

**HTML5**

目录

[目录 2](#_Toc455376892)

[第1章 前言 5](#_Toc455376893)

[1.1 认识HTML5 5](#_Toc455376894)

[1.2 语法规范 6](#_Toc455376895)

[第2章 语义化 7](#_Toc455376896)

[2.1 语义化标签 7](#_Toc455376897)

[2.1.1 常用语义标签 7](#_Toc455376898)

[2.1.2 兼容处理 8](#_Toc455376899)

[2.2 微数据 9](#_Toc455376900)

[2.3 WAI-ARIA 9](#_Toc455376901)

[第3章 表单 9](#_Toc455376902)

[3.1 输入类型 10](#_Toc455376903)

[3.2 表单元素 11](#_Toc455376904)

[3.3 表单属性 11](#_Toc455376905)

[第4章 多媒体 12](#_Toc455376906)

[4.1 音频 12](#_Toc455376907)

[4.2 视频 13](#_Toc455376908)

[第5章 DOM扩展 14](#_Toc455376909)

[5.1 获取元素 14](#_Toc455376910)

[5.2 类名操作 14](#_Toc455376911)

[5.3 自定义属性 15](#_Toc455376912)

[2 新增API 15](#_Toc455376913)

[2.1 网络状态 15](#_Toc455376914)

[2.2 全屏 16](#_Toc455376915)

[2.3 文件读取 16](#_Toc455376916)

[2.3.1 FileList对象 17](#_Toc455376917)

[2.3.2 FileReader对象 17](#_Toc455376918)

[2.4 拖拽 17](#_Toc455376919)

[2.4.1 拖拽元素 17](#_Toc455376920)

[2.4.2 目标元素 18](#_Toc455376921)

[2.4.3 事件监听 18](#_Toc455376922)

[2.5 历史管理 18](#_Toc455376923)

[2.5.1 旧版本 19](#_Toc455376924)

[2.5.2 新增方法 19](#_Toc455376925)

[2.5.3 事件监听 19](#_Toc455376926)

[2.6 地理定位 19](#_Toc455376927)

[2.6.1 获取地理信息方式 20](#_Toc455376928)

[2.6.2 隐私 20](#_Toc455376929)

[2.6.3 API详解 21](#_Toc455376930)

[2.7 Web存储 21](#_Toc455376931)

[2.7.1 特性 22](#_Toc455376932)

[2.7.2 window.sessionStorage 22](#_Toc455376933)

[2.7.3 window.localStorage 22](#_Toc455376934)

[2.7.4 方法详解 22](#_Toc455376935)

[2.7.5 其它 22](#_Toc455376936)

[2.8 应用缓存 23](#_Toc455376937)

[2.8.1 优势 23](#_Toc455376938)

[2.8.2 缓存清单 23](#_Toc455376939)

[2.8.3 manifest文件格式 23](#_Toc455376940)

[2.8.4 事件监听 24](#_Toc455376941)

[2.8.5 其它 24](#_Toc455376942)

[2.9 多媒体 24](#_Toc455376943)

[2.10 应用 24](#_Toc455376944)

# 前言

## 认识HTML5

HTML5并不仅仅只是作为HTML标记语言的一个最新版本，更重要的是它制定了Web应用开发的一系列标准，成为第一个将Web做为应用开发平台的HTML语言。

HTML5定义了一系列新元素，如新语义标签、智能表单、多媒体标签等，可以帮助开发者创建富互联网应用，还提供了一系列Javascript API，如地理定位、重力感应、硬件访问等，可以在浏览器内实现类原生应用，甚至结合Canvas我们可开发网页版游戏，同时结合CSS3的过渡、转换、动画等特性，可以极大的增强用户体验，提升开发功能的可应用性。

我们日常讨论的H5其实是一个泛称，它指的是由HTML5 + CSS3 + Javascript等技术组合而成的一个应用开发平台。

## 语法规范

随着Web技术的更新，HTML也先后经历了HTML 4.01、XHTML 1.0、HTML5几个重要的版本，在版本的演变过程中新增或废弃了一些属性，同时对语法规范也做了一些调整，为了能够保证浏览器可以兼容不同版本语法规范的，我们可以使用<!DOCTYPE>指示浏览器应该如何处理我们的HTML。

见示例代码1-1.html

HTML5在语法规范上也做了比较大的调整，去除了许多冗余的内容，书写规则更加简洁、清晰。

见示例代码1-2.html

通过以上的比较我们可以总结得出HTML5在语法规范上的特点：

1、更简洁

2、更宽松

但是我们在实际开发中应规范书写，不建议太随意 ！并且现在基本所有网站都采用了HTML5的语法规范。

**W3C验证地址**

https://validator.w3.org/

# 语义化

所谓语义化是要使HTML标签具备很好的可读性，可以清晰传达每个标签所要表达的意义，以方便其被友好的处理和解析。

## 语义化标签

对于语义化标签我们并不陌生，如<p>表示一个段落、<ul>表示一个无序列表<h1> ~ <h6>表示一系列标题等，在此基础上HTML5增加了大量更有意义的语义标签，更有利于搜索引擎或辅助设备来理解HTML页面内容。

传统的做法我们通过添加类名如class="header"、class="footer"，使HTML页面具有语义性的，但是不具有通用性（如class="header"也可能被写成class="head"）。

见示例代码2-1.html

HTML5则是通过增加语义化标签的形式来解决这个问题，例如<header></header>、<footer></footer>等，这样就可以保证其具有通用性。

见示例代码2-2.html

此章节学习目的为了解增加语义标签的目的，以及各语义标签所表达的意义，在网页布局中能够合理使用标签。

### 常用语义标签

|  |  |
| --- | --- |
| 标签 | 语义 |
| <nav> | 导航 |
| <header> | 页眉 |
| <footer> | 页脚 |
| <section> | 区块 |
| <article> | 文章 |
| <aside> | 侧边栏 |
| <progress> | 进度条 |
|  |  |

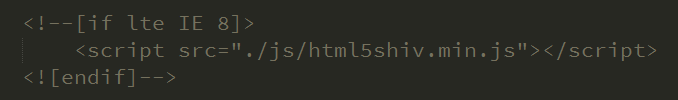
见示例代码2-3.html

本质上新语义标签与<div>、<span>没有区别，只是其具有语义性，使用时除了在HTML结构上需要注意外，其它和普通标签的使用无任何差别，可以理解成<div class="nav"> 相当于 <nav>。**不要好奇，它只是一个标签！**

尽量避免全局使用header、footer、aside等语义标签。

### 兼容处理

在不支持HTML5新标签的浏览器里，会将这些新的标签解析成行内元素(inline)对待，所以我们只需要将其转换成块元素(block)即可使用，但是在IE9版本以下，并不能正常解析这些新标签，但是却可以识别通过document.createElement('tagName')创建的自定义标签，于是我们的解决方案就是将HTML5的新标签全部通过document.createElement('tagName')来创建一遍，这样IE低版本也能正常解析HTML5新标签了，但在实际开发中我们更多采用的是通过检测IE浏览器的版本来加载第三方的一个JS库来解决兼容问题，这个库文件会帮自动通过document.createElement('tagName')创建所有HTML5的新标签。



见示例代码2-4.html

## 微数据

微数据是在如 span、div 的标签内添加属性，让机器（如搜索引擎）识别其含义，某些特定类型的信息，例如评论、人物信息或事件都有相应的属性，用来描述其含义，可以理解成新语义标签的一种补充。

见示例代码2-5.html

[更多学习资料](http://www.zhangxinxu.com/wordpress/2011/12/html5%E6%89%A9%E5%B1%95-%E5%BE%AE%E6%95%B0%E6%8D%AE-%E4%B8%B0%E5%AF%8C%E7%BD%91%E9%A1%B5%E6%91%98%E8%A6%81/)

## WAI-ARIA

WAI-ARIA**[1]**, 是Web Accessibility Initiative - Accessible Rich Internet Applications 的缩写，她主要解决的一个问题：让残障人士能无障碍地访问网页上的动态内容。

[见参考资料](http://www.zhangxinxu.com/wordpress/2012/03/wai-aria-%E6%97%A0%E9%9A%9C%E7%A2%8D%E9%98%85%E8%AF%BB/)

# 表单

伴随着互联网富应用以及移动开发的兴起，传统的Web表单已经越来越不能满足开发的需求，所以HTML5在Web表单方向也做了很大的改进，如拾色器、日期/时间组件等，使表单处理变的更加高效。

## 输入类型

|  |  |  |
| --- | --- | --- |
| 类型 | 使用示例 | 含义 |
| email | <input type="email"> | 输入邮箱格式 |
| tel | <input type="tel"> | 输入手机号码格式 |
| url | <input type="url"> | 输入url格式 |
| number | <input type="number"> | 输入数字格式 |
| search | <input type="search"> | 搜索框（体现语义化） |
| range | <input type="range"> | 自由拖动滑块 |
| color | <input type="color"> | 拾色器 |
| time | <input type="time"> |  |
| date | <input type="date"> |  |
| datetime | <input type="datetime"> |  |
| month | <input type="month"> |  |
| week | <input type="week"> |  |

见示例代码3-1.html

部分类型是针对移动设备生效的，且具有一定的兼容性，在实际应用当中应选择性的使用，扫描二维码可以对比测试。



## 表单元素

|  |  |
| --- | --- |
| 元素 | 含义 |
| <datalist> | 数据列表 |
| <keygen> | 生成加密字符串 |
| <output> | 输出结果 |
| <meter> | 度量器 |

见示例代码3-2.html

## 表单属性

|  |  |  |
| --- | --- | --- |
| 属性 | 用法 | 含义 |
| placeholder | <input type="text" placeholder="请输入用户名"> | 占位符 |
| autofocus | <input type="text" autofocus> | 自动获得焦点 |
| multiple | <input type="file" multiple> | 多文件上传 |
| autocomplete | <input type="text" autocomplete="off"> | 自动完成 |
| form | <input type="text" form="某表单ID"> |  |
| novalidate | <form novalidate></form> | 关闭验证 |
| required | <input type="text" required> | 必填项 |
| pattern | <input type="text" pattern="\d"> | 自定义验证 |

见示例代码3-3.html

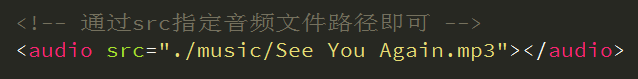
# 多媒体

在HTML5之前，在网页上播放音频/视频的通用方法是利用Flash来播放，但是大多情况下，并非所有用户的浏览器都安装了Flash插件，由此使得处理音频/视频播放变的非常复杂，并且移动设备的浏览器并不支持Flash插件。

## 音频

HTML5通过<audio>标签来解决音频播放的问题。

使用相当简单，如下图所示



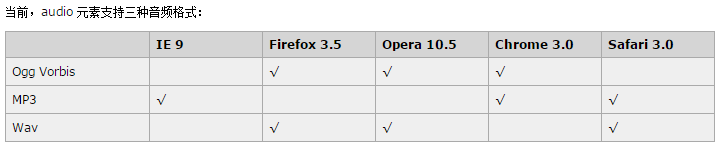
并且可以通过附加属性可以更友好控制音频的播放，如：

autoplay 自动播放

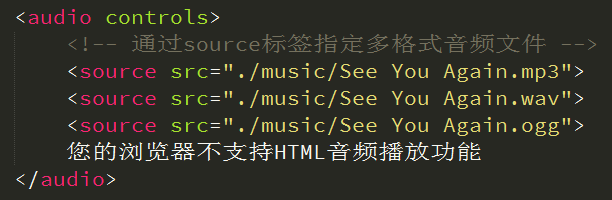
controls 是否显不默认播放控件

loop 循环播放

由于版权等原因，不同的浏览器可支持播放的格式是不一样的，如下图供参考



**多浏览器支持的方案，如下图**

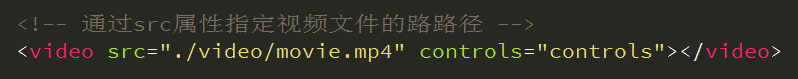


见示例代码4-1.html

## 视频

HTML5通过<video>标签来解决音频播放的问题。

同音频播放一样，<video>使用也相当简单，如下图



同样，通过附加属性可以更友好的控制视频的播放

autoplay 自动播放

controls 是否显示默认播放控件

loop 循环播放

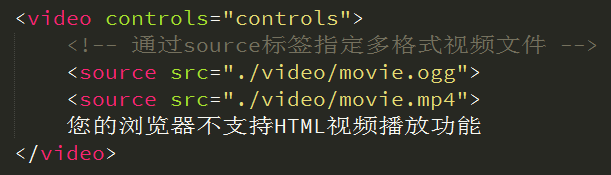
width 设置播放窗口宽度

height 设置播放窗口的高度

由于版权等原因，不同的浏览器可支持播放的格式是不一样的，如下图供参考



**多浏览器支持的方案，如下图**



见示例代码4-2.html

# DOM扩展

## 获取元素

1、document.getElementsByClassName ('class') 通过类名获取元素，以伪数组形式存在。

见示例代码5-1.html

2、document.querySelector('selector') 通过CSS选择器获取元素，符合匹配条件的第1个元素。

见示例代码5-2.html

3、document.querySelectorAll('selector') 通过CSS选择器获取元素，以伪数组形式存在。

见示例代码5-3.html

## 类名操作

1、Node.classList.add('class') 添加class

2、Node.classList.remove('class') 移除class

3、Node.classList.toggle('class') 切换class，有则移除，无则添加

4、Node.classList.contains('class') 检测是否存在class

Node指一个有效的DOM节点，是一个通称。

见示例代码5-4.html

## 自定义属性

在HTML5中我们可以自定义属性，其格式如下data-\*=""，例如

data-info="我是自定义属性"，通过Node.dataset['info'] 我们便可以获取到自定义的属性值。

Node.dataset是以对象形式存在的，当我们为同一个DOM节点指定了多个自定义属性时，Node.dataset则存储了所有的自定义属性的值。

假设某元素 <div id="demo" data-name="itcast" data-age="10">

var demo = document.querySelector('#demo');

1、读取 demo.dataset['name'] 或者 demo.dataset['age']

2、设置demo.dataset['name'] = 'web developer'

见示例代码5-5.html

注：当我们如下格式设置时，则需要以驼峰格式才能正确获取

<div data-my-name="itcast"> 这样获取Node.dataset['myName']

# 新增API

## 网络状态

我们可以通过window.onLine来检测，用户当前的网络状况，返回一个布尔值。

window.online用户网络连接时被调用

window.offline用户网络断开时被调用

见代码示例online/6-1.html

## 全屏

HTML5规范允许用户自定义网页上任一元素全屏显示。

1、Node.requestFullScreen() 开启全屏显示

2、Node.cancelFullScreen() 关闭全屏显示

由于其兼容性原因，不同浏览器需要添加前缀如：

webkit内核浏览器：webkitRequestFullScreen、webkitCancelFullScreen，如chrome浏览器。

Gecko内核浏览器：mozRequestFullScreen、mozCancelFullScreen，如火狐浏览器。

3、document.fullScreen检测当前是否处于全屏

不同浏览器需要添加前缀

document.webkitIsFullScreen、document.mozFullScreen

全屏伪类选择器

:full-screen .box {}、:-webkit-full-screen {}、:moz-full-screen {}

见代码示例fullscreen/6-2.html

## 文件读取

通过FileReader对象我们可以读取本地存储的文件，使用 [File](https://developer.mozilla.org/zh-CN/docs/DOM/File)对象来指定所要读取的文件或数据。其中File对象可以是来自用户在一个 [<input>](https://developer.mozilla.org/zh-CN/docs/HTML/Element/input)元素上选择文件后返回的[FileList](https://developer.mozilla.org/zh-CN/docs/DOM/FileList)对象，也可以来自由拖放操作生成的  [DataTransfer](https://developer.mozilla.org/zh-CN/DragDrop/DataTransfer)。

### FileList对象

由于HTML5中我们可以通过为表单元素添加multiple属性，因此我们通过<input>上传文件后得到的是一个FileList对象（伪数组形式）。

见代码示例file/6-3.html

### FileReader对象

HTML5新增内建对象，可以读取本地文件内容。

var reader = new FileReader; 可以实例化一个对象

**实例方法**

1、[readAsDataURL](https://developer.mozilla.org/zh-CN/docs/Web/API/FileReader#readAsDataURL())() 以DataURL形式读取文件

**事件监听**

onload 当文读取完成时调用

**属性**

result 文件读取结果

见代码示例file/6-4.html

[参考资料](https://developer.mozilla.org/zh-CN/docs/Web/API/FileReader#toc)

## 拖拽

在HTML5的规范中，我们可以通过为元素增加draggable="true"来设置此元素是否可以进行拖拽操作，其中图片、链接默认是开启的。

### 拖拽元素

页面中设置了draggable="true"属性的元素，其中<img>、<a>标签默认是可以被拖拽的

见代码示例drag/6-5.html

### 目标元素

页面中任何一个元素都可以成为目标元素，不需要特殊指定。

### 事件监听

根据元素类型不同，需要设置不同的事件监听

**1、拖拽元素**

ondrag 应用于拖拽元素，整个拖拽过程都会调用

ondragstart 应用于拖拽元素，当拖拽开始时调用

ondragleave 应用于拖拽元素，当鼠标离开拖拽元素时调用

ondragend 应用于拖拽元素，当拖拽结束时调用

**2、目标元素**

ondragenter 应用于目标元素，当拖拽元素进入时调用

ondragover 应用于目标元素，当停留在目标元素上时调用

ondrop 应用于目标元素，当在目标元素上松开鼠标时调用

ondragleave 应用于目标元素，当鼠标离开目标元素时调用

见代码示例drag/6-6.html

## 地理定位

在HTML规范中，增加了获取用户地理信息的API，这样使得我们可以基于用户位置开发互联网应用，即基于位置服务 (Location Base Service)

### 获取地理信息

通过IP地址、三维坐标、GPS、Wi-Fi、手机信息等多种方式获取地址信息，如下图对不同获取方式的优缺点进行了比较，浏览器会自动以最优方式去获取用户地理信息。



### 隐私

HTML5 Geolocation 规范提供了一套保护用户隐私的机制。必须先得到用户明确许可，才能获取用户的位置信息。

### API详解

1、获取当前地理信息

navigator. geolocation.getCurrentPosition(successCallback, errorCallback, options)

2、重复获取当前地理信息

navigator. geolocation.watchPosition(successCallback, errorCallback, options)

当成功获取地理信息后，会调用succssCallback，并返回一个包含位置信息的对象position。

position.coords.latitude纬度

position.coords.longitude经度

position.coords.accuracy精度

position.coords.altitude海拔高度

当获取地理信息失败后，会调用errorCallback，并返回错误信息error

可选参数 options 对象可以调整位置信息数据收集方式

a) enableHighAccuracy 高精度模式

b) timeout 超时设置，单位为ms

c) maximumAge表示浏览器重新获取位置信息的时间间隔，单位为ms

### 应用

在现实开发中，通过调用第三方API（如百度地图）来实现地理定位信息，这些API都是基于用户当前位置的，并将用位置位置（经/纬度）当做参数传递，就可以实现相应的功能。

见代码示例geolocation/6-7.html

## 历史管理

提供window.history，对象我们可以管理历史记录，可用于单页面应用，Single Page Application，可以无刷新改变网页内容。

### 新增方法

1、pushState(data, title, url) 追加一条历史记录

data用于存储自定义数据，通常设为null

title网页标题，基本上没有被支持，一般设为空

url 以当前域为基础增加一条历史记录，不可跨域设置

见代码示例history/pushState.html

2、replaceState(data, title, url) 与pushState()基本相同，不同之处在于replaceState()，只是替换当前url，不会增加/减少历史记录。

见代码示例histroy/replaceState.html

### 事件监听

onpopstate事件，当前进或后退时则触发，通过事件对象ev.state可以读取到存储的数据。

见代码示例history/onpopstate.html

## Web存储

随着互联网的快速发展，基于网页的应用越来越普遍，同时也变的越来越复杂，为了满足各种各样的需求，会经常性在本地存储大量的数据，传统方式我们以document.cookie来进行存储的，但是由于其存储大小只有4k左右，并且解析也相当的复杂，给开发带来诸多不便，HTML5规范则提出解决方案。

### 特性

1、设置、读取方便

2、容量较大，sessionStorage约5M、localStorage约20M

4、只能存储字符串，可以将对象JSON.stringify() 编码后存储

### window.sessionStorage

1、生命周期为关闭浏览器窗口

2、在同一个窗口下数据可以共享

见代码示例storage/sessionStorage.html

### window.localStorage

1、永久生效，除非手动删除

2、可以多窗口共享

见代码示例storage/localStorage.html

### 方法详解

setItem(key, value) 设置存储内容

getItem(key) 读取存储内容

removeItem(key) 删除键值为key的存储内容

clear() 清空所有存储内容

key(n) 以索引值来获取存储内容

### 其它

WebSQL、IndexDB

## 应用缓存

HTML5中我们可以轻松的构建一个离线（无网络状态）应用，只需要创建一个cache manifest文件。

### 优势

1、可配置需要缓存的资源

2、网络无连接应用仍可用

3、本地读取缓存资源，提升访问速度，增强用户体验

4、减少请求，缓解服务器负担

### 缓存清单

一个普通文本文件，其中列出了浏览器应缓存以供离线访问的资源，推荐使用.appcache为后缀名，添加MIME类型

AddType text/cache-manifest .appcache

例如我们创建了一个名为demo.appcache的文件，然后在需要应用缓存在页面的根元素(html)添加属性manifest="demo.appcache"，路径要保证正确。

### manifest文件格式

1、顶行写CACHE MANIFEST

2、CACHE: 换行 指定我们需要缓存的静态资源，如.css、image、js等

3、NETWORK: 换行 指定需要在线访问的资源，可使用通配符

4、FALLBACK: 换行 当被缓存的文件找不到时的备用资源

### 事件监听

可自行查阅资料

### 其它

1、CACHE: 可以省略，这种情况下将需要缓存的资源写在CACHE MANIFEST

2、可以指定多个CACHE: NETWORK: FALLBACK:，无顺序限制

3、#表示注释，只有当demo.appcache文件内容发生改变时或者手动清除缓存后，才会重新缓存。

4、chrome 可以通过chrome://appcache-internals/工具和离线（offline）模式来调试管理应用缓存

## 多媒体

方法：load()、play()、pause()

属性：currentSrc、currentTime、duration

事件：ontimeupdate，onended等

[**参考文档**](http://www.w3school.com.cn/tags/html_ref_audio_video_dom.asp)

## 应用

1、自定义视频播放器