

Project

视频点播及下载系统

特点

- 接收遵循H.264协议的流
- HTML嵌入
- Javascript wrapper
- 基于HTML5解码，不引入额外的插件
- 灵活调用

依赖项

本播放器基于HTML5技术构建，所以需要您的浏览器支持HTML5解析

建议您使用IE9及其以上浏览器，或者使用Google Chrome等对HTML5有着良好支持的浏览器

HTML嵌入

你需要在HTML的head标签里面嵌入对video-js.css的引用

```
<!--video css-->
<link href="video/video-js.css" rel="stylesheet">
```

为了更加快速的加载页面内容，我们建议将js的引用放置在文档底部

```
<!--For jquery -->
<script src="js/jquery.min.js" charset="UTF-8"></script>
<!--For video core js-->
<script src="video/video.js"></script>
<!--For video player core logical control -->
<script src="js/core.js"></script>
```

然后你需要在body标签内放置一个容器，并有一个id标示，如下：

```
<div class="playerbody center-block">
  <video id="vjsplayer" class="video-js vjs-default-skin vjs-big-play-centered"
```

```

        controls preload="auto" width="800" height="400"
        data-setup='{}'
        onended="core.playEnded()"
        onwaiting="core.playWaiting()"
        onplaying="core.playPlaying()"
        onpause="core.playPaused()">
<source src="" type='video/mp4' />
</video>
</div>

```

然后你需要做一些初始化工作，即ready函数：

```

<!--For document ready-->
<script type="text/javascript">
    //Initially,it's empty
    var playlist = null;
    $(document).ready(function(){
        //initial something...
        //initial something...
        core.getPlayerId("vjsplayer");
        core.getStatusId("statusImg","statusStr");
        core.getDatetimepickerId("startYYYYMMDDHHII","startSec","endYYYYMMDDHHII","endSec");
        core.getDeviceId("deviceId");
        core.getPlaylistviewId("playlistview");
    });
</script>

```

datetimepicker控件样式的定义

```

<!--For define datatime picker style -->
<script type="text/javascript">
    $(".form_datetime").datetimepicker({
        format: "yyyymmddhhii",
        language:"zh-CN",
        autoclose: true,
        todayBtn: true,
        todayHighlight: 1,
        pickerPosition: "bottom-left",
        startDate: "2013-12-01",
        pickDate: true,
        pickTime: true,
        hourStep: 1,
        minuteStep: 1,
        secondStep: 1,
        inputMask: true
    });
</script>

```

测试页面框架

Bootstrap是一个简洁、直观、强悍、移动设备优先的前端开发框架，可以让web开发更迅速、简单。因此我们选择了基于Bootstrap 3.03开发测试页面,更多关于Bootstrap的技术请移步Bootstrap中文站点

<http://v3.bootcss.com/>

你需要在HTML的head标签里面嵌入对bootstrap的样式文件的引用

```
<!-- Bootstrap core CSS -->
<link href="css/bootstrap.min.css" rel="stylesheet">
<!-- Bootstrap theme -->
<link href="css/bootstrap-theme.min.css" rel="stylesheet">
<!-- Bootstrap datetimepicker -->
<link href="css/bootstrap-datetimepicker.min.css" rel="stylesheet">
<!-- template -->
<link href="css/theme.css" rel="stylesheet">
```

然后你需要在body的底部添加对js的引用

```
<!--For jquery -->
<script src="js/jquery.min.js"></script>
<script src="js/jquery-1.8.3.min.js"></script>
<!--For bootstrap -->
<script src="js/bootstrap.min.js"></script>
<script src="js/holder.min.js"></script>
```

测试页面的时间选择控件使用了bootstrap datetimepicker，为此，需要在head标签添加对Bootstrap datetimepicker的css引用

```
<!-- Bootstrap datetimepicker -->
<link href="css/bootstrap-datetimepicker.min.css" rel="stylesheet">
```

添加完css引用后需要引用js

```
<!--For bootstrap datetime picker -->
<script src="js/bootstrap-datetimepicker.min.js"></script>
<script src="js/locales/bootstrap-datetimepicker.zh-CN.js"></script>
```

接下来你需要对日期时间控件进行一些定制，以满足要求

```
<!--For define datetime picker style -->
<script type="text/javascript">
$(".form_datetime").datetimepicker({
    format: "yyymmddhhii",
    language: "zh-CN",
```

```
    autoclose: true,  
    todayBtn: true,  
    todayHighlight: 1,  
    pickerPosition: "bottom-left",  
    startDate: "2013-12-01",  
    pickDate: true,  
    pickTime: true,  
    hourStep: 1,  
    minuteStep: 1,  
    secondStep: 1,  
    inputMask: true  
});  
</script>
```

Video方法

videojs()

- 说明

HTML5 video的javascript包装

- 参数表—id

id-播放器video标签的标示符

- 调用方法

```
var player = videojs('video_id');
```

play()

- 说明

对媒体进行播放

- 参数表—null

- 调用方法

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

pause()

- 说明

对媒体进行暂停操作

- 参数表—null
- 调用方法

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

currentTime()

- 说明

设置新的播放点

获取当前的播放时间点

- 参数表—timeInt or null

timeInt-设置当前的播放时间（秒）

无参数获取当前时间点

- 调用方法

```
player.currentTime(120);  
var currentTimepoint = player.currentTime();
```

src(url)

- 说明

更新url,更新完需要调用play()方法执行

- 参数表—url

url-设置新的播放url

- 调用方法

```
player.src("http://yourserver/file.mp4");  
player.src({ type: "video/mp4", src: "http://yourserver/video.mp4" });
```

duration()

- 说明

获取当前媒体的时长

- 参数表—null
- 调用方法

```
player.duration();
```

requestFullScreen()

- 说明

请求全屏动作

- 参数表—null
- 调用方法

```
player.requestFullScreen();
```

core(核心逻辑)方法

playEnded()

- 说明

HTML5 video的监听事件，实现媒体结束事件的处理

类似的方法playPaused(),playWaiting(),playPlaying()不再列举

- 参数表—null
- 调用方法

```
core.playEnded();  
core.playPaused();  
core.playWaiting();  
core.playPlaying();
```

getPlayerId()

- 说明

初始化

- 参数表—playerId

playerId-初始化工作，保存当前的播放器标示符

- 调用方法

```
//initial something...
core.getPlayerId("vjsplayer");
```

- 类似的方法还有getStatusId(), getDateTimepickerId(), getDeviceId().调用方法如下

```
core.getStatusId("statusImg","statusStr");
core.getDateTimepickerId("startYYYYMMDDHHII","startSec","endYYYYMMDDHHII","endSec");
core.getDeviceId("deviceId");
core.getPlaylistviewId("playlistview");
```

playFromList()

- 说明

执行play动作

- 参数表—id

播放列表的id值

- 调用方法

```
core.playFromList(id);
```

getDatetimepickerStr()

- 说明

获取用户的请求时间戳信息，并进行合法性检测。该函数会调用一些子方法，不再细述。

- 参数表—startYYYYMMDDHHIIId,startSecId,endYYYYMMDDHHIIId,endSecId,deviceId

- startYYYYMMDDHHIIId——起始的日历控件id
- startSecId——起始秒钟输入框id
- endYYYYMMDDHHIIId——结束的日历控件id
- endSecId——结束秒钟输入框id
- deviceId——设备卡口号

- 调用方法

```
core.getDatetimepickerStr(startYYYYMMDDHHIIId,startSecId,endYYYYMMDDHHIIId,endSecId,dev:
```

generateRowCell()

- 说明

动态生成列表

- 参数表—url,id

- 视频的url值
- 播放列表的id值

- 调用方法

```
var playlistStr = core.generateRowCell(url,index);
```

formatStr()

- 说明

格式化日期字符串

- 参数表—YYYYMMDDHHIISS格式的字符串

正则后的文件名

- 调用方法

```
var listStr = this.formatStr(filename);
```


timeDiff()

- 说明

计算两个格式化时间戳的差值

- 参数表—timeStamp1,timeStamp2

timeStamp1——开始时间

timeStamp2——结束时间

- 返回值

该方法返回两者的差值，以秒为单位

- 调用方法 `var startAt = this.timeDiff(fileTimeStr,this.requestedStartTimeStr);`

getResponseFilenameFromUrl()

- 说明

正则表达式，从返回的URL里面提取文件名

- 参数表—url

url——服务器返回的url字符串

- 返回值

该方法返回文件名

- 调用方法 `var filename = getResponseFilenameFromUrl(url);`

queryList()

- 说明

查询按钮的实际响应函数，该函数是一个入口函数，不带任何参数，但是该函数会转而调用一个global的函数，即ajaxRequest()函数，

- ajaxRequest()参数表—null

- 返回值

该方法无返回值，但是该方法会从服务器POST得到一个JSON格式的url列表，该返回值会保存

在playlist结构中，然后生成列表供用户点选;

- 调用方法

```
ajaxRequest();
```

doDownload()

- 说明

下载按钮的实际响应函数，该函数是一个入口函数，不带任何参数，但是该函数会转而调用一个global的函数，`ajaxDownload()`函数，

- `ajaxDownload()`参数表—null
- 返回值

该方法无返回值，但是该方法会从服务器GET得到一个id值，然后启动定时任务，该任务的作用是POST请求查看返回值，若None，则继续等待服务器裁剪视频，若否，则执行实际的下载，其中`checkIfMerged()`函数用于POST检查视频是否已经在服务器完成裁剪，并修改相应的flag。

- 调用方法

```
ajaxDownload();
```

其他方法大部分为函数间的内部调用，不再赘述，详细可查看`core.js`