
{
    "message": "",
    "code": 0,
```

开始播放

```
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

暂停播放

方法名

`pause`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.pause(res => {
     this.writeLog("暂停播放:" + JSON.stringify(res))
   });
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

继续播放

方法名

`resume`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.resume(res => {
     this.writeLog("继续播放:" + JSON.stringify(res))
   });
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

停止播放

方法名

`stop`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.stop(res => {
     this.writeLog("停止播放:" + JSON.stringify(res))
   });
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置视频播放比例

方法名

`setScreenScaleType`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setScreenScaleType({
    value: "SCREEN_TYPE_DEFAULT"
  }, res => {
    // console.log(res)
    this.writeLog(JSON.stringify(res))
  })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	String	否	scale_default	播放比例值，请参考下方说明

> 播放比例值说明：
> SCREEN_TYPE_DEFAULT：默认尺寸
> SCREEN_TYPE_16_9：16:9
> SCREEN_TYPE_4_3：4:3
> SCREEN_TYPE_18_9：18:9
> SCREEN_TYPE_FULL：全屏
> SCREEN_MATCH_FULL：全屏拉伸

回调

- 示例

```
{
    "message": "",
    "code": 0,
```

```
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

获取播放属性

方法名

`getPlayerData`

用法

- 用法如下：

```
if (this.$refs.refVideoPlayer) {
  this.$refs.refVideoPlayer.getPlayerData(res => {
    this.writeLog("获取播放器数据:" + JSON.stringify(res))
  });
}
```

- 参数说明

  无

回调

- 示例


```
{
  "data": {
    "header": {
      "allowCrossProtocolRedirects": "true",
      "User-Agent": "LEVEN",
      "ee": "333"
    },
    "currentPosition": 17783,
    "looping": false,
    "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5
    "isPlaying": true,
    "speed": 1,
    "scaleType": "SCREEN_TYPE_DEFAULT",
    "danmakuCurrentTime": 17903,
    "duration": 142656
  },
  "message": "",
```

```
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.header	Object	网络请求的头信息
data.currentPosition	Integer	当前播放位置，单位：毫秒
data.looping	Boolean	是否循环播放
data.url	String	视频地址
data.isPlaying	Boolean	是否正在播放
data.speed	Integer	当前播放速度
data.scaleType	String	视频播放比例
data.danmakuCurrentTime	Integer	弹幕播放当前时间，单位：毫秒
data.duration	Integer	视频时长，单位：毫秒
code	Integer	返回类型，0.成功，其他：失败

设置音量

方法名

`setVolume`

用法

- 用法如下：

```
if (this.$refs.refVideoPlayer) {
  this.$refs.refVideoPlayer.setVolume({
    //音量取值范围：0.0-1.0
    left: 0.3,
    right: 0.3
  }, res => {
    this.writeLog("设置音量:" + JSON.stringify(res))
  });
}
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
left	Integer	是	无	左声道音量
right	Integer	是	无	右声道音量

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

设置音量

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置播放进度

方法名

`seekTo`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.seekTo({
    //播放进度，单位：毫秒
    position: 30000
  }, res => {
    this.writeLog("设置播放进度:" + JSON.stringify(res))
  });
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
position	Integer	是	无	播放进度，单位：毫秒

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

设置播放进度

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

设置播放速度

方法名

`setSpeed`

用法

- 用法如下：

```
if (this.$refs.refVideoPlayer) {
  this.$refs.refVideoPlayer.setSpeed({
    value: 2.5,
  }, res => {
    this.writeLog("设置播放速度:" + JSON.stringify(res))
  });
}
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	播放速度，1：正常播放

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

设置播放速度

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

截图

方法名

`taskShotPic`

用法

- 用法如下：

```
if (this.$refs.refVideoPlayer) {
  this.$refs.refVideoPlayer.taskShotPic(res => {
    this.writeLog("截图:" + JSON.stringify(res))
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "path": "/storage/emulated/0/Pictures/LevenPlayer/1739351162257.jpg"
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.path	String	图片保存路径
code	Integer	返回类型，0.成功，其他：失败

是否正在播放

方法名

`isPlaying`

用法

- 用法如下：

```
if (this.$refs.refVideoPlayer) {
  this.$refs.refVideoPlayer.isPlaying(res => {
    this.writeLog("是否正在播放:" + JSON.stringify(res))
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "isPlaying": true
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.isPlaying	Boolean	是否正在播放
code	Integer	返回类型，0.成功，其他：失败

全屏播放

方法名

`fullScreen`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.fullScreen(res => {
     this.writeLog("全屏播放:" + JSON.stringify(res))
   });
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

退出全屏

方法名

`quitFullScreen`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.quitFullScreen(quitRes => {
     this.writeLog("退出全屏:" + JSON.stringify(quitRes))
   });
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

小窗口播放

方法名

`smallVideo`

用法

- 用法如下：

```
 this.$refs.refVideoPlayer.smallVideo({
  //小窗口大小
  size: [150, 150]
}, res => {
  this.writeLog("小窗口播放:" + JSON.stringify(res))
});
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
size	Array[Integer]	是	无	小窗口尺寸

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

退出小窗口

方法名

`quitSmallVideo`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.quitSmallVideo(quitRes => {
     this.writeLog("退出小窗口:" + JSON.stringify(quitRes))
   });
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

是否静音播放

方法名

`setMute`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setMute({
    value: true
  }, res => {
    // console.log(res)
    this.writeLog(JSON.stringify(res))
  })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	Boolean	否	false	是否静音，true/false

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

是否静音播放

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

获取视频时长

方法名

`getDuration`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.getDuration(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "duration": 144410
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.duration	Integer	视频时长，单位：毫秒
code	Integer	返回类型，0.成功，其他：失败

获取当前播放的位置

方法名

`getCurrentPosition`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.getCurrentPosition(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "position": 95821
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.position	Integer	当前播放时长，单位：毫秒
code	Integer	返回类型，0.成功，其他：失败

获取当前缓冲百分比

方法名

`getBufferedPercentage`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.getBufferedPercentage(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "process": 100
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.process	Integer	当前缓存百分比
code	Integer	返回类型，0.成功，其他：失败

获取视频尺寸

方法名

`getSize`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.getSize(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "height": 592,
    "width": 368
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.width	Integer	视频宽度
data.height	Integer	视频高度

获取视频尺寸

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

旋转

方法名

`setRotation`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setRotation({
     rotation: 90
 },res => {
     // console.log(res)
     this.writeLog(JSON.stringify(res))
   })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
rotation	Float	是	无	旋转的角度

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

翻转

方法名

`setTransform`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setTransform({
   //翻转类型，0：正常，1.上下翻转，2.左右翻转
   transform: 1
 }, res => {
   // console.log(res)
   this.writeLog("旋转视频画面：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
transform	Integer	是	无	翻转类型，0：正常，1.上下翻转，2.左右翻转

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示

翻转

参数名	参数类型	参数描述
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

滤镜

说明

如果需要支持滤镜，必须设置视频播放模式为：GLSURFACE，否则滤镜无效果，目前插件支持20多种滤镜，同时也可以自定义滤镜

清除滤镜

方法名

`clearFilter`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.clearFilter(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

自动修正

方法名

`setFilterAutoFix`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterAutoFix({
   //修正值,0-1
   scale: 0.8
 }, res => {
   this.writeLog("自动修正：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
scale	Float	是	无	修正值,0-1

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

马赛克

方法名

`setFilterPixelation`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterPixelation({
   //马赛克大小，1-100
   pixel: 40
 }, res => {
   this.writeLog("马赛克：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
pixel	Integer	是	无	马赛克大小，1-100

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

黑白滤镜

方法名

`setFilterBlackAndWhite`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.setFilterBlackAndWhite(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

对比度

方法名

`setFilterContrast`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterContrast({
   //对比度值，0.1-2.0
   contrast: 0.8
 }, res => {
   this.writeLog("对比度：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
contrast	Float	是	无	对比度值，0.1-2.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

交叉线

方法名

`setFilterCrossProcess`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.setFilterCrossProcess(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

纪录片

方法名

`setFilterDocumentary`

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.setFilterDocumentary(res => {
         // console.log(res)
         this.writeLog(JSON.stringify(res))
       })
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

双色调

方法名

`setFilterDuotone`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterDuotone({
   firstColor: "#0000FF",
   secondColor: "#FFFF00"
 }, res => {
   this.writeLog("双色调：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
firstColor	String	是	无	第一种颜色
secondColor	String	是	无	第二种颜色

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

补光

方法名

`setFilterFillLight`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterFillLight({
   //光强度，0.0 - 1.0
   strength: 0.8
 }, res => {
   this.writeLog("补光：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
strength	Float	是	无	光强度，0.0 - 1.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

伽马线

方法名

`setFilterGamma`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterGamma({
   //值，0.0 - 2.0
   value: 0.8
 }, res => {
   this.writeLog("伽马线：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	Float	是	无	值，0.0 - 2.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

纹理

方法名

`setFilterGrain`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterGrain({
   //强度值，0.0 - 1.0
   strength: 0.8
 }, res => {
   this.writeLog("纹理：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
strength	Float	是	无	强度值，0.0 - 1.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象

参数名	参数类型	参数描述
code	Integer	返回类型，0.成功，其他：失败

灰度

方法名

`setFilterGrayscale`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterGrayscale(res => {
  this.writeLog("灰度：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

色度

方法名

`setFilterHue`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterHue({
   //值，0.0 - 1.0
   value: 0.8
 }, res => {
   this.writeLog("色度：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	Float	是	无	值，0.0 - 1.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

反色

方法名

`setFilterInvertColors`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterInvertColors(res => {
  this.writeLog("反色：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

简单风格

方法名

`setFilterLamoish`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterLamoish(res => {
  this.writeLog("简单风格：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

色调分离

方法名

`setFilterPosterize`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterPosterize(res => {
  this.writeLog("色调分离：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

barrel模糊

方法名

`setFilterBarrelBlur`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterBarrelBlur({
   //模糊数量值
   countLevel: 5
 }, res => {
   this.writeLog("Barrel类型模糊：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
countLevel	Integer	是	无	模糊数量值

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

饱和度

方法名

`setFilterSaturation`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterSaturation({
  //值,-1 - 1
  value: 0.8
}, res => {
  this.writeLog("饱和度：" + JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	值,-1 - 1

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

怀旧

方法名

`setFilterSepia`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterSepia(res => {
  this.writeLog("怀旧：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

温度

方法名

`setFilterTemperature`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterTemperature({
   //值，0 - 1
   value: 0.8
 }, res => {
   this.writeLog("温度：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	值，0 - 1

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

锐度

方法名

`setFilterSharpness`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterSharpness({
   //值，0 - 1
   value: 0.8
 }, res => {
   this.writeLog("锐度：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	值，0 - 1

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

色调

方法名

`setFilterTint`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterTint({
    //颜色值
    color: "#00FF00"
  }, res => {
    this.writeLog("色调：" + JSON.stringify(res))
  })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
color	String	是	无	颜色值

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

晕影

方法名

`setFilterVignette`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterVignette({
   //值 0 - 1.0
   value: 0.8
 }, res => {
   this.writeLog("晕影：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	值，0 - 1.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

重叠

方法名

`setFilterOverlay`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterOverlay({
   //值 0 - 1.0
   value: 0.0015
 }, res => {
   this.writeLog("重叠：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	值，0 - 1.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

简单模糊

方法名

`setFilterSampleBlur`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterSampleBlur({
    //值 1 - 10.0
    value: 4.0
}, res => {
    this.writeLog("简单模糊：" + JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	值，1 - 10.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

高斯模糊

方法名

`setFilterGaussianBlur`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterGaussianBlur({
   //值 1 - 10.0
   radius: 6.0,
   //类型，可选值：TYPEX，TYPEY，TYPEXY
   type: "TYPEXY"
 }, res => {
   this.writeLog("高斯模糊：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
radius	float	是	无	值，1 - 10.0
type	String	是	无	类型，可选值：TYPEX，TYPEY，TYPEXY

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

亮度

方法名

`setFilterBrightness`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setFilterBrightness({
   //值 0 - 1.0
   value: 0.8,
 }, res => {
   this.writeLog("亮度：" + JSON.stringify(res))
 })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	float	是	无	值，值 0 - 1.0

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

自定义滤镜

方法名

`customFilter`

用法

- 用法如下：

```
let shader = this.getCustomFilterShader();
this.$refs.refVideoPlayer.customFilter({
  //滤镜处理内容
  shader: shader,
}, res => {
  this.writeLog("自定义滤镜：" + JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
shader	String	是	无	滤镜内容

> getCustomFilterShader函数内容示例如下：

```
//自定义滤镜内容
  getCustomFilterShader() {
    //以反色为例
    let shader = "#extension GL_OES_EGL_image_external : require\n";
    shader += "precision mediump float;\n";
    shader += "varying vec2 vTextureCoord;\n";
    shader += "uniform samplerExternalOES sTexture;\n";
    shader += "void main() {\n";
    shader += "  vec4 color = texture2D(sTexture, vTextureCoord);\n";
    shader += "  float colorR = (1.0 - color.r) / 1.0;\n";
    shader += "  float colorG = (1.0 - color.g) / 1.0;\n";
    shader += "  float colorB = (1.0 - color.b) / 1.0;\n";
    shader += "  gl_FragColor = vec4(colorR, colorG, colorB, color.a);\n";
    shader += "}\n";
    return shader;
  }
```

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

开始生成gif

方法名

`startGif`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.startGif(res => {
  this.writeLog("开始生成gif：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "status": "start"
  },
  "message": "",
  "code": 0
}
{
  "data": {
    "status": "progress",
    "total": 81,
    "curPosition": 1
  },
  "message": "",
  "code": 0
}
{
  "data": {
    "status": "finish",
    "path": "/storage/emulated/0/Pictures/LevenPlayer/1739355354354.gif"
```

```
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.status	String	状态，start：开始生成gif，progress：gif处理进度，finish：生成完成
data.total	Integer	gif总帧数，progress状态下返回
data.curPosition	Integer	gif当前处理的帧数，progress状态下返回
data.path	String	gif文件保存路径，finish状态下返回
code	Integer	返回类型，0.成功，其他：失败

结束生成gif

方法名

`stopGif`

> 结束后生成的文件在startGif中返回

用法

- 用法如下：

```
this.$refs.refVideoPlayer.stopGif(res => {
  this.writeLog("结束生成gif：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

清除缓存

方法名

`clearCache`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.clearCache(res => {
  this.writeLog("清除缓存：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

获取所有音轨

方法名

`getAllMediaTrack`

> 目前音轨只支持EXO内核

用法

- 用法如下：

```
this.$refs.refVideoPlayer.getAllMediaTrack(res => {
  this.writeLog("获取所有音轨：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "list": [
      {
        "type": 1,
        "id": "1",
        "length": 1,
        "formats": [
          {
            "maxInputSize": 797,
            "sampleRate": 44100,
            "sampleMimeType": "audio/mp4a-latm",
            "bitrate": 129747,
            "language": "und",
            "codecs": "mp4a.40.2",
            "id": "2",
            "channelCount": 2
          }
```

```
        ]
      },
      {
        "type": 1,
        "id": "2",
        "length": 1,
        "formats": [
          {
            "maxInputSize": 815,
            "sampleRate": 44100,
            "sampleMimeType": "audio/mp4a-latm",
            "bitrate": 130036,
            "language": "und",
            "codecs": "mp4a.40.2",
            "id": "3",
            "channelCount": 2
          }
        ]
      }
    ]
  },
  "message": "",
  "code": 0
}
```


- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.list	Array[Object]	音轨列表
data.list.type	Integer	音轨类型
data.list.id	Integer	音轨id
data.list.length	Integer	音轨格式集合长度
data.list.formats	Array[Object]	音轨格式集合
data.list.formats.maxInputSize	Integer	最大输入大小
data.list.formats.sampleRate	Integer	采样率
data.list.formats.sampleMimeType	String	音频类型

参数名	参数类型	参数描述
data.list.formats.bitrate	Integer	比特率
data.list.formats.language	String	语言
data.list.formats.codecs	String	解码器格式
data.list.formats.id	Integer	id
data.list.formats.channelCount	Integer	通道数量
code	Integer	返回类型，0.成功，其他：失败

切换音轨

方法名

`changeMediaTrack`

> 目前音轨只支持EXO内核

用法

- 用法如下：

```
    this.$refs.refVideoPlayer.changeMediaTrack({
     //音轨的id
     id: 1
   }, res => {
     this.writeLog("切换音轨：" + JSON.stringify(res))
   })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
id	Integer	是	无	音轨的id，可以从获取所有音轨中取到

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

开启弹幕

方法名

`danmakuShow`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.danmakuShow(res => {
this.writeLog("开启弹幕：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

关闭弹幕

方法名

`danmakuHide`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.danmakuHide(res => {
    this.writeLog("关闭弹幕：" + JSON.stringify(res))
})
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

添加弹幕

方法名

`addDanmaku`

用法

- 用法如下：

```
this.$refs.refVideoPlayer.addDanmaku({
    //文本
    text: "这是一条测试弹幕 ",
    //内边距
    padding: 5,
    //弹幕优先级,0为低优先级,>0为高优先级不会被过滤器过滤
    priority: 8,
    //是否是直播弹幕
    isLive: true,
    //弹幕出现时间，弹幕当前时间延迟时间，单位：毫秒
    time: 500,
    //字体大小
    textSize: 45,
    //文本颜色
    textColor: "#FF0000",
    //阴影/描边颜色
    textShadowColor: "#FFFFFF",
    //边框的颜色
    borderColor: "#00FF00"
  }, res => {
    this.writeLog("添加弹幕：" + JSON.stringify(res))
  })
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
text	String	是	无	弹幕内容
padding	Integer	否	0	内边距

参数名	参数类型	是否必填	默认值	参数描述
priority	Integer	否	0	弹幕优先级,0为低优先级,>0为高优先级不会被过滤器过滤
isLive	Boolean	否	false	是否是直播弹幕
time	Integer	是	无	弹幕出现时间，弹幕当前时间延迟时间，单位：毫秒
textSize	Integer	否	无	字体大小
textColor	String	否	无	文本颜色
textShadowColor	String	否	无	阴影/描边颜色
borderColor	String	否	无	边框的颜色

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

组件事件

播放进度

事件名称

`onProgress`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config='
</leven-uts-vlcPlayer>
```

回调示例

```
onProgress(e) {
    this.writeLog("onProgress:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "progress": 3,
  "currentPosition": 5163,
  "secProgress": 0,
  "duration": 142656
}
```

回调说明

参数名	参数类型	参数描述
progress	Integer	播放进度百分比
currentPosition	Integer	当前播放的时长，单位：毫秒
duration	Integer	视频总时长，单位：毫秒

播放进度

参数名	参数类型	参数描述
secProgress	Integer	当前内存缓冲进度

系统错误

事件名称

`onError`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onError(e) {
    this.writeLog("onProgress:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
      "data": {},
      "message": "系统错误",
      "code": -1
    }
```

回调说明

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	错误值

开始加载

事件名称

`onStartPrepared`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config='
</leven-uts-vlcPlayer>
```

回调示例

```
onStartPrepared(e) {
    this.writeLog("onStartPrepared:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

加载成功

事件名称

`onPrepared`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onPrepared(e) {
    this.writeLog("onPrepared:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

点击了开始按键播放

事件名称

`onClickStartIcon`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onClickStartIcon(e) {
    this.writeLog("onClickStartIcon:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

点击了错误状态下的开始按键

事件名称

`onClickStartError`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onClickStartError(e) {
    this.writeLog("onClickStartError:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

点击了停止

事件名称

`onClickStop`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onClickStop(e) {
    this.writeLog("onClickStop:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

点击了重新播放

事件名称

`onClickResume`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onClickResume(e) {
    this.writeLog("onClickResume:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

正常播放完成

事件名称

`onAutoComplete`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onAutoComplete(e) {
    this.writeLog("onAutoComplete:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

非正常播放完成

事件名称

`onComplete`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config='
</leven-uts-vlcPlayer>
```

回调示例

```
onComplete(e) {
    this.writeLog("onComplete:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

进入全屏

事件名称

`onEnterFullscreen`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onEnterFullscreen(e) {
    this.writeLog("onEnterFullscreen:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

退出全屏

事件名称

`onQuitFullscreen`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config='
</leven-uts-vlcPlayer>
```

回调示例

```
onQuitFullscreen(e) {
    this.writeLog("onQuitFullscreen:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

进入小窗口

事件名称

`onEnterSmallWidget`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onEnterSmallWidget(e) {
    this.writeLog("onEnterSmallWidget:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

退出小窗口

事件名称

`onQuitSmallWidget`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onQuitSmallWidget(e) {
    this.writeLog("onQuitSmallWidget:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

触摸调整声音

事件名称

`onTouchScreenSeekVolume`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onTouchScreenSeekVolume(e) {
    this.writeLog("onTouchScreenSeekVolume:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

触摸调整进度

事件名称

`onTouchScreenSeekPosition`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onTouchScreenSeekPosition(e) {
    this.writeLog("onTouchScreenSeekPosition:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

触摸调整亮度

事件名称

`onTouchScreenSeekLight`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config='
</leven-uts-vlcPlayer>
```

回调示例

```
onTouchScreenSeekLight(e) {
    this.writeLog("onTouchScreenSeekLight:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

播放错误

事件名称

`onPlayError`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onPlayError(e) {
    this.writeLog("onPlayError:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

点击了空白区域开始播放

事件名称

`onClickStartThumb`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="
</leven-uts-vlcPlayer>
```

回调示例

```
onClickStartThumb(e) {
    this.writeLog("onClickStartThumb:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

点击了播放中的空白区域

事件名称

`onClickBlank`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config='
</leven-uts-vlcPlayer>
```

回调示例

```
onClickBlank(e) {
    this.writeLog("onClickBlank:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题

点击了返回按钮

事件名称

`onBackButtonClick`

用法

```
<leven-uts-vlcPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config='
</leven-uts-vlcPlayer>
```

回调示例

```
onBackButtonClick(e) {
    this.writeLog("onBackButtonClick:" + JSON.stringify(e.detail))
}
```

回调内容

```
{
  "url": "https://aliyuncdnsaascloud.xjhktv.com/video/A%20Lin%2B%E5%80%AA%E5%AD%90
  "title": "安卓高质量视频播放器原生插件"
}
```

回调说明

参数名	参数类型	参数描述
url	String	播放地址
title	String	标题