Web前端开发

第6章：多媒体

阮晓龙

13938213680 / rxl@hactcm.edu.cn
http://web.book.51xueweb.cn
http://www.51xueweb.cn

河南中医药大学管理科学与工程学科

2018.5

本章主要内容

- 多媒体基础
- HTML5音频与视频
- 播放控制
- 解决兼容性问题
- 字幕
- 案例：使用播放器插件实现视频播放

1.多媒体基础

- 媒体（Media）是人与人之间实现信息交流的中介，简单地说，就是信息的载体，也称为媒介。
- 多媒体（Multimedia）就是多重媒体的意思，可以理解为直接作用于人感官的文字、图形、图像、动画、声音和视频等各种媒体的统称，即多种信息载体的表现形式和传递方式。
- Web上使用的多媒体技术，就是利用计算机把文字、图形、影像、动画、声音及视频等媒体信息都数字化，并将其整合在一定的交互式界面上，使计算机具有交互展示不同媒体形态的能力。

1.多媒体基础

- 音频编码：
  - 将声音调制成模拟信号，通过抽样、量化、编码三个步骤再经算法的方式将连续变化的模拟信号转换为数字编码。
- 音频解码：
  - 将已经编码好的音频还原成连续变化的模拟信号，并给扬声器传递声音信号。

1.多媒体基础

- 编解码器包括有损和无损两种：
  - 无损文件太大，不适合在Web中进行播放。
  - 有损编解码器在编码的过程中会丢失一些原来的音频信息。如果希望编码后的音频能够清晰，需要有良好的音频源、优秀的编码算法、高性能的编码软件和恰当的编码参数。
- 常见的音频格式：
  - CD、WAVE、AIFF、AU、MPEG、MP3、MPEG-4、MIDI、RealAudio、VQF、OggVorbis、AMR、WMA等。

1.多媒体基础

- 视频编码：
  - 通过特定的压缩算法，将某个视频的视频容器转换成另一个视频容器的方式。
- 视频解码：
  - 获取视频容器中的视频、音频等文件并播放的过程。

1.多媒体基础

- 视频播放器（解码器）的工作过程：
  - 解析容器格式以找出可以使用的视频和音频轨道，并分析它们的存储结构，以便接下来的解码工作。
  - 对视频流解码，并在屏幕上显示一幅幅的图像。
  - 对音频流解码，同时给扬声器传输声音信号。
- 常见的视频格式：
  - AVI、MPEG、MOV、ASF、WMV、NAVI、RMVB、3GP、REAL VIDEO、FLV、MKV、F4V、RMVB、WebM等。

1.多媒体基础

1.4在Web上能够使用的音频和视频格式

表 6-01 可在 Web 上播放的音频格式

格式	文件	描述
MIDI	.mid, .midi	MIDI (Musical Instrument Digital Interface) 是一种针对电子音乐设备（比如合成器和声卡）的格式。MIDI 文件不含有声音，但包含可被电子产品（比如声卡）播放的数字音乐指令。
RealAudio	.rm, .ram	RealAudio 格式是由 RealMedia 针对因特网开发的。该格式也支持视频。该格式允许低带宽条件下的音频流（在线音乐、网络音乐）。由于是低带宽优先的，质量常会降低。
Wave	.wav	Wave (waveform) 格式是由 IBM 和微软开发的。所有运行 Windows 的计算机和几乎所有网络浏览器都支持它。
WMA	.wma	WMA 格式 (Windows Media Audio)，质量优于 MP3，兼容大多数播放器。WMA 文件可作为连续的数据流来传输，这使它对于网络电台或在线音乐很实用。
MP3	.mp3, .mpga	MP3 文件实际上是 MPEG 文件的声音部分。MPEG 格式最初是由运动图像专家组开发的。MP3 是其中最受欢迎的针对音乐的声音格式。
OggVorbis	.ogg	OggVorbis 是一种新的音频压缩格式，类似于 MP3 等现有的音乐格式。但有一点不同的是，它是完全免费、开放和没有专利限制的。Vorbis 是这种音频压缩机制的名字，而 Ogg 则是一个计划的名字，该计划意图设计一个完全开放性的多媒体系统。

表 6-02 可在 Web 上播放的视频格式

格式	文件	描述
AVI	.avi	AVI(Audio Video Interleave)格式是由微软开发的。所有运行 Windows 的计算机都支持 AVI 格式。它是因特网上很常见的格式，但非 Windows 计算机通常不能够播放。
WMV	.wmv	Windows Media 格式是由微软开发的。Windows Media 在因特网上很常见，但是如果未安装额外的(免费)组件，就无法播放 Windows Media 视频。
MPEG	.mpg,.mpeg	MPEG (Moving Pictures Expert Group) 格式是因特网上最流行的格式。它是跨平台的，得到了所有主流浏览器的支持。
QuickTime	.mov	QuickTime 格式是由苹果公司开发的。QuickTime 是因特网上常见的格式，但是 QuickTime 视频不能在没有安装额外的(免费)组件的 Windows 计算机上播放。
RealVideo	.rm,.ram	RealVideo 格式是由 Real Media 针对因特网开发的。该格式允许低带宽条件下(在线视频、网络电视)的视频流。由于是低带宽优先的，质量常会降低。
Flash	.swf,.flv	Flash(Shockwave)格式是 Macromedia 开发的。Shockwave 格式需要额外的组件来播放。
WebM	.webm	由 Google 提出，是一个开放、免费的媒体文件格式。WebM 格式其实是以 Matroska(即 MKV)容器格式为基础开发的新容器格式，里面包括了 VP8 影片轨和 Ogg Vorbis 音轨，其中 Google 将其拥有 VP8 视频编码技术以类似 BSD 授权开源，Ogg Vorbis 本来就是开放格式。WebM 标准的网络视频更加偏向于开源并且是基于 HTML5 标准的，WebM 项目旨在为对每个人都开放的网络开发高质量、开放的视频格式，其重点是解决视频服务这一核心的网络用户体验。WebM 的格式相当有效率，可以在 netbook、tablet、手持式装置等上面顺畅地使用。
Mpeg-4	.mp4	Mpeg-4(with H.264 video compression)是一种针对因特网的新格式。事实上，YouTube 推荐使用 MP4。YouTube 接收多种格式，然后全部转换为.flv 或.mp4 以供分发。越来越多的视频发布者转到 MP4，将其作为 Flash 播放器和 HTML5 的因特网共享格式。

1.多媒体基础

- 在HTML5出现之前，向网页中嵌入视频是一件非常麻烦的事，需要引入Flash并且只能使用<object>和<embed>元素来进行。
- 这样的嵌入方式的缺点是：
  - 给Web前端开发者的开发带来了一定的困难。
  - 使得用户在进行视频播放的时候必须安装Flash的浏览器插件才可以播放视频，不方便用户的使用。

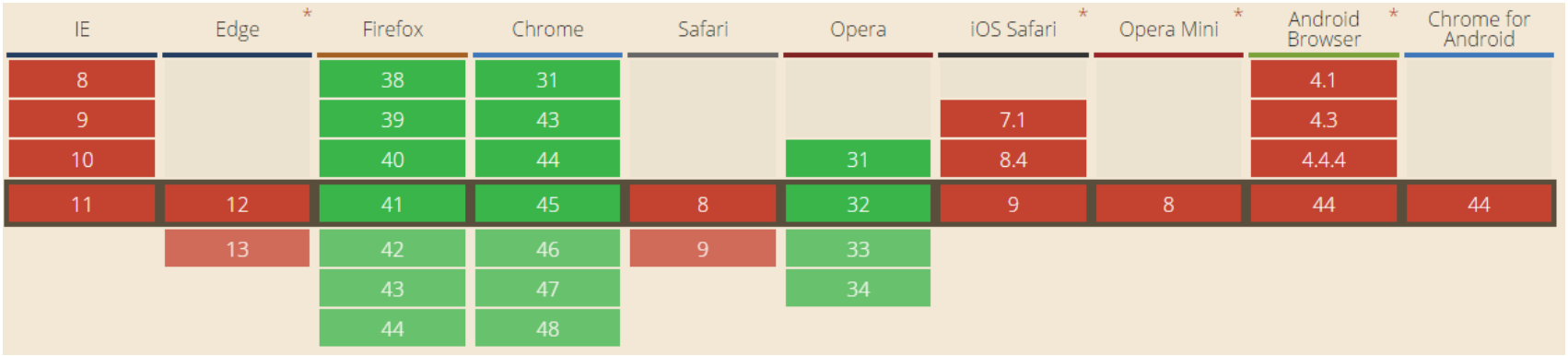
现场演示：

- 案例6-01：在HTML4页面中播放视频文件

2.HTML5音频与视频

- Web前端开发者可以通过audio元素播放声音文件或音频流。
- audio元素支持三种音频格式：
  - Ogg
  - MP3
  - WAV
- Web前端开发者可以通过<source>元素来为同一个音频指定多个源，供不同的浏览器来选择适合自己的播放源。

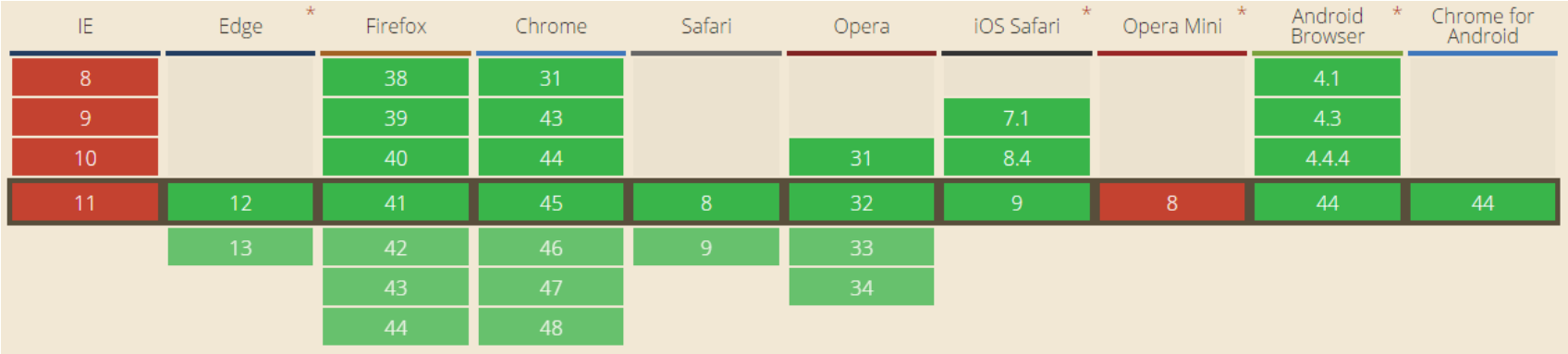
2.HTML5音频与视频



IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8		38	31					4.1	
9		39	43			7.1		4.3	
10		40	44		31	8.4		4.4.4	
11	12	41	45	8	32	9	8	44	44
	13	42	46	9	33				
		43	47		34				
		44	48						

0gg格式音频在浏览器中的支持情况

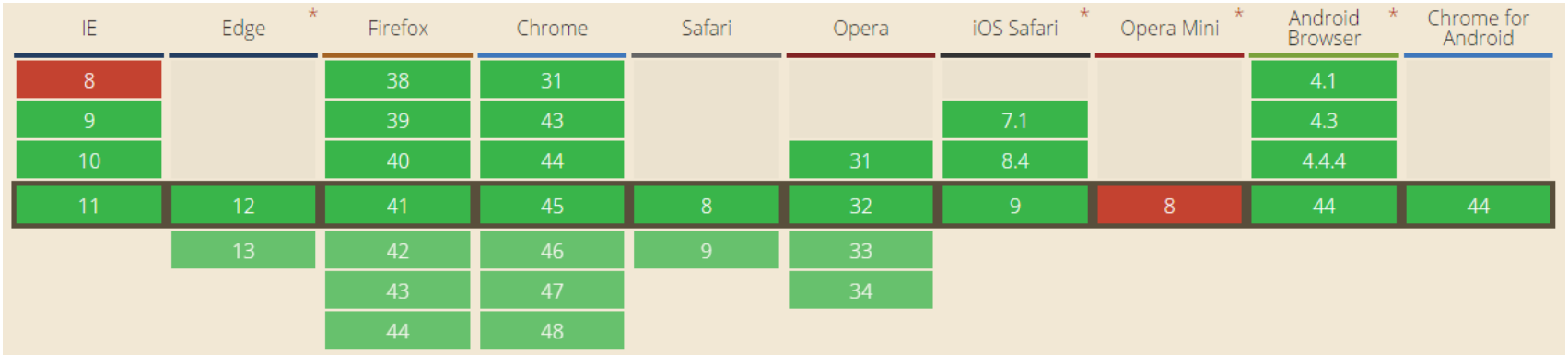
2.HTML5音频与视频



IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8		38	31					4.1	
9		39	43			7.1		4.3	
10		40	44		31	8.4		4.4.4	
11	12	41	45	8	32	9	8	44	44
	13	42	46	9	33				
		43	47		34				
		44	48						

WAV格式音频在浏览器中的支持情况

2.HTML5音频与视频



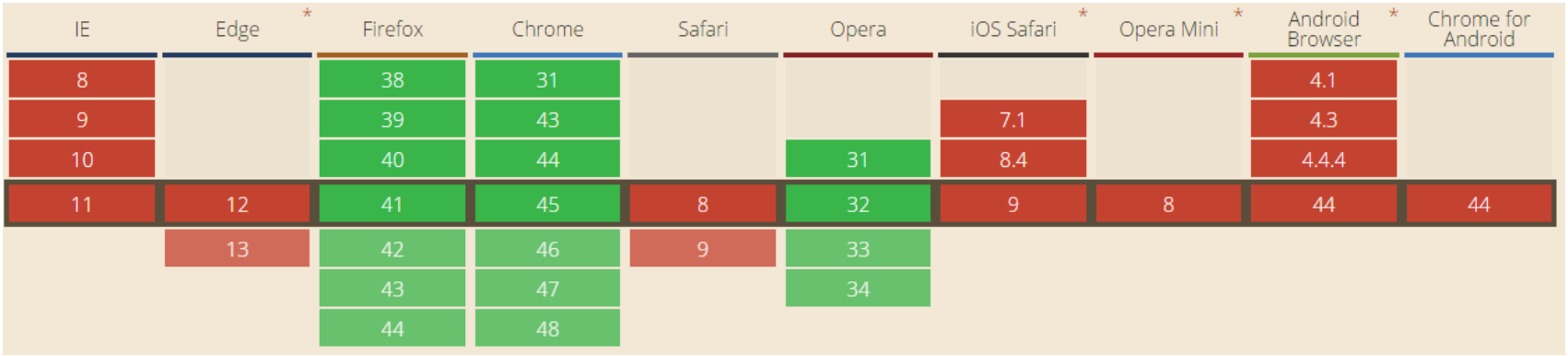
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8		38	31					4.1	
9		39	43			7.1		4.3	
10		40	44		31	8.4		4.4.4	
11	12	41	45	8	32	9	8	44	44
	13	42	46	9	33				
		43	47		34				
		44	48						

MP3格式音频在浏览器中的支持情况

2.HTML5音频与视频

- HTML5 也提供了一系列通过video元素来进行视频的标准方法。
- video元素支持三种视频格式：
  - Ogg：带有Theora视频编码和Vorbis音频编码的Ogg文件。
  - MPEG4：带有H.264视频编码和AAC音频编码的MPEG 4文件
  - WebM：带有VP8视频编码和Vorbis音频编码的WebM文件。
- Web前端开发者可以通过<source>元素来为同一个视频指定多个源，供不同的浏览器来选择适合自己的播放源。

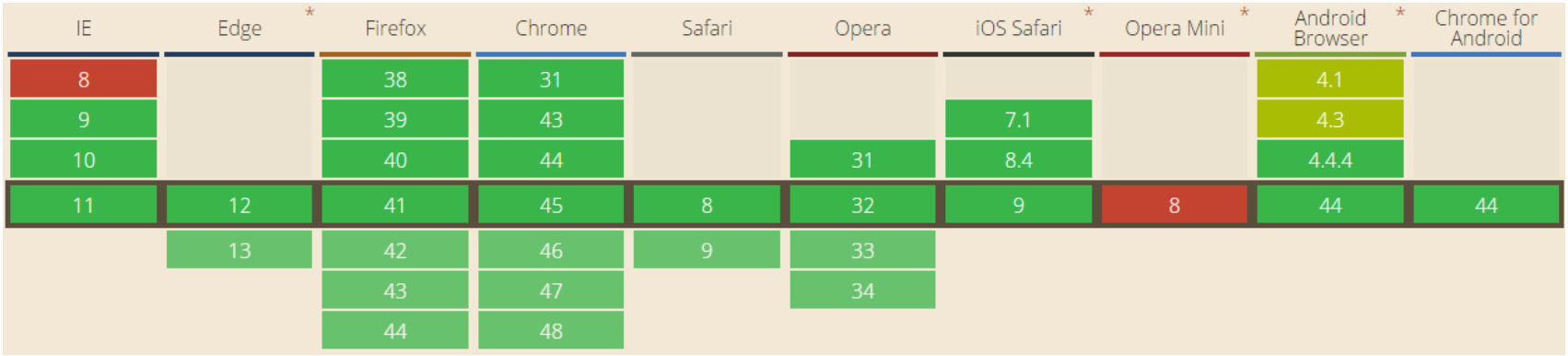
2.HTML5音频与视频



IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8		38	31					4.1	
9		39	43			7.1		4.3	
10		40	44		31	8.4		4.4.4	
11	12	41	45	8	32	9	8	44	44
	13	42	46	9	33				
		43	47		34				
		44	48						

0gg格式音频在浏览器中的支持情况

2.HTML5音频与视频



IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8		38	31					4.1	
9		39	43			7.1		4.3	
10		40	44		31	8.4		4.4.4	
11	12	41	45	8	32	9	8	44	44
	13	42	46	9	33				
		43	47		34				
		44	48						

MPG4格式音频在浏览器中的支持情况

2.HTML5音频与视频

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8		38	31					4.1	
9		39	43			7.1		4.3	
10		40	44		31	8.4		4.4.4	
11	12	41	45	8	32	9	8	44	44
	13	42	46	9	33				
		43	47		34				
		44	48						

WEBM格式音频在浏览器中的支持情况

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
  - src：指定媒体数据的URL地址，即播放视频或音频文件的URL地址。
  - preload：表明视频或音频文件是否需要进行预加载。
    - none：表示不进行预先加载。
    - metadata：表示只预先加载媒体的元数据，主要包括媒体字节数、第一帧、播放列表、持续时间等信息。
    - auto：表示预加载全部视频或音频，该值是默认值。

```
<video src="example.mp4" preload="auto"></video>
```

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
  - poster：video元素的独有属性，用来在视频不可用时，向用户展示一张替代图片，从而避免视频不可用时，页面出现一片空白。

```
<video src="example.mp4" poster="poster.png"></video>
```

  - loop：指定是否循环播放视频或音频。

```
<video src="example.mp4" loop="loop"></video>
```

  - controls：指定是否为视频或音频添加浏览器自带的播放控制条。

```
<video src="example.mp4" controls ="controls"></video>
```

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
  - width和height：video元素的独有属性，指定视频的宽度和高度。

```
<video src="example.mp4" width="400" height="300"></video>
```

  - error：在读取、使用媒体数据的过程中，出现错误时，error属性将返回一个MediaError对象，该对象通过code的方式将错误状态提供出来。
    - 1（MEDIA_ERR_ABORTED）：数据在下载中因用户操作的原因而被中止。
    - 2（MEDIA_ERR_NETWORK）：确认媒体资源可用，但是在下载时出现网络错误，媒体数据的下载过程被中止。
    - 3（MEDIA_ERR_DECODE）：确认媒体资源可用，但是解码时发生错误。
    - 4（MEDIA_ERR_SRC_NOT_SUPPORTER）：媒体格式不被支持。

现场演示：

- 案例6-02：读取错误状态

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
  - networkState：媒体数据在加载过程中读取当前网络状态。
    - 0（NETWORK_EMPTY）：初始状态。
    - 1（NETWORK_IDLE）：浏览器已经选择好用什么编码格式来播放媒体，但尚未建立网络连接。
    - 2（NETWORK_LOADING）：媒体数据加载中。
    - 3（NETWORK_NO_SOURCE）：没有支持的编码格式，不进行加载。
  - currentSrc：读取正在播放中的媒体数据的URL地址。
  - buffered：返回一个对象，该对象实现TimeRange接口，以确认浏览器是否已缓存媒体数据，属性值为只读属性。

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
  - readyState：返回媒体当前播放位置的就绪状态，其为只读属性。
    - 0（HAVE_NOTHING）：没有获得任何媒体的信息，当前没有播放数据。
    - 1（HAVE_METADATA）：已经获得到足够的媒体信息，但是当前播放位置没有有效的媒体数据，暂时不能够播放。
    - 2（HAVE_CURRENT_DATA）：当前播放位置已经有数据可以播放，但没有获取到可以让播放器前进的数据。
    - 3（HAVE_FUTURE_DATA）：当前播放位置已经有数据可以播放，而且也获取到了可以让播放器前进的数据。
    - 4（HAVE_ENOUGH_DATA）：当前播放位置已经有数据可以播放，下一帧数据已经获得，且浏览器确认媒体数据以某一种速度进行加载，可以保证有足够的后续数据进行播放。

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
  - seeking和seekable
    - seeking属性：返回布尔值，表示浏览器是否正在请求某一特定播放位置的数据，true表示浏览器正在请求数据；false表示浏览器已经停止请求。
    - seekable属性：返回TimeRange对象，表示请求到的数据的时间范围。
  - currentTime、startTime和duration
    - currentTime属性：读取媒体的当前播放位置，通过修改该属性值可以修改当前播放位置。
    - startTime属性：读取媒体播放的开始时间，通常为0。
    - duration属性来读取媒体文件总的播放时间。

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
  - played、paused和ended
    - played属性：返回一个TimeRange对象，读取媒体文件已经播放部分的时间段。
    - paused属性：返回一个布尔值，表示是否处于暂停播放状态。
    - ended属性：返回一个布尔值，表示是否已经播放完毕。
  - efaultPlaybackRate和playbackRate
    - defaultPlaybackRate属性：读取或修改媒体默认的播放速率。
    - playbackRate属性：读取或修改媒体当前的播放速率。

2.HTML5音频与视频

- audio和video元素的元素属性大致相同：
- volume和muted
  - volume属性：读取或修改媒体播放的音量，范围为0到1。
  - muted属性：读取或修改媒体的静音状态，该属性值为布尔值。
- autoplay：设置或返回音视频是否在加载后即开始播放，属性值为true或false。

```
<video src="example.mp4" autoplay="ture">
```

2.HTML5音频与视频

- video元素和audio元素具有四种方法：
  - play：使用该方法来播放媒体，自动将元素的paused值变为false。
  - pause：使用该方法来暂停播放，自动将元素的paused值变为true。
  - load：使用load方法来重新载入媒体进行播放。
  - canPlayType：使用该方法来测试浏览器是否支持指定的媒体类型。
    - 空字符串：表示浏览器不支持此种媒体类型。
    - maybe：表示浏览器可能支持此种媒体类型。
    - probably：表示浏览器确定支持此种媒体类型。

```
var supportTypeInfo = videoElement.canPlayType(type);
```

2.HTML5音频与视频

- 事件处理方式
  - 监听：使用video或audio元素的addEventListener方法来监听事件发生。

```
videoElement.addEventListener(type,listener,userCapture);
```

    - videoElement：表示页面上的video或audio元素。
    - type：事件名称。
    - listener：表示绑定的函数。
    - useCapture：一个布尔值，表示该事件的响应顺序。
  - 使用JavaScript脚本中的获取事件句柄。

2.HTML5音频与视频

- 事件

表 6-03 HTML5 中的 Audio/Video 事件

事件	描述
abort	当音频/视频的加载已放弃时
canplay	当浏览器可以播放音频/视频时
canplaythrough	当浏览器可在不因缓冲而停顿的情况下进行播放时
durationchange	当音频/视频的时长已更改时
emptied	当目前的播放列表为空时
ended	当目前的播放列表已结束时
error	当在音频/视频加载期间发生错误时
loadeddata	当浏览器已加载音频/视频的当前帧时

loadedmetadata	当浏览器已加载音频/视频的元数据时
loadstart	当浏览器开始查找音频/视频时
pause	当音频/视频已暂停时
play	当音频/视频已开始或不再暂停时
playing	当音频/视频在已因缓冲而暂停或停止后已就绪时
progress	当浏览器正在下载音频/视频时
ratechange	当音频/视频的播放速度已更改时
seeked	当用户已移动/跳跃到音频/视频中的新位置时
seeking	当用户开始移动/跳跃到音频/视频中的新位置时
stalled	当浏览器尝试获取媒体数据，但数据不可用时
suspend	当浏览器刻意不获取媒体数据时
timeupdate	当目前的播放位置已更改时
volumechange	当音量已更改时

2.HTML5音频与视频

- 简介
  - 使用HTML5 <audio>元素实现简单的背景音乐播放，页面被打开后直接加载音频文件，音频文件加载完成后播放，并且可以循环播放音乐。

现场演示：

- 案例6-03：在网页上使用背景音乐

2.HTML5音频与视频

- 简介
  - 使用HTML5 <video>元素实现简单的视频播放功能。

现场演示：

- 案例6-04：在网页上播放视频

3.播放控制

- 预加载媒体文件是播放器中很重要的一项功能，预先将媒体文件加载，可节省等待的时间，提高用户体验。
- HTML5实现预加载功能十分简单，只需添加preload属性并设置其属性值即可。

```
<video src="medias/Wo-ShangWenjie.mp4" controls="controls" width="552"
preload="auto" height="331"></video>
```

3.播放控制

- <video>中提供了poster属性，可以为视频播放器添加封面图。
- poster属性使用十分简单，只需要在<video>标签中加入poster属性并指定封面图路径即可。

```
<video src="medias/Wo-ShangWenjie.mp4" controls="controls" width="552" preload="auto"
poster="medias/Wo-ShangWenjie.png" height="331"></video>
```

3.播放控制

- video元素声明了autoplay属性，页面加载完成后，视频马上会被自动播放。
- video元素增加了两个事件处理函数：
  - 当视频加载完毕，准备开始播放的时候，会触发oncanplay函数来执行预设的动作。
  - 当视频播放完成后，会触发onended函数以停止帧的创建。

```
<video src="medias/Wo-ShangWenjie.mp4" controls="controls" width="552" preload="auto"
height="331" autoplay="ture"></video>
```

3.播放控制

- 在video元素中添加loop属性即可实现视频的循环播放。

```
<video src="medias/Wo-ShangWenjie.mp4" controls="controls" width="552" preload="auto"
height="331" autoplay="ture"></video>
```

3.播放控制

- 进行播放控制前，需要设置一些有助于调整实例的变量：
  - speed是视频播放速度，默认为1。
  - volume为视频音量，默认为1。
  - muted设置静音状态，默认为不静音。

```
//默认播放速度为 1
var speed=1;
//默认音量为 1
var volume=1;
//默认静音状态为否
var muted=false;
```

3.播放控制

- 除了<video>元素中自带的控制条之外，还可以使用<video>的方法、属性和事件来自定义控制条，其步骤为：
  - 清除<video>默认的播放控制条。
  - 使用CSS先定义按钮的样式。
  - 向页面中添加播放按钮元素。
  - 添加一个用于展示播放器状态的div元素。
  - 添加<video>的播放函数videoPlay()。
  - 将播放函数与播放按钮绑定，点击播放按钮后触发播放事件。

3.播放控制

3.7暂停

- 暂停操作是基于<video>的pasue事件开发的。
- 添加暂停操作的具体步骤和播放按钮的制作相类似。

3.播放控制

- <video>中可以通过playbackRate获取或设置video播放速度，可以通过此属性进行快放、慢放、慢动作的播放控制。

3.播放控制

- <video>中的currentTime属性可以获取或设置视频中当前的播放位置，其获取的时间以秒为单位。

3.播放控制

3.10进度拖动

- 进度条是视频播放是常用的功能，不仅显示视频播放的进度，还可以直接控制视频的播放进度。
- 进度条的实现是基于currentTime属性。

3.播放控制

- 在进行视频播放时，会经常的根据需要调整音量，可增大音量、降低音量和设置静音。
  - <video>的属性muted可以设置是否静音。
  - <video>的volume属性用于设置视频的音量。

3.播放控制

- 全屏播放是视频播放中很重要的一个功能，其实现过程如下：
  - 设置全屏播放时的样式。
  - 添加全屏播放控制按钮。
  - 添加反射调用函数。
  - 添加全屏函数launchFullscreen()。
  - 添加退出全屏函数exitFullscreen()。
  - 点击全屏按钮后，执行全屏函数，将播放器全屏。

3.播放控制

- <video>元素中提供了表示视频错误状态的error属性，根据此属性进行播放器的容错处理。

现场演示：

- 案例6-05：HTML5播放器-播放控制

4.解决兼容性问题

- HTML5的Video和Audio元素具有通用、集成和可视化的播放控制API，极大的方便了用户和开发人员。
- 并不是所有的浏览器都支持该两种元素，又由于解码方式不同，不同浏览器对其元素的媒体格式支持也不同。
- 查看浏览器对元素的支持情况，主要有以下两种方法：
  - 可以通过网站http://html5test.com查看自己浏览器对video详细支持情况。
  - 以使用动态脚本的方式创建并检测特定函数是否存在。

```
var hasVideo=!!(docment.creatElement('video').canPlayType);
```

现场演示：

- 案例6-06：使用Video和Audio的备选内容

4.解决兼容性问题

- source元素为<video>和 <audio>定义媒介资源，允许规定可替换的视频/音频文件供浏览器根据自身对媒体类型或者编解码器的支持进行选择。
- <source>元素的属性：
  - src：指定视频源的URL地址。
  - type：指定视频源的类型。
  - media：指定视频的预期媒体类型。

现场演示：

- 案例6-07：使用多媒体元素提升兼容性

4.解决兼容性问题

- 使用多种媒体格式提升兼容性只能提升支持video和audio元素的浏览器的兼容性，如果浏览器不支持该元素，就需要使用Flash代替video和audio元素播放多媒体。
- 利用脚本检测浏览器对video的支持情况，如果支持的话，使用video播放视频，不支持的话引入Flash播放视频。

现场演示：

- 案例6-08：使用Flash提升兼容性

5.字幕

- 网络视频文本轨道（简称为WebVTT）是用于标记文本轨道的文件格式。
- 它与HTML5的<track>元素相结合，可给音频、视频等媒体资源添加字幕、标题和其他描述信息，并同步显示。

5.字幕

- 文件内容
  - WebVTT文件是一个简单的纯文本文件，里面包含了以下几种类型的视频信息：
    - 字幕：关于对话的转译或者翻译。
    - 标题：类似于标题，但是还包括音响效果和其他音频信息。
    - 说明：预期为一个单独的文本文件，通过屏幕阅读器描述视频。
    - 章节：旨在帮助用户浏览整个视频。
    - 元数据：默认不打算展示给用户的、和视频有关的信息和内容。

5.字幕

- 文件格式
  - WebVTT文件是一个以UTF-8为编码，以.vtt 为文件扩展名。
  - WebVTT文件的头部按如下顺序定义：
    - 可选的字节顺序标记（BOM）。
    - 字符串WEBVTT。
    - 一个空格（Space）或制表符（Tab），后面接任意非回车换行的元素。
    - 两个或两个以上的 “WEBVTT行结束符”：回车\r，换行\n，或者同时回车换行\r\n。

```
1
00:00:15.000 --> 00:00:18.000
字幕或标题内容
```

5.字幕

- WebVTT标记
  - WebVTT 文件可以包含一个或多个“WebVTT Cues”，各个之间用两个或多个WebVTT 行结束符分隔开。
  - WebVTT 标记允许指定特定时间戳范围内的文字（如字幕），同时也指定一个唯一的标识符，标识符由简单字符串构成，不包含-->，也不包含任何的WebVTT行结束符。

```
[idstring]
[hh:]mm:ss.msmsms --> [hh:]mm:ss.msmsms
字幕或标题内容
```

5.字幕

- WebVTT标记
  - 时间戳遵循一个标准格式：
    - 小时部分[hh:]是可选的，毫秒和秒用一个点（如“.”）分离，而不是冒号，时间戳范围的后者必须大于前者。对于不同的Cues，时间戳可以重叠，但在单个标记中，不能有字符串-->或两个连续的行结束符。

```
1
00:00:52.000 --> 00:00:54.000
Web 前端开发技术
2
00:00:55.167 --> 00:00:57.042
视频播放技术
```

5.字幕

- WebVTT Cue设置
  - 在时间范围值后面，可以为标记做设置，具体设置规则如下：

```
[idstring]
[hh:]mm:ss.msmsms --> [hh:]mm:ss.msmsms [标记设置]
文本内容
```

5.字幕

表 6-04 WebVTT Cue 设置

设置	值	功能说明
vertical	rl || lr	将文本纵向向左对齐 （lr） 或向右对齐 （rl） （如：日文的字幕）
line	[-][0 or more]	行位置，负数从框底部数起，正数从顶部数起
position	[0-100]%	百分数意味着文字开始时离框左边的位置（如：英文字幕）
size	[0-100]%	百分数意味着 cue 框的大小是整体框架宽度的百分比
align	start || middle || end	指定 cue 中文本的对齐方式

注：如果没有设置 Cue 选项，默认位置是底部居中。

5.字幕

- WebVTT标记内联样式

表 6-05 WebVTT Cue 内联样式

值	功能说明
c	用 c 定义（CSS）类，例如，〈c.className〉Cue text〈/c〉
i	斜体字
b	粗体字
u	添加下划线
ruby	定义类似于 HTML5 的〈ruby〉元素。在这样的内联样式中，允许出现一个或多个〈rt〉元素。（〈ruby〉元素用于标注汉字等东亚字符的发音）
v	如有提供，则用来指定声音标签。例如，〈v Ian〉This is useful for adding subtitles〈/v〉。注意此声音标签不会显示，它只是作为一个样式标记。

5.字幕

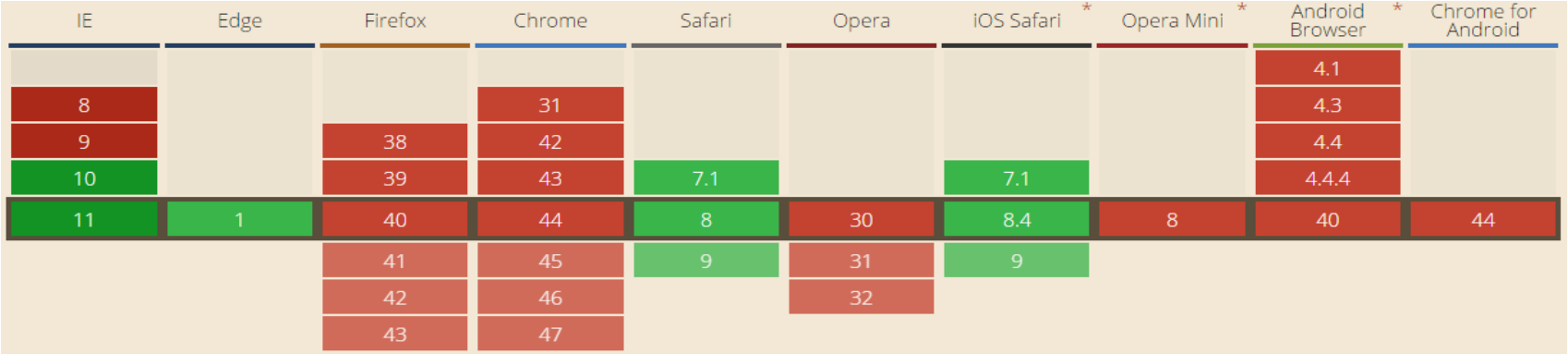
- 使用<track>元素

表 6-06　track 元素属性

名称	值	说明
kind	subtitles	字幕
	captions	标题，不仅仅是标题，还包括音效及其他音频信息
	descriptions	描述，视频的文本描述
	chapters	章节导航
	metadata	元数据
src	URL	指定资源 URL
srclang	Language code	src 资源的语言
label	Free text	给元素添加标签
default	n/a	如果存在，且用户无其他特别设定，这个元素默认启用

5.字幕

- 浏览器支持情况



IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
								4.1	
8			31					4.3	
9		38	42					4.4	
10		39	43	7.1		7.1		4.4.4	
11	1	40	44	8	30	8.4	8	40	44
		41	45	9	31	9			
		42	46		32				
		43	47						

track的浏览器支持情况

5.字幕

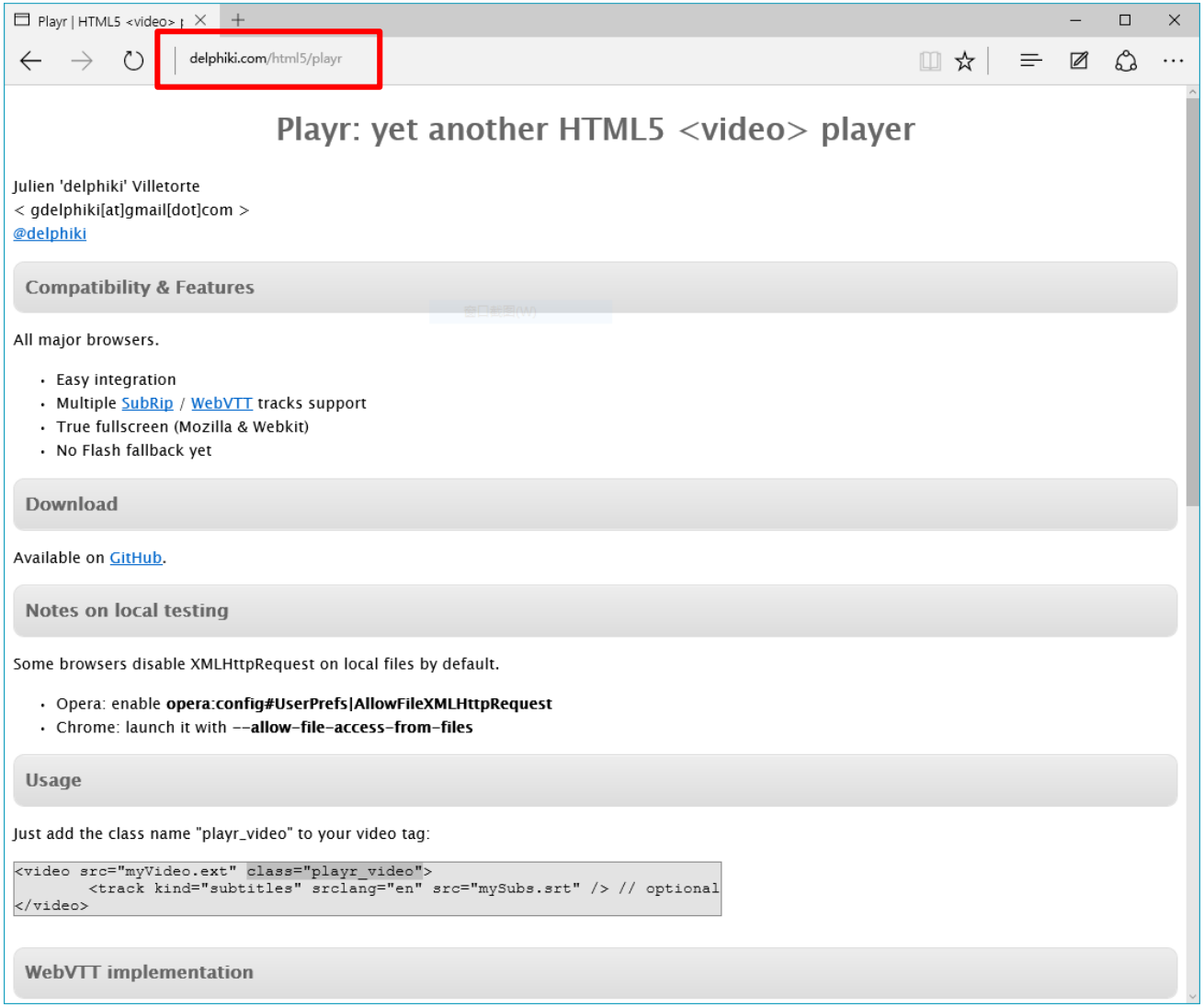
- 制作一个简单的包含视频字幕的播放器，主要实现在IE11浏览器上的视频音乐MV的功能，并实现字幕浏览以及播放控制功能。

现场演示：

- 案例6-09：带字幕的视频播放器

6.案例：使用播放器插件实现视频播放

- 案例说明：
  - 插件是在Web前端开发中经常会用到的工具，本案例将通过使用播放器插件的方式实现Web前端中的视频播放。
- 插件获取：
  - 插件名称为Playr，作者是Julien Villetorte可实现支持字幕，标题，及章节。
  - 通过Github下载Playr（https://github.com/delphiki/Playr）。
- 案例代码：
  - 将JavaScript和CSS文件引入到网页中，并在video中添加类名称player_video。


Playr | HTML5 <video> p
delphiki.com/html5/playr
Playr: yet another HTML5 <video> player
Julien 'delphiki' Villetorte
< gdelphiki[at]gmail[dot]com >
@delphiki
Compatibility & Features
All major browsers.
Easy integration
Multiple SubRip / WebVTT tracks support
True fullscreen (Mozilla & Webkit)
No Flash fallback yet
Download
Available on GitHub.
Notes on local testing
Some browsers disable XMLHttpRequest on local files by default.
Opera: enable opera:config#UserPrefs|AllowFileXMLHttpRequest
Chrome: launch it with --allow-file-access-from-files
Usage
Just add the class name "playr_video" to your video tag:
<video src="myVideo.ext" class="playr_video">
<track kind="subtitles" srclang="en" src="mySubs.srt" /> // optional
</video>
WebVTT implementation

现场演示：

- 案例6-10：视频播放器的实现

Thanks.