

uniapp 安卓视频播放器原生插件

目 录

使用说明	1
使用方法	1.1
更新日志	1.2
插件方法	2
开始播放	2.1
暂停播放	2.2
继续播放	2.3
停止播放	2.4
视频截图	2.5
是否播放中	2.6
设置播放位置	2.7
全屏播放	2.8
退出全屏	2.9
小窗口播放	2.10
关闭小窗口	2.11
设置播放比例 v1.1.0	2.12
倍速播放 v1.1.0	2.13
设置是否静音播放 v1.1.0	2.14
获取视频时长 v1.1.0	2.15
获取当前播放的位置 v1.1.0	2.16
获取当前缓冲百分比 v1.1.0	2.17
是否处于静音状态 v1.1.0	2.18
获取视频的尺寸 v1.1.0	2.19
旋转视频画面 v1.1.0	2.20
设置小屏的尺寸 v1.1.0	2.21
插件属性	3
属性配置	3.1
插件事件	4
播放状态监听	4.1
播放器状态监听	4.2
系统错误	4.3

使用方法

介绍

安卓视频播放器原生插件，支持自定义播放器内核(IJKPlayer,EXOPlayer2,系统内核)，支持直播拉流，rtmp，rtsp，m3u8等流文件播放，支持协议h263\4\5、Https、concat、rtsp、hls、rtmp、crypto、mpeg等等，全屏和小窗口播放等

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=17013>

权限

- android.permission.INTERNET
- android.permission.READ_EXTERNAL_STORAGE
- android.permission.WRITE_EXTERNAL_STORAGE

示例文件

演示程序

使用方法

百度网盘

下载地址：https://pan.baidu.com/s/14kftMGgGcteis5OQ32_nHA

提取码：zmb3

用法

在需要使用插件的页面加载以下代码

```
<leven-videoPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="config"
  @onPlayerStatus="onPlayerStatus" @onError="onError">
</leven-videoPlayer>
```

插件仅支持`nvue`页面

页面内容

```
<template>
  <view>
    <uni-card title="安卓视频播放器原生插件">
      <leven-videoPlayer ref="refVideoPlayer" style="flex:1; height: 300px;" :config="config"
        @onPlayerStatus="onPlayerStatus" @onError="onError">
      </leven-videoPlayer>
      <uni-section title="内核">
        <uni-data-checkbox v-model="config.kernel" :localdata="kernelType"></uni-data-checkbox>
      </uni-section>
      <uni-section title="播放地址">
        <uni-easyinput v-model="config.url" :maxlength="-1" placeholder="请输入播放地址">
        </uni-easyinput>
      </uni-section>
      <uni-section title="是否是直播">
        <uni-data-checkbox v-model="config.isLive" :localdata="isLiveType"></uni-data-checkbox>
      </uni-section>
      <button type="primary" @click="play">开始播放</button>
      <button type="primary" @click="pause">暂停播放</button>
      <button type="primary" @click="resume">继续播放</button>
      <button type="primary" @click="stop">停止播放</button>
      <button type="primary" @click="shotPic">视频截图</button>
      <button type="primary" @click="isPlaying">是否播放中</button>
      <button type="primary" @click="setPlayPosition">设置播放位置</button>
      <button type="primary" @click="fullScreen">全屏播放</button>
      <button type="primary" @click="smallVideo">小窗口播放</button>
      <button type="primary" @click="setScreenScaleType">播放比例</button>
      <button type="primary" @click="setSpeed">倍速播放</button>
      <button type="primary" @click="setMute">设置是否静音播放</button>
```

```

<button type="primary" @click="getDuration">获取视频时长</button>
<button type="primary" @click="getCurrentPosition">获取当前播放的位置</button>
<button type="primary" @click="getBufferedPercentage">获取当前缓冲百分比</button>
<button type="primary" @click="isMute">是否处于静音状态</button>
<button type="primary" @click="getSize">获取视频的尺寸</button>
<button type="primary" @click="setRotation">旋转视频画面</button>
<button type="primary" @click="setSmallVideoSize">设置小屏尺寸</button>
<button type="primary" @click="logStr = ''">清空日志</button>
</uni-card>
<uni-card class="uni-card-box" title="日志">
  <view><text style="font-size: 14px; flex-wrap: wrap;">{{logStr}}</text></view>
</uni-card>

<!-- 弹窗 -->
<uni-popup ref="refSelectSizePop">
  <view style="width: 300px;">
    <uni-card title="请选择播放比例">
      <view class="view-button" v-for="(item, index) in sizeList" :key="index"
        class="button-text">{{item.text}}</text></view>
    </uni-card>
  </view>
</uni-popup>

<!-- 弹窗 -->
<uni-popup ref="refSpeedPop">
  <view style="width: 300px;">
    <uni-card title="请选择播放倍速">
      <view class="view-button" v-for="(item, index) in speedList" :key="index"
        <text class="button-text">{{item.text}}</text>
      </view>
    </uni-card>
  </view>
</uni-popup>
</view>
</template>

<script>
export default {
  data() {
    return {
      logStr: "",
      //配置
      config: {
        //播放地址, rtmp://liteavapp.qcloud.com/live/liteavdemoplayerstreamid
        //http://devimages.apple.com.edgekey.net/streaming/examples/bipbop_4x3/
        //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-bCJbio780yeyu41:
        // url: "/assets/base.mp4",
        //如果播放项目里面的文件, 比如static下面的文件, url设置为"/static/***.mp4"

```

```

//如果播放assets下面的文件url请设置为："/assets/***.mp4"
url: "/static/base.mp4",
//播放器预览背景图片，支持网络地址
backgroundImage: "http://www.yeyuboke.com/svg/image_default.jpg",
//视频标题
title: "安卓高质量视频播放器原生插件",
//默认播放内核
kernel: 3,
//是否循环播放
loop: true,
//开始播放位置，单位：毫秒
startPosition: 0,
//是否是直播
isLive: false,
//是否静音播放
isMute: false,
//小屏的尺寸
smallVideoSize: [300, 300],
//是否根据屏幕方向自动进入全屏
enableOrientation: false,
//是否显示底部控制条
bottomProgress: true,
//是否开启双击视频暂停和播放，默认：true
doubleTapTogglePlayEnabled: true,
//是否展示组件ui，如组件背景图片，loading等，默认：true
showControllerUi: true
},
// 内核
kernelType: [{
    text: "IJK内核",
    value: 1
}, {
    text: "EX0内核",
    value: 2
}, {
    text: "系统内核",
    value: 3
}],
//是否是直播
isLiveType: [{
    text: "是",
    value: true
}, {
    text: "否",
    value: false
}],
// 播放比例列表
sizeList: [{
    text: "默认",
    value: "scale_default"
}

```

```

    }, {
        text: "16:9",
        value: "scale_169"
    }, {
        text: "4:3",
        value: "scale_43"
    }, {
        text: "原始大小",
        value: "scale_original"
    }, {
        text: "填充",
        value: "scale_match"
    }, {
        text: "居中裁剪",
        value: "scale_center_crop"
    }],
    //倍速播放
    speedList: [{
        text: "0.5倍",
        value: 0.5
    }, {
        text: "0.75倍",
        value: 0.75
    }, {
        text: "正常倍速",
        value: 1
    }, {
        text: "1.5倍",
        value: 1.5
    }, {
        text: "2倍",
        value: 2
    }
    ]
},
methods: {
    // 开始播放
    play() {
        if (this.$refs.refVideoPlayer) {
            this.$refs.refVideoPlayer.play(this.config, res => {
                this.writeLog("开始播放:" + JSON.stringify(res))
            });
        }
    },
    // 暂停播放
    pause() {
        if (this.$refs.refVideoPlayer) {
            this.$refs.refVideoPlayer.pause(res => {
                this.writeLog("暂停播放:" + JSON.stringify(res))
            });
        }
    }
}

```

```

    }
  },
  // 继续播放
  resume() {
    if (this.$refs.refVideoPlayer) {
      this.$refs.refVideoPlayer.resume(res => {
        this.writeLog("继续播放:" + JSON.stringify(res))
      });
    }
  },
  // 停止播放
  stop() {
    if (this.$refs.refVideoPlayer) {
      this.$refs.refVideoPlayer.stop(res => {
        this.writeLog("停止播放:" + JSON.stringify(res))
      });
    }
  },
  // 视频截图
  shotPic() {
    if (this.$refs.refVideoPlayer) {
      this.$refs.refVideoPlayer.shotPic(res => {
        this.writeLog("视频截图:" + JSON.stringify(res))
      });
    }
  },
  // 是否播放中
  isPlaying() {
    if (this.$refs.refVideoPlayer) {
      this.$refs.refVideoPlayer.isPlaying(res => {
        this.writeLog("是否播放中:" + JSON.stringify(res))
      });
    }
  },
  // 设置播放位置
  setPlayPosition() {
    if (this.$refs.refVideoPlayer) {
      this.$refs.refVideoPlayer.setPlayPosition({
        //播放位置, 单位: 毫秒
        position: 30000
      }, res => {
        this.writeLog("设置播放位置:" + JSON.stringify(res))
      });
    }
  },
  // 全屏播放
  fullScreen() {
    if (this.$refs.refVideoPlayer) {
      this.$refs.refVideoPlayer.fullScreen(res => {
        this.writeLog("全屏播放:" + JSON.stringify(res))
      });
    }
  }
}

```

```

        //10秒后退出全屏
        setTimeout(() => {
            this.$refs.refVideoPlayer.quitFullScreen(quitRes => {
                this.writeLog("退出全屏:" + JSON.stringify(quitRes))
            });
        }, 10000)
    });
}
},
// 小窗口播放
smallVideo() {
    if (this.$refs.refVideoPlayer) {
        this.$refs.refVideoPlayer.smallVideo(res => {
            this.writeLog("小窗口播放:" + JSON.stringify(res))

            //10秒后退出小窗口
            setTimeout(() => {
                this.$refs.refVideoPlayer.quitSmallVideo(quitRes => {
                    this.writeLog("退出小窗口:" + JSON.stringify(quitRes))
                });
            }, 10000)
        });
    }
},
//设置播放比例
setScreenScaleType() {
    if (this.$refs.refSelectSizePop) {
        this.$refs.refSelectSizePop.open();
    }
},
//播放比例的选择
selectSizeItem(item) {
    console.log(item)
    if (this.$refs.refVideoPlayer) {
        this.$refs.refVideoPlayer.setScreenScaleType({
            value: item.value
        }, res => {
            // console.log(res)
            this.writeLog(JSON.stringify(res))
            if (this.$refs.refSelectSizePop) {
                this.$refs.refSelectSizePop.close();
            }
        })
    }
},
//倍速播放
setSpeed() {
    if (this.$refs.refSpeedPop) {
        this.$refs.refSpeedPop.open();
    }
}

```

```

},
//倍速播放的选择
selectSpeedItem(item) {
  if (this.$refs.refVideoPlayer) {
    this.$refs.refVideoPlayer.setSpeed({
      value: item.value
    }, res => {
      // console.log(res)
      this.writeLog(JSON.stringify(res))
      if (this.$refs.refSpeedPop) {
        this.$refs.refSpeedPop.close();
      }
    })
  }
},
//设置是否静音播放
setMute() {
  this.$refs.refVideoPlayer.setMute({
    value: !this.config.isMute
  }, res => {
    // console.log(res)
    this.writeLog(JSON.stringify(res))
    this.config.isMute = !this.config.isMute
  })
},
//获取视频时长
getDuration() {
  this.$refs.refVideoPlayer.getDuration(res => {
    // console.log(res)
    this.writeLog("获取视频时长：" + JSON.stringify(res))
  })
},
//获取当前播放的位置
getCurrentPosition() {
  this.$refs.refVideoPlayer.getCurrentPosition(res => {
    // console.log(res)
    this.writeLog("获取当前播放的位置：" + JSON.stringify(res))
  })
},
//获取当前缓冲百分比
getBufferedPercentage() {
  this.$refs.refVideoPlayer.getBufferedPercentage(res => {
    // console.log(res)
    this.writeLog("获取当前缓冲百分比：" + JSON.stringify(res))
  })
},
//是否处于静音状态
isMute() {
  this.$refs.refVideoPlayer.isMute(res => {
    // console.log(res)
  })
}

```

```

        this.writeLog("是否处于静音状态：" + JSON.stringify(res))
    })
},
//获取视频的尺寸
getSize() {
    this.$refs.refVideoPlayer.getSize(res => {
        // console.log(res)
        this.writeLog("获取视频的尺寸：" + JSON.stringify(res))
    })
},
//旋转视频画面
setRotation() {
    this.$refs.refVideoPlayer.setRotation({
        rotation: 90
    }, res => {
        // console.log(res)
        this.writeLog("旋转视频画面：" + JSON.stringify(res))
    })
},
//设置小屏的尺寸
setSmallVideoSize() {
    this.$refs.refVideoPlayer.setSmallVideoSize({
        size: [500, 500]
    }, res => {
        // console.log(res)
        this.writeLog("设置小屏的尺寸：" + JSON.stringify(res))
    })
},
//播放状态回调
onVideoStatus(e) {
    this.writeLog("onVideoStatus:" + JSON.stringify(e.detail))
},
//播放器状态回调
onPlayerStatus(e) {
    this.writeLog("onPlayerStatus:" + JSON.stringify(e.detail))
},
// 系统错误
onError(e) {
    this.writeLog("onError:" + JSON.stringify(e))
},
// 写日志
writeLog(str) {
    console.log(str);
    let logStr = uni.$lv.date.format(null, "yyyy-mm-dd hh:MM:ss") + " " + str
    this.logStr = logStr + this.logStr;
}
}
}
</script>

```

使用方法

```
<style>
  .view-button {
    background-color: #007aff;
    border-radius: 5px;
    height: 50px;
    justify-content: center;
    align-items: center;
    margin-bottom: 5px;
  }

  .button-text {
    color: #FFFFFF;
  }
</style>
```

更新日志

2024-07-19 v1.1.2

- [优化] 优化播放状态监听，修复每次播放监听重复的bug
- [新增] 新增可以播放项目中static目录下的文件和assets下的文件，播放assets下的文件请参考插件示例文件
- [新增] 新增播放配置[是否开启双击视频暂停和播放](#) `doubleTapTogglePlayEnabled`
- [新增] 新增播放配置[是否展示组件ui](#)，如组件背景图片，loading等 `showControllerUi`

2024-04-07 v1.1.1

- [优化] 视频属性新增 [是否根据屏幕方向自动进入全屏](#)
- [优化] 视频属性新增 [是否显示底部控制条](#)

2024-03-19 v1.1.0

- [新增] 新增接口 [【设置播放比例】](#)
- [新增] 新增接口 [【倍速播放】](#)
- [新增] 新增接口 [【设置是否静音播放】](#)
- [新增] 新增接口 [【获取视频时长】](#)
- [新增] 新增接口 [【获取当前播放位置】](#)
- [新增] 新增接口 [【获取当前缓冲百分比】](#)
- [新增] 新增接口 [【是否处于静音状态】](#)
- [新增] 新增接口 [【获取视频的尺寸】](#)
- [新增] 新增接口 [【旋转视频画面】](#)
- [新增] 新增接口 [【设置小屏尺寸】](#)

2024-03-14

首次发布

开始播放

方法名

play

用法

- 用法如下：

```
this.$refs.refVideoPlayer.play({
  //播放地址
  url: "http://alvideo.ippzone.com/zyvd/98/90/b753-55fe-11e9-b0d8-00163e0c",
  //播放器预览背景图片，支持网络地址
  backgroundImage: "http://www.yeyuboke.com/svgas/image_default.jpg",
  //视频标题
  title: "安卓视频播放器原生插件",
  //默认播放内核，1.IJK内核（默认），2.EXO内核，3.系统内核
  kernel: 1,
  //全屏时是否横屏显示
  fullIsLand: true,
  //是否循环播放
  loop: true,
  //开始播放位置，单位：毫秒
  startPosition: 0,
  //是否是直播
  isLive: false,
  //是否静音播放
  isMute: false,
  //小屏的尺寸
  smallVideoSize: [300, 300]
}, 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	播放地址，本地或网络地址
backgroundImage	String	否	无	视频背景图片，本地或网络地址

开始播放

参数名	参数类型	是否必填	默认值	参数描述
title	String	否	无	视频标题
kernel	Integer	否	1	播放器内核， 1.IJK内核（默认），2.EXO内核，3.系统内核
fullIsLand	Boolean	否	false	全屏时是否横屏显示
loop	Boolean	否	false	是否循环播放
startPosition	Integer	否	0	开始播放的位置，单位：毫秒
isLive	Boolean	否	false	是否是直播
isMute	Boolean	否	false	是否静音播放
smallVideoSize	Array	否	无	小屏的尺寸,[宽，高]

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

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

视频截图

方法名

shotPic

用法

- 用法如下：

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

- 参数说明

无

回调

- 示例

```
{
  "data": {
    "path": "/storage/emulated/0/Android/data/test.leven.uniplugin.com/images/"
  },
  "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	截图路径
code	Integer	返回类型，0.成功，其他：失败

是否播放中

方法名

isPlaying

用法

- 用法如下：

```
this.$refs.refVideoPlayer.isPlaying(res => {
  this.writeLog("是否播放中:" + JSON.stringify(res))
});
```

- 参数说明

无

回调

- 示例

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

- 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	true:正在播放，false：未播放
code	Integer	返回类型，0.成功，其他：失败

设置播放位置

方法名

setPosition

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setPosition({
  //播放位置，单位：毫秒
  position: 30000
}, res => {
  this.writeLog("设置播放位置:" + 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	否	播放位置，单位：毫秒

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

设置播放位置

参数名	参数类型	参数描述
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

- 用法如下：

- 参数说明

无

- 示例

- 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

设置播放比例 v1.1.0

方法名

setScreenScaleType

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setScreenScaleType({
  value: "scale_169"
}, 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：

scale_default：默认尺寸

scale_169：16:9

scale_43：4:3

scale_original：原始大小

scale_match：填充整个父级

scale_center_crop：居中裁剪

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

倍速播放 v1.1.0

- 用法如下：

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
value	Float	否	1	倍速的值

- 示例

- 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

设置是否静音播放 v1.1.0

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

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 v1.1.0

- 用法如下：

- 参数说明

无

- 示例

- 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 v1.1.0

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 v1.1.0

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

是否处于静音状态 v1.1.0

方法名

isMute

用法

- 用法如下：

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

- 参数说明

无

回调

- 示例

```
{  
  "data": {  
    "isMute": false  
  },  
  "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.isMute	Boolean	是否静音
code	Integer	返回类型，0.成功，其他：失败

获取视频的尺寸 v1.1.0

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

获取视频的尺寸 v1.1.0

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

旋转视频画面 v1.1.0

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

设置小屏的尺寸 v1.1.0

方法名

setSmallVideoSize

用法

- 用法如下：

```
this.$refs.refVideoPlayer.setSmallVideoSize({
  size: [500, 500]
}, 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
size	Array	是	无	尺寸，[宽，高]

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

属性配置

属性配置 config

- 示例

```
//配置
config: {
  //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://alvideo.ippzone.com/zyvd/98/90/b753-55fe-11e9-b0d8-00163e0c
  //播放器预览背景图片，支持网络地址
  backgroundImage: "http://www.yeyuboke.com/svgas/image_default.jpg",
  //视频标题
  title: "安卓高质量视频播放器原生插件",
  //默认播放内核
  kernel: 3,
  //是否循环播放
  loop: true,
  //开始播放位置，单位：毫秒
  startPosition: 0,
  //是否是直播
  isLive: false,
  //是否静音播放
  isMute: false,
  //小屏的尺寸
  smallVideoSize: [300, 300],
  //是否根据屏幕方向自动进入全屏
  enableOrientation: false,
  //是否显示底部控制条
  bottomProgress: false,
  //是否开启双击视频暂停和播放，默认：true
  doubleTapTogglePlayEnabled: true,
  //是否展示组件ui，如组件背景图片，loading等，默认：true
  showControllerUi: true
},
```

- 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	播放地址，本地或网络地址

参数名	参数类型	是否必填	默认值	参数描述
backgroundImage	String	否	无	视频背景图片，本地或网络地址
title	String	否	无	视频标题
kernel	Integer	否	1	播放器内核，1.IJK内核（默认），2.EXO内核，3.系统内核
fullIsLand	Boolean	否	false	全屏时是否横屏显示
loop	Boolean	否	false	是否循环播放
startPosition	Integer	否	0	开始播放的位置，单位：毫秒
isLive	Boolean	否	false	是否是直播
isMute	Boolean	否	false	是否静音播放
smallVideoSize	Array	否	无	尺寸，[宽，高]
enableOrientation	Boolean	否	false	是否根据屏幕方向自动进入全屏
bottomProgress	Boolean	否	true	是否显示底部控制条
doubleTapTogglePlayEnabled v1.1.2	Boolean	否	true	是否开启双击视频暂停和播放
showControllerUi v1.1.2	Boolean	否	true	是否展示组件ui，如组件背景图片，loading等

url设置为assets下的文件具体方法为：

在项目根目录下依次创建【nativeplugins】>【leven-videoPlayer】>【android】>【assets】

把需要播放的视频文件放到assets目录下即可，url的配置路径可填写：/assets/***.mp4

播放状态监听

方法名

onVideoStatus

用法

- 用法如下：

```
//播放状态回调
onVideoStatus(e) {
  this.writeLog("onVideoStatus:" + JSON.stringify(e.detail))
},
```

- 参数说明

无

回调

- 示例

```
{
  "status": 1
}
```

- 回调说明：

参数名	参数类型	参数描述
status	Integer	当前播放的状态

当前播放的状态说明：

- 0：未开始播放
- 1：准备中
- 2：准备完成
- 3：播放中
- 4：暂停
- 5：播放完成
- 6：开始缓冲

播放状态监听

7：缓冲结束

8：开始播放中止

播放器状态监听

方法名

onPlayerStatus

用法

- 用法如下：

```
//播放状态的变化
onPlayerStatus(e) {
  this.writeLog("onPlayerStatus:" + JSON.stringify(e.detail))
},
```

- 参数说明

无

回调

- 示例

```
{
  "status": 2
}
```

- 回调说明：

参数名	参数类型	参数描述
status	Integer	播放状态，请参考下方说明

- 10：普通播放器
- 11：全屏播放器
- 12：小屏播放器

系统错误

- 用法如下：

- 参数说明

无

- 示例

暂无

- 回调说明：

无