HZ BOOKS
华章科技

**畅销书全新升级，第1版上市后已印刷近10次，累计销量超过50000册，是该领域公认的标杆之作，销量和口碑都极佳。**

资深专家撰写，内容系统而全面，详尽讲解了HTML 5与CSS 3的所有功能和特性；注重实战，不仅包含大量辅助理解的小案例，而且还包含两个综合性案例，可操作性极强。

實戰

第2版 · 上册

陆凌牛 著

*HTML 5 and CSS 3: The Definitive Guide, Second Edition*

# HTML 5与CSS 3 权威指南

机械工业出版社
China Machine Press

# HTML 5与CSS 3权威指南

（第2版 · 上册）

陆凌牛　著



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

HTML 5与CSS 3权威指南（第2版 · 上册）/ 陆凌牛著. —北京：机械工业出版社，2013.1

ISBN 978-7-111-41247-2

Ⅰ. H… Ⅱ. 陆… Ⅲ. ①超文本标记语言 - 程序设计 - 自学参考资料 ②网页制作工具 - 自学参考资料
Ⅳ. ①TP312 ②TP393.092

中国版本图书馆CIP数据核字（2013）第015596号



第1版2年内印刷近10次，累计销量超过50000册，4大网上书店的读者评论超过4600条，98%以上的评论都是五星级的好评。不仅是HTML 5与CSS 3图书领域当之无愧的领头羊，而且在整个原创计算机图书领域也是佼佼者。本书已经成为HTML 5与CSS 3图书领域的一个标杆，被读者誉为“系统学习HTML 5与CSS 3技术的最佳指导参考书之一”和“Web前端工程师案头必备图书之一”。第2版首先从技术的角度结合最新的HTML 5和CSS 3标准对内容进行了更新和补充，其次从结构组织和写作方式的角度对原有的内容进行了进一步优化，使之更具价值且更便于读者阅读。

全书共29章，本书分为上下两册：上册（1～17章）全面系统地讲解了HTML 5相关的技术，以HTML 5对现有Web应用产生的变革开篇，顺序讲解了HTML 5与HTML 4的区别、HTML 5的结构、表单元素、HTML编辑API、图形绘制、History API、本地存储、离线应用、文件API、通信API、扩展的XML HttpRequest API、Web Workers、地理位置信息、多媒体相关的API、页面显示相关的API、拖放API与通知API等内容；下册（18～29章）全面系统地讲解了CSS 3相关的技术，以CSS 3的功能和模块结构开篇，顺序讲解了各种选择器及其使用、文字与字体的相关样式、盒相关样式、背景与边框相关样式、布局相关样式、变形处理、动画、颜色相关样式等内容。上下两册共351个示例页面，所有代码均通过作者上机调试。下册的最后有2个综合案例，以迭代的方式详细讲解了整个案例的实现过程，可操作性极强。

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）
责任编辑：陈佳媛
印刷
2013年3月第2版第1次印刷
186mm×240mm • 30.5印张
标准书号：ISBN 978-7-111-41247-2
定 价：79.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991 88361066 投稿热线：（010）88379604
购书热线：（010）68326294 88379649 68995259 读者信箱：hzjsj@hzbook.com

# 前　言

## 为何写作本书

如果要盘点2010年IT届的十大热门技术，云计算、移动开发、物联网等无疑会在其中，HTML 5肯定也是少不了的。2010年，随着HTML 5的迅猛发展，各大浏览器开发公司如Google、微软、苹果、Mozilla和Opera的浏览器开发业务都变得异常繁忙。在整个2010年度，无论是Mozilla的Firefox、Google的Chrome、苹果的Safari，还是微软的Internet Explorer，以及Opera都处于不断地推陈出新的状态当中。

2010年3月，在微软的MIX2010大会上，微软的工程师在介绍Internet Explorer 9浏览器的同时，从前端技术的角度把互联网的发展分为了三个阶段：

第一阶段：Web 1.0的以内容为主的网络，前端主流技术是HTML和CSS；

第二阶段：Web 2.0的Ajax应用，热门技术是JavaScript/DOM/异步数据请求；

第三阶段：即将迎来的HTML 5时代，亮点是富图形和富媒体内容（Graphically-Rich and Media-Rich）。

前端技术将进入一个崭新的时代，至少已经开启了这扇门。

在这种局势下，学习HTML 5无疑成为Web开发者的一个重要任务，谁先学会HTML 5，谁就掌握了迈向未来Web平台的一把钥匙。因此，我希望能够借助此书帮助国内的Web开发者更好地学习HTML 5以及与之相伴随的CSS 3技术，使他们能够早日运用这些技术开发出一个具有现代水平的、在未来的Web平台上能够流畅运行的Web网站或Web应用程序。

## 第2版与第1版的区别

自2011年上半年本书第1版出版以来，一直受到广大读者的欢迎，笔者在这里首先感谢广大读者的支持。自2011年之后，HTML 5与CSS 3标准处于不断发展中，各主流浏览器也以最快的速度对HTML 5中各种最新公布的API提供了支持，其中包括各种新增元素、IndexedDB API、FileSystem API、Server-Sent Events API、扩展的XMLHttpRequest API等。因此，本书第2版以第1版的内容为基础，添加2011年到2012年之间HTML 5中新增的各种元素及API，同时更新各主流浏览器CSS 3的最新支持情况，以使读者能够学习到截至2012年HTML 5与CSS 3标准中的各种知识，了解各种最新的浏览器对HTML 5与CSS 3标准的最新支持情况，能够早日用这些新的知识打造一个HTML 5时代的功能强大的Web网站或Web应用程序。

具体来说，第2版在第1版的基础上，做出如下主要修改：

- “第2章 HTML 5与HTML 4的区别”中新增部分元素属性。
- “第3章 HTML 5的结构”中扩充“3.3.1大纲”一节中的内容。
- “第4章 表单及其他新增和改良元素”中新增大量表单元素与页面元素，移除“4.4 文件API”一节与“4.5 拖放API”一节。
- 新增“第5章 HTML编辑API”一章（原“第5章 绘制图形”修改为第6章，原“第6章 多媒体播放”修改为“第15章 多媒体相关API”中“15.1 多媒体播放”一节）。
- 新增“第7章 HistoryAPI”一章（原“第7章 本地存储”章修改为“第8章 本地存储”，原“第8章 离线应用程序”修改为“第9章 离线应用程序”）。
- “第8章 本地存储”中新增“8.1.4 利用storage事件实时监视Web Storage中的数据”小节，新增“8.3 indexedDB数据库”一节。
- 新增“第10章 文件API”一章。
- 原“第9章 通信API”修改为“第11章 通信API”，“11.1 跨文档消息传输”中新增“11.1.3 通道通信”小节，“11.2 Web Sockets通信”中新增“11.2.5 发送与接收原始二进制数据”、“11.2.6 实现Web Sockets API的开发框架”、“11.2.7 WebSocket 协议”与“11.2.8 Web Sockets API的适用场景”等小节，新增“11.3 Server-Sent Events API”一节。
- 新增“第12章 扩展的XMLHttpRequest API”。
- 原“第10章 使用Web Workers处理线程”修改为“第13章 使用Web Workers处理线程”，新增“13.5 适用场合”一节和“13.6 SharedWorker”一节。
- 原“第11章 获取地理位置信息”修改为“第14章 获取地理位置信息”。
- 新增“第15章 多媒体相关API”一章。
- 新增“第16章 与页面显示相关的API”一章。
- 新增“第17章 拖放API与通知 API”（“17.1 拖放API”一节中内容为原“第4章 表单及其他新增和改良元素”一章中“4.5 拖放API”一节内容）。

- 原“第12章 CSS 3概述”修改为“第18章 CSS 3概述”。
- “第19章 选择器”中“19.4 UI元素状态伪类选择器”一节中新增“19.4.6 E:invalid伪类选择器与E:valid伪类选择器”小节、“19.4.7 E:required伪类选择器与E:optional伪类选择器”小节与“19.4.8 E:in-range伪类选择器与E:out-of-range伪类选择器”小节。
- “第22章 盒相关样式”与“第23章 背景与边框相关样式”的样式代码及其说明中移除部分浏览器供应商前缀。
- 重新编写“第29章 综合实例”一章中两个代码示例，“29.2 实例2：使用HTML 5+CSS 3来构建Web应用程序”中添加了使用indexedDB API将数据书写到indexedDB数据库中的代码示例部分。

## 本书面向的读者

本书主要适合如下人群阅读：

- 具有一定基础的Web前端开发工程师。
- 具有一定美术功底的Web前端设计师和UI设计师。
- Web项目的项目管理人员。
- 开设了Web开发等相关专业的高等院校师生和相关培训机构的学员及教师。

## 如何阅读本书

相较于第1版，第2版有了很大的变化，篇幅也大量增加，于是我们将第2版分成上下两册。

上册对HTML 5中新增的语法与标记方法、新增元素、新增API以及这些元素与API到目前为止受到了哪些浏览器的支持等进行了详细的介绍。在对它们进行介绍的同时将其与HTML 4中的各种元素与功能进行了对比，以帮助读者更好地理解为什么需要使用HTML 5、使用HTML 5有什么好处、HTML 5中究竟增加了哪些目前HTML 4不具备而在第3代Web平台上将会起到重要作用的功能与API，以及这些功能与API的详细使用方法。

下册详细介绍了CSS 3中各种新增样式与属性，其中主要包括CSS 3中的各种选择器、文字与字体、背景与边框、各种盒模型、CSS 3中的布局方式、CSS 3中的变形与动画、CSS 3中与媒体类型相关的一些样式与属性等。在介绍的同时也详细讲述了这些样式与属性目前为止受到了哪些浏览器的支持，以及针对各种浏览器应该怎样在样式代码中进行各种属性的正确书写。最后，详细讲解了两个实例，第一个实例展示了如何在一个用HTML 5语言编写而成的页面中综合运用HTML 5中新增的各种结构元素，如何对这些结构元素综合使用CSS 3样式；第二个实例展示了如何使用HTML 5中新增的表单元素，以及操作本地数据库的功能来实现一个具有现代风格的Web应用程序，如何在这个利用HTML 5语言及其功能编写而成的Web应用程序中综合使用CSS 3样式来完成页面的布局以及视觉效果的美化工作。

上下两册一共351个示例页面，每个页面都经过笔者上机实践，确保运行结果正确无误。每个页面的详细代码及其使用到的脚本文件、各种资源文件都可在华章公司的官方网站

（www.hzbook.com）本书的相应页面上下载，因为是由HTML 5编写而成的网页，所以可直接在各种浏览器中打开该文件并查看运行结果（少量页面需要首先建立网站，然后通过访问网站中该页面的方式来进行查看，少量页面使用服务器端PHP脚本语言，可在Apache服务器中运行）。同时，对于HTML 5中的各种元素和各种API，以及CSS 3中的各种属性和样式受到了哪些浏览器的支持在书中都进行了详细介绍，读者可以针对不同的页面选择正确的浏览器来查看其正确的运行结果。

## 致谢

在本书的写作过程中，策划编辑杨福川先生和姜影女士给予了很大的帮助和支持，并提出了很多中肯的建议，在此表示感谢。同时，还要感谢机械工业出版社的所有编审人员为本书的出版所付出的辛勤劳动。本书的成功出版是大家共同努力的结果，谢谢你们。

另外，在本书的写作过程当中，由于时间及水平上的原因，有可能存在一些对HTML 5及CSS 3上认识不全面或疏漏的地方，敬请读者批评更正，作者的联系QQ为240824399，联系邮箱为240824399@qq.com，谨以最真诚的心希望能与读者共同交流，共同成长。

陆凌牛

# 目　录

前言

## 上　册

第**1**章　**Web**时代的变迁 / 1
- 1.1　迎接新的Web时代 / 2
  - 1.1.1　HTML 5时代即将来临 / 2
  - 1.1.2　HTML 5的目标 / 4
- 1.2　HTML 5会深受欢迎的理由 / 4
  - 1.2.1　世界知名浏览器厂商对HTML 5的支持 / 4
  - 1.2.2　第一个理由：时代的要求 / 5
  - 1.2.3　第二个理由：Internet Explorer 8 / 6
- 1.3　可以放心使用HTML 5的三个理由 / 6
- 1.4　HTML 5要解决的三个问题 / 7

第**2**章　**HTML 5**与**HTML 4**的区别 / 8
- 2.1　语法的改变 / 9
  - 2.1.1　HTML 5的语法变化 / 9
  - 2.1.2　HTML 5中的标记方法 / 10
  - 2.1.3　HTML 5确保了与之前HTML版本的兼容性 / 10
  - 2.1.4　标记示例 / 12

2.2 新增的元素和废除的元素 / 12
2.2.1 新增的结构元素 / 12
2.2.2 新增的其他元素 / 14
2.2.3 新增的input元素的类型 / 18
2.2.4 废除的元素 / 18
2.3 新增的属性和废除的属性 / 19
2.3.1 新增的属性 / 19
2.3.2 废除的属性 / 21
2.4 全局属性 / 22
2.4.1 contentEditable属性 / 22
2.4.2 designMode属性 / 24
2.4.3 hidden属性 / 24
2.4.4 spellcheck属性 / 24
2.4.5 tabindex属性 / 25
第**3**章 **HTML 5**的结构 / 26
3.1 新增的主体结构元素 / 27
3.1.1 article元素 / 27
3.1.2 section元素 / 29
3.1.3 nav元素 / 31
3.1.4 aside元素 /32
3.1.5 time元素与微格式 / 34
3.1.6 pubdate属性 / 35
3.2 新增的非主体结构元素 / 35
3.2.1 header元素 / 36
3.2.2 hgroup元素 / 37
3.2.3 footer元素 / 37
3.2.4 address元素 / 38
3.3 HTML 5结构 / 39
3.3.1 大纲 / 39
3.3.2 大纲的编排规则 / 44
3.3.3 对新的结构元素使用样式 / 47
第**4**章 表单及其他新增和改良元素 / 48
4.1 新增元素与属性 / 49
4.1.1 新增属性 / 49

4.1.2 大幅度地增加与改良input元素的种类 / 61
4.1.3 对新的表单元素使用样式 / 71
4.1.4 output元素的追加 / 72
4.2 表单验证 / 73
4.2.1 自动验证 / 73
4.2.2 取消验证 / 73
4.2.3 显式验证 / 74
4.3 增强的页面元素 / 74
4.3.1 新增的figure元素与figcaption元素 / 75
4.3.2 新增的details元素与summary元素 / 76
4.3.3 新增的mark元素 / 77
4.3.4 新增的progress元素 / 80
4.3.5 新增的meter元素 / 81
4.3.6 改良的ol列表 / 82
4.3.7 改良的dl列表 / 83
4.3.8 加以严格限制的cite元素 / 85
4.3.9 重新定义的small元素 / 85
4.3.10 安全性增强的iframe元素 / 86
4.3.11 增强的script元素 / 89
第5章 **HTML编辑API** / 94
5.1 Range对象与Selection对象 / 95
5.1.1 基本概念 / 95
5.1.2 Range对象的属性与方法 / 97
5.1.3 Selection对象的属性与方法 / 118
5.2 命令 / 128
5.2.1 基本概念 / 128
5.2.2 execCommand方法 / 128
5.2.3 queryCommandSupported方法 / 129
5.2.4 queryCommandState方法 / 130
5.2.5 queryCommandIndeterm方法 / 131
5.2.6 queryCommandEnabled方法 / 133
5.2.7 queryCommandValue方法 / 135
5.2.8 可以在各种浏览器中运行的所有命令 / 136

第**6**章　绘制图形 / 136
6.1　canvas元素的基础知识 / 138
6.1.1　在页面中放置canvas元素 / 138
6.1.2　绘制矩形 / 139
6.2　使用路径 / 141
6.2.1　绘制圆形 / 141
6.2.2　如果没有关闭路径会怎么样 / 143
6.2.3　moveTo与lineTo / 145
6.2.4　使用bezierCurveTo绘制贝济埃曲线 / 146
6.3　绘制渐变图形 / 148
6.3.1　绘制线性渐变 / 148
6.3.2　绘制径向渐变 / 150
6.4　绘制变形图形 / 151
6.4.1　坐标变换 / 151
6.4.2　坐标变换与路径的结合使用 / 153
6.4.3　矩阵变换 / 154
6.5　图形组合 / 158
6.6　给图形绘制阴影 / 160
6.7　使用图像 / 162
6.7.1　绘制图像 / 162
6.7.2　图像平铺 / 164
6.7.3　图像裁剪 / 166
6.7.4　像素处理 / 168
6.8　绘制文字 / 170
6.9　补充知识 / 172
6.9.1　保存与恢复状态 / 172
6.9.2　保存文件 / 173
6.9.3　简单动画的制作 / 174
第**7**章　**History API** / 177
7.1　History API的基本概念 / 178
7.2　History API使用示例 / 179
7.2.1　使用History API / 179
7.2.2　结合使用Canvas API与History API / 188

第**8**章　本地存储 / 193
8.1　Web Storage / 194
8.1.1　Web Storage概述 / 194
8.1.2　简单Web留言本 / 197
8.1.3　作为简易数据库来利用 / 200
8.1.4　利用storage事件实时监视Web Storage中的数据 / 202
8.2　本地数据库 / 204
8.2.1　本地数据库的基本概念 / 204
8.2.2　用executeSql来执行查询 / 205
8.2.3　使用数据库实现Web留言本 / 206
8.2.4　transaction方法中的处理 / 209
8.3　indexedDB数据库 / 211
8.3.1　indexedDB数据库的基本概念 / 211
8.3.2　连接数据库 / 211
8.3.3　数据库的版本更新 / 213
8.3.4　创建对象仓库 / 218
8.3.5　创建索引 / 222
8.3.6　索引的multiEntry属性值 / 226
8.3.7　使用事务 / 227
8.3.8　保存数据 / 229
8.3.9　获取数据 / 232
8.3.10　根据主键值检索数据 / 235
8.3.11　根据索引属性值检索数据 / 241
8.3.12　复合索引 / 246
8.3.13　统计对象仓库中的数据数量 / 250
8.3.14　使用indexedDB API制作Web留言本 / 252
第**9**章　离线应用程序 / 259
9.1　离线Web应用程序详解 / 260
9.1.1　新增的本地缓存 / 260
9.1.2　本地缓存与浏览器网页缓存的区别 / 260
9.2　manifest文件 / 261
9.3　浏览器与服务器的交互过程 / 263
9.4　applicationCache对象 / 264
9.4.1　swapCache方法 / 265
9.4.2　applicationCache对象的事件 / 267

第**10**章　文件**API** / 270
10.1　FileList对象与file对象 / 271
10.2　ArrayBuffer对象与ArrayBufferView对象 / 272
10.2.1　基本概念 / 272
10.2.2　ArrayBuffer对象 / 272
10.2.3　ArrayBufferView对象 / 273
10.2.4　DataView对象 / 274
10.3　Blob对象与BlobBuilder对象 / 278
10.3.1　Blob对象 / 278
10.3.2　BlobBuilder对象 / 280
10.3.3　Blob对象的slice方法 / 283
10.4　FileReader对象 / 284
10.4.1　FileReader对象的方法 / 284
10.4.2　FileReader对象的事件 / 285
10.4.3　FileReader对象的使用示例 / 285
10.5　FileSystem API / 292
10.5.1　FileSystem API概述 / 292
10.5.2　FileSystem API的适用场合 / 292
10.5.3　请求访问文件系统 / 293
10.5.4　申请磁盘配额 / 296
10.5.5　创建文件 / 300
10.5.6　写入文件 / 302
10.5.7　在文件中追加数据 / 305
10.5.8　读取文件 / 307
10.5.9　复制磁盘中的文件 / 309
10.5.10　删除文件 / 311
10.5.11　创建目录 / 312
10.5.12　读取目录中的内容 / 316
10.5.13　删除目录 / 319
10.5.14　复制文件或目录 / 321
10.5.15　移动文件或目录与重命名文件或目录 / 323
10.5.16　filesystem:URL前缀 / 326
10.5.17　综合案例 / 328

10.6 Base64编码支持 / 336
10.6.1 Base64编码概述 / 336
10.6.2 在HTML 5中支持Base64编码 / 338
第**11**章 通信**API** / 342
11.1 跨文档消息传输 / 343
11.1.1 跨文档消息传输的基本知识 / 343
11.1.2 跨文档消息传输示例 / 344
11.1.3 通道通信 / 346
11.2 WebSockets通信 / 350
11.2.1 WebSockets通信的基本知识 / 350
11.2.2 使用WebSockets API / 350
11.2.3 WebSockets API使用示例 / 351
11.2.4 发送对象 / 353
11.2.5 发送与接收原始二进制数据 / 354
11.2.6 实现WebSockets API的开发框架 / 355
11.2.7 WebSocket 协议 / 355
11.2.8 WebSockets API的适用场景 / 356
11.3 Server-Sent Events API / 356
11.3.1 Server-Sent Events API的基本概念 / 356
11.3.2 Server-Sent Events API的实现方法 / 356
11.3.3 事件ID的使用示例 / 363
第**12**章 扩展的**XMLHttpRequest API** / 366
12.1 从服务器端获取二进制数据 / 367
12.1.1 ArrayBuffer响应 / 368
12.1.2 Blob响应 / 373
12.2 发送数据 / 374
12.2.1 发送字符串 / 374
12.2.2 发送表单数据 / 376
12.2.3 上传文件 / 378
12.2.4 发送Blob对象 / 379
12.2.5 发送ArrayBuffer对象 / 381
12.3 跨域数据请求 / 385

第**13**章　使用**Web Workers**处理线程 / 388
13.1　基础知识 / 389
13.2　与线程进行数据的交互 / 392
13.3　线程嵌套 / 394
13.3.1　单层嵌套 / 395
13.3.2　在多个子线程中进行数据的交互 / 397
13.4　线程中可用的变量、函数与类 / 398
13.5　适用场合 / 399
13.6　SharedWorker / 399
13.6.1　基础知识 / 399
13.6.2　实现前台页面与后台线程之间的通信 / 400
13.6.3　定义页面与共享的后台线程开始通信时的处理 / 400
13.6.4　SharedWorker的使用示例 / 401
第**14**章　获取地理位置信息 / 406
14.1　Geolocation API的基本知识 / 407
14.1.1　取得当前地理位置 / 407
14.1.2　持续监视当前地理位置的信息 / 409
14.1.3　停止获取当前用户的地理位置信息 / 409
14.2　position对象 / 409
14.3　在页面上使用google地图 / 411
第**15**章　多媒体相关**API** / 414
15.1　多媒体播放 / 415
15.1.1　video元素与audio元素的基础知识 / 415
15.1.2　属性 / 417
15.1.3　方法 / 421
15.1.4　事件 / 423
15.2　Web Audio API / 426
15.2.1　AudioContext对象 / 426
15.2.2　加载声音 / 427
15.2.3　播放声音 / 428
15.2.4　将声音加载处理封装在类中 / 429
15.2.5　控制节奏 / 431
15.2.6　控制音量 / 433

15.2.7 两个声音的交叉混合 / 436
15.2.8 多个音频文件之间的平滑过渡 / 439
15.2.9 对音频使用滤波处理 / 443
第**16**章 与页面显示相关的**API** / 447
16.1 Page Visibility API / 448
16.1.1 Page Visibility API概述 / 448
16.1.2 Page Visibility API的使用场合 / 448
16.1.3 实现Page Visibility API / 448
16.2 Fullscreen API / 451
16.2.1 Fullscreen API概述 / 451
16.2.2 实现Fullscreen API / 451
16.2.3 Fullscreen API代码使用示例 / 454
第**17**章 拖放**API**与通知**API** / 457
17.1 拖放API / 458
17.1.1 实现拖放的步骤 / 458
17.1.2 DataTransfer对象的属性与方法 / 461
17.1.3 设定拖放时的视觉效果 / 461
17.1.4 自定义拖放图标 / 462
17.2 通知API / 463
17.2.1 通知API的基础知识 / 463
17.2.2 通知API的代码使用示例 / 465

## 下 册

第**18**章 **CSS 3**概述 / 467
18.1 概要介绍 / 468
18.1.1 CSS 3是什么 / 468
18.1.2 CSS 3的历史 / 468
18.2 使用CSS 3能做什么 / 469
18.2.1 模块与模块化结构 / 469
18.2.2 一个简单的CSS 3示例 / 470
第**19**章 选择器 / 473
19.1 选择器概述 / 475
19.2 属性选择器 / 475

19.2.1 属性选择器概述 / 475
19.2.2 CSS3中的属性选择器 / 477
19.2.3 灵活运用属性选择器 / 478
19.3 结构性伪类选择器 / 479
19.3.1 CSS中的伪类选择器及伪元素 / 479
19.3.2 选择器root、not、empty和target / 483
19.3.3 选择器：first-child、last-child、nth-child和nth-last-child / 488
19.3.4 选择器：nth-of-type和nth-last-of-type / 492
19.3.5 循环使用样式 / 494
19.3.6 only-child选择器 / 496
19.4 UI元素状态伪类选择器 / 498
19.4.1 选择器：E:hover、E:active和E:focus / 498
19.4.2 E:enabled伪类选择器与E:disabled伪类选择器 / 500
19.4.3 E:read-only伪类选择器与E:read-write伪类选择器 / 501
19.4.4 伪类选择器：E:checked、E:default和E:indeterminate / 502
19.4.5 E::selection伪类选择器 / 505
19.4.6 E:invalid伪类选择器与E:valid伪类选择器 / 506
19.4.7 E:required伪类选择器与E:optional伪类选择器 / 507
19.4.8 E:in-range伪类选择器与E:out-of-range伪类选择器 / 508
19.5 通用兄弟元素选择器 / 509
第**20**章 使用选择器在页面中插入内容 / 511
20.1 使用选择器来插入文字 / 512
20.1.1 使用选择器来插入内容 / 512
20.1.2 指定个别元素不进行插入 / 513
20.2 插入图像文件 / 514
20.2.1 在标题前插入图像文件 / 514
20.2.2 插入图像文件的好处 / 515
20.2.3 将alt属性的值作为图像的标题来显示 / 517
20.3 使用content属性来插入项目编号 / 518
20.3.1 在多个标题前加上连续编号 / 518
20.3.2 在项目编号中追加文字 / 519
20.3.3 指定编号的样式 / 519
20.3.4 指定编号的种类 / 519
20.3.5 编号嵌套 / 520

20.3.6 中编号中嵌入大编号 / 521
20.3.7 在字符串两边添加嵌套文字符号 / 523
第**21**章 文字与字体相关样式 / 525
21.1 给文字添加阴影 —— text-shadow属性 / 526
21.1.1 text-shadow属性的使用方法 / 526
21.1.2 位移距离 / 528
21.1.3 阴影的模糊半径 / 528
21.1.4 阴影的颜色 / 529
21.1.5 指定多个阴影 / 529
21.2 让文本自动换行 —— word-break属性 / 530
21.2.1 依靠浏览器让文本自动换行 / 530
21.2.2 指定自动换行的处理方法 / 530
21.3 让长单词与URL地址自动换行 —— word-wrap属性 / 532
21.4 使用服务器端字体 —— Web Font与@font-face属性 / 532
21.4.1 在网页上显示服务器端字体 / 532
21.4.2 定义斜体或粗体字体 / 534
21.4.3 显示客户端本地的字体 / 536
21.4.4 属性值的指定 / 537
21.5 修改字体种类而保持字体尺寸不变 —— font-size-adjust属性 / 538
21.5.1 字体不同导致文字大小的不同 / 538
21.5.2 font-size-adjust属性的使用方法 / 540
21.5.3 浏览器对于aspect值的计算方法 / 540
21.5.4 font-size-adjust属性的使用示例 / 541
第**22**章 盒相关样式 / 543
22.1 盒的类型 / 544
22.1.1 盒的基本类型 / 544
22.1.2 inline-block类型 / 545
22.1.3 inline-table类型 / 552
22.1.4 list-item类型 / 554
22.1.5 run-in类型与compact类型 / 555
22.1.6 表格相关类型 / 556
22.1.7 none类型 / 558
22.1.8 各种浏览器对于各种盒类型的支持情况 / 559
22.2 对于盒中容纳不下的内容的显示 / 560

22.2.1 overflow属性 / 560
22.2.2 overflow-x属性与overflow-y属性 / 562
22.2.3 text-overflow属性 / 563
22.3 对盒使用阴影 / 565
22.3.1 box-shadow属性的使用方法 / 565
22.3.2 将参数设定为0 / 566
22.3.3 对盒内子元素使用阴影 / 567
22.3.4 对第一个文字或第一行使用阴影 / 568
22.3.5 对表格及单元格使用阴影 / 568
22.4 指定针对元素的宽度与高度的计算方法 / 570
22.4.1 box-sizing属性 / 570
22.4.2 为什么要使用box-sizing属性 / 572
第**23**章 背景与边框相关样式 / 574
23.1 与背景相关的新增属性 / 575
23.1.1 指定背景的显示范围 —— background-clip属性 / 575
23.1.2 指定绘制背景图像的绘制起点 —— background-origin属性 / 577
23.1.3 指定背景图像的尺寸 —— background-size属性 / 579
23.2 在一个元素中显示多个背景图像 / 582
23.3 圆角边框的绘制 / 583
23.3.1 border-radius属性 / 583
23.3.2 在border-radius属性中指定两个半径 / 584
23.3.3 不显示边框时 / 585
23.3.4 修改边框种类时 / 585
23.3.5 绘制四个角不同半径的圆角边框 / 585
23.4 使用图像边框 / 586
23.4.1 border-image属性 / 586
23.4.2 border-image属性的最简单的使用方法 / 587
23.4.3 使用border-image属性来指定边框宽度 / 589
23.4.4 中央图像的自动拉伸 / 590
23.4.5 指定四条边中图像的显示方法 / 591
23.4.6 使用背景图像 / 594
第**24**章 **CSS 3**中的变形处理 / 596
24.1 transform功能的基础知识 / 597
24.1.1 如何使用transform功能 / 597

24.1.2 transform功能的分类 / 598
24.2 对一个元素使用多种变形的方法 / 602
24.2.1 两个变形示例 / 602
24.2.2 指定变形的基准点 / 604
第**25**章 **CSS 3**中的动画功能 / 607
25.1 Transitions功能 / 608
25.1.1 Transitions功能的使用方法 / 608
25.1.2 使用Transitions功能同时平滑过渡多个属性值 / 609
25.2 Animations功能 / 612
25.2.1 Animations功能的使用方法 / 612
25.2.2 实现多个属性值同时改变的动画 / 614
25.2.3 实现动画的方法 / 616
25.2.4 实现网页的淡入效果 / 618
第**26**章 布局相关样式 / 619
26.1 多栏布局 / 620
26.1.1 使用float属性或position属性的缺点 / 620
26.1.2 使用多栏布局方式 / 622
26.2 盒布局 / 625
26.2.1 盒布局的基础知识 / 625
26.2.2 弹性盒布局 / 629
第**27**章 **Media Queries**相关样式 / 641
27.1 根据浏览器的窗口大小来选择使用不同的样式 / 642
27.2 在iPhone中的显示 / 646
27.3 Media Queries的使用方法 / 647
第**28**章 **CSS 3**的其他重要样式和属性 /650
28.1 颜色相关样式 / 651
28.1.1 利用alpha通道来设定颜色 / 651
28.1.2 alpha通道与opacity属性的区别 / 653
28.1.3 指定颜色值为transparent / 655
28.2 用户界面相关样式 / 656
28.2.1 轮廓相关样式 / 656
28.2.2 resize属性 / 659
28.3 取消对元素的样式指定 —— initial属性值 / 660

28.3.1 取消对元素的样式指定 / 660
28.3.2 使用initial属性值并不等于取消样式设定的特例 / 662
第**29**章 综合实例 / 664
29.1 实例1：使用HTML 5中新增结构元素来构建网页 / 665
29.1.1 组织网页结构 / 665
29.1.2 构建网页标题 / 667
29.1.3 构建侧边栏 / 670
29.1.4 构建主体内容 / 672
29.1.5 构建版权信息 / 678
29.2 实例2：使用HTML 5+CSS 3来构建Web应用程序 / 678
29.2.1 HTML 5页面代码分析 / 679
29.2.2 CSS 3样式代码分析 / 682
29.2.3 JavaScript脚本代码分析 / 685
附录**A** 可以在各种浏览器中运行的所有命令 / 698
附录**B** **2012**年**7**月五大浏览器的最新版对**HTML 5**的支持情况 / 706

# 第1章 Web时代的变迁

1.1 迎接新的Web时代
1.2 HTML 5会深受欢迎的理由
1.3 可以放心使用HTML 5的三个理由
1.4 HTML 5要解决的三个问题

自从2010年HTML 5正式推出以来，它立刻受到了世界各大浏览器的热烈欢迎与支持。根据世界上各大IT界知名媒体评论，新的Web时代，HTML 5的时代马上就要到来。本章重点介绍什么是HTML 5，HTML 5产生的时代背景，为什么HTML 5会如此深受业界欢迎，以及HTML能够解决什么问题。

学习内容：

- ❑ 初步了解什么是HTML 5，HTML 5与之前版本的HTML大致上有哪些区别。
- ❑ 了解世界各大知名浏览器目前的发展策略，为什么它们都不约而同地把支持HTML 5当成目前的工作重点，就连微软也把全面支持HTML 5作为新版Internet Explorer 9（IE 9）浏览器的开发重点与主要宣传手段。
- ❑ 了解为什么说开发者今后可以放心大胆地使用HTML 5进行Web网站与Web应用程序的开发，HTML 5被正式推广以后之前的Web网站与Web应用程序怎么办。
- ❑ 了解使用HTML 5到底可以解决哪些问题。

## 1.1 迎接新的Web时代

### 1.1.1 HTML 5时代即将来临

自从2010年HTML 5正式推出以来，它就以一种惊人的速度被迅速推广着，就连微软也因此为下一代IE 9做了标准上的改进，使其能够支持HTML 5。关于各主流浏览器对于HTML 5所表现出来的热烈欢迎、积极支持的详细情况，以及为什么HTML 5会如此受欢迎，我们将在后面几节中详细介绍，这里，笔者要告诉大家的是，目前业界全体都步调一致地朝着HTML 5的方向迈进着，HTML 5的时代马上就要到来了。

在全面介绍HTML 5的相关知识之前，我们先来认识一下HTML 5中的部分代码，对HTML 5有个初步的了解。

首先，我们来看一段HTML 4中常见的JavaScript代码，如代码清单1-1所示。

代码清单1-1 HTML 4中的JavaScript代码示例

```
<form>
<p><label>Username:<input name=search type="text" id="search"></label></p>
<script type="text/javascript">
       document.getElementById ('search').focus()
</script>
</form>
```

在HTML 5中，这段代码将会以怎样的形式出现呢？具体如代码清单1-2所示。

代码清单1-2 用HTML 5实现代码清单1-1中的JavaScript代码

```
<form>
<p><label>Search:<input name=search autofocus></label></p>
</form>
```

我们来看一下在HTML 4中常见的一种页面结构，代码如代码清单1-3所示。

代码清单1-3　div标签示例（用HTML 4实现）

```
<div id="header">...</div>
<div id="nav">...</div>
<div class="article">
</div>
<div id="side-bar">...</div>
<div id="footer">...</div>
```

页面中有关该部分的结构示意图如图1-1所示。

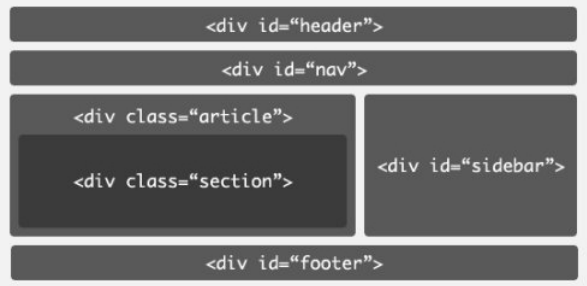


图1-1　HTML 4中的页面结构

那么，在HTML 5中，又会用怎样的页面代码来描述这种结构呢？具体如代码清单1-4所示。

代码清单1-4　HTML 5中的新型结构示例

```
<header>...</header>
<nav>...</nav>
<article>
</article>
<aside>...</aside>
<footer>...</footer>
```

页面中有关该部分的结构示意图如图1-2所示。

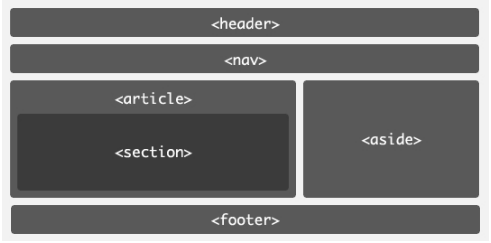


图1-2　HTML 5中的页面结构

怎么样？看出区别来了吗？在第一个示例中，我们可以看见，在HTML 4中的一段JavaScript代码，在HTML 5中消失了，取而代之的是一个在HTML 5中新出现的属性。在第二个示例中，我们可以看见，在HTML 4中常见的用div来划分页面结构的方法，到了HTML 5中，也被一种HTML 5中新出现的标签给替代了。那么究竟为什么HTML 5要对

HTML 4的脚本以及页面代码做这种修改呢？做这种修改的目的又是什么呢？

### 1.1.2 HTML 5的目标

HTML 5的目标是为了能够创建更简单的Web程序，书写出更简洁的HTML代码。例如，为了使Web应用程序的开发变得更容易，提供了很多API；为了使HTML变得更简洁，开发出了新的属性、新的元素，等等。总体来说，为下一代Web平台提供了许许多多新的功能。

那么让我们先来初步接触一下在HTML 5中究竟提供了哪些革命性的新功能。在第2章中，我们会针对这些功能做一个全面介绍。

首先，在HTML 5之前，有很多功能必须要使用JavaScript等脚本语言才能实现，譬如前面例子中提到，登录画面中经常使用的让文本框获得光标焦点的功能。如果使用HTML 5，同样的功能只要使用元素的属性标签就可以实现了。这样的话，整个页面就变得非常清楚直观，容易理解。因此，Web设计者可以非常放心大胆地使用这些HTML 5中新增的属性标签。由于HTML 5中提供了大量的这种可以替代脚本的属性标签，使得开发出来的界面语言也变得更加简洁易懂。

不但如此，HTML 5使页面结构也变得清楚明了。之前使用的div标签也不再使用了，而是使用前面HTML 5示例中所提到的更加语义化的结构标签。这样的话，书写出来的界面结构显得非常清晰，各部位要展示什么内容也让人一目了然。

虽然HTML 5宣称的立场是“非革命性的发展”，但是它所带来的功能是让人渴望的，使用它所进行的设计也是很简单的，因此，它深受Web设计者与Web开发者的欢迎。

## 1.2 HTML 5会深受欢迎的理由

### 1.2.1 世界知名浏览器厂商对HTML 5的支持

HTML 5被说成是划时代也好，具有革命性也好，如果不能被业界承认并且大面积地推广使用，这些都是没有意义的。事实上，今后HTML 5被正式地、大规模地投入应用的可能性是相当高的。

通过对Internet Explore、Google、Firefox、Safari、Opera等主要的Web浏览器的发展策略的调查，发现它们都在支持HTML 5上采取了措施。

- 微软：2010年3月16日，微软于拉斯维加斯市举行的MIX10技术大会上宣布已推出IE9浏览器开发者预览版。微软称，IE9完成开发后，将更多支持CSS 3、SVG和HTML 5等互联网浏览通用标准。
- Google：2010年2月19日，谷歌Gears项目经理伊安 - 费特通过博客宣布，谷歌将放弃对Gears浏览器插件项目的支持，以此重点开发HTML 5项目。据费特表示，目前，在谷歌看来，Gears面临的主要问题是，该应用与HTML 5的诸多创新非常相似，而且谷歌一直积极发展HTML 5项目。因此，只要谷歌不断以加强新网络标准的应用功能为工作重点，那么为Gears增加新功能就无太大意义了。目前，多种浏览器将会越来越

多地为GMail及其他服务提供更多脱机功能方面的支持，因此Gears面临的需求也在日益下降，这是谷歌做出上述调整的重要原因。

- 苹果：2010年6月7日，苹果在开发者大会的会后发布了Safari 5，这款浏览器支持10个以上的HTML 5新技术，包括全屏幕播放、HTML 5视频、HTML 5地理位置、HTML 5切片元素、HTML 5的可拖动属性、HTML 5的形式验证、HTML 5的Ruby、HTML 5的AJAX历史和WebSocket字幕。
- Opera：2010年5月5日，Opera软件公司首席技术官Hakon Wium Lie先生在访华之际，接受了中国软件资讯网等少数几家媒体的采访。号称“CSS之父”的Hakon Wium Lie认为，HTML 5与CSS 3将是全球互联网发展的未来趋势，目前包括Opera在内的诸多浏览器厂商，纷纷在研发HTML 5相关产品，Web的未来属于HTML 5。
- Mozilla：2010年7月，Mozilla基金会发布了即将推出的Firefox 4浏览器的第一个早期测试版。在该版本中的Firefox浏览器中进行了大幅改进，包括新的HTML 5语法分析器，以及支持更多HTML 5形式的控制等。从官方文档来看，Firefox 4对HTML 5是完全级别的支持。目前包括在线视频、在线音频等多种应用都已在该版中实现。

以上证据表明，目前这些浏览器都纷纷地朝着支持HTML 5、结合HTML 5的方向迈进着，因此HTML 5已经被广泛地推行开来了。为什么HTML 5会如此受欢迎，理由如1.2.2节和1.2.3节所示。

### 1.2.2 第一个理由：时代的要求

现在的时代已经迫切地要求有一个统一的互联网通用标准。HTML 5之前的情况是，由于各浏览器之间的不统一，光是修改Web浏览器之间的由于兼容性而引起的bug就浪费了大量时间。而HTML 5的目标就是将Web带入一个成熟的应用平台，在HTML 5平台上，视频、音频、图像、动画，以及同电脑的交互都被标准化。

关于Web浏览器，网页标准计划小组设计并推出了Acid3测试，它是针对网页浏览器及设计软件之标准相容性的一项测试。它针对Web应用程序中使用着的动态内容进行检查，测试焦点主要集中在ECMAScript、DOM Level 3、Media Queries和data: URL。

Acid3测试推出后，各大浏览器都认真接受了它的测试并希望能够获得比较高的分数。这个测试的设计者，正是在W3C开发及设计者，HTML 5的重要人物Ian Hickson。Ian Hickson是WHATWG（Web Hypertext Application Technology Working Group）开发团体的成员，担任Web标准规格的设计，现在是W3C的HTML 5工作组的负责人之一。

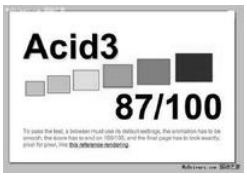

图1-3 Acid3测试图

Ian Hickson设计Acid3测试的意图是给声称“让开发者能够什么都不必担心，可以放心大胆地进行开发”的各大Web浏览器提供一个机会，让他们能够以此来证明自己是优秀的。Acid3的宣传是很重要的，要想扩大Web浏览器的市场份额，宣称遵从它所依赖的标准是最有效的宣传方法。图1-3为Acid3的一个测试图。

### 1.2.3 第二个理由：Internet Explorer 8

Internet Explorer也积极地朝着支持HTML 5的方向迈进着。Internet Explorer对此十分重视。虽然它的使用者依然很多，但是由于最近被Firefox等其他Web浏览器抢去了很多市场份额，它很不甘心。于是继Internet Explorer 7（IE 7）的发表后不久，立刻推出了Internet Explorer 8（IE 8）的Release版。

新推出的IE 8宣称遵从互联网通用标准。虽然其他的浏览器由于标榜遵从该标准而获得了很多市场份额，但是Internet Explorer肯定是要对此采取强有力的对策的。因此Internet Explorer把宣称遵从互联网通用标准看成了很重要的一件事，并且开始在IE 8里支持HTML 5。

例如，HTML 5中代替Cookie的sessionStorage功能与globalStorage功能在IE 8里都获得了支持。使用Ajax时如果点击返回按钮也可以真正让操作返回了（在IE 7中点击返回按钮，画面跳转到其他画面）。很多Internet Explorer自己独特的处理方法与特性，今后也会有所改变。

因为现在市场份额最高的Internet Explorer也在针对HTML 5做出积极对应，微软也对新的互联网通用标准表示了赞同和支持，所以可以说HTML 5在市场上大面积推广的势头是非常强的。

## 1.3 可以放心使用HTML 5的三个理由

Web开发者最担心的是新技术推出时由于其不成熟所产生的问题。如果能够实现互联网通用标准，可以避免各浏览器之间的不统一，这一点已经被明确了，但是在朝着这方面前进的过程中会不会出现什么周折是令人担心的。

虽然Web开发者普遍认为有了HTML 5是比较好的，但是还是会很担心诸如“它在老版本的浏览器上也能正常运行吗？”,“会不会产生错误？”等各种问题。但是可以很高兴地告诉你，请放心，HTML 5就像以前CSS刚开始普及时一样不会存在什么问题。

有三个理由证明可以放心使用HTML 5：

- ❑ 兼容性：HTML 5在老版本的浏览器上也可以正常运行。
- ❑ 实用性：HTML 5内部并没有封装什么很复杂的、不切实际的功能，而只是封装了简单实用的功能。
- ❑ 非革命性的发展：HTML 5的内部功能不是革命性的，只是发展性的。

以上三点就是所谓的“HTML设计原则”，HTML 5也是以该设计原则为基本原则而开发出来的，各主流浏览器使用HTML 5的前提也就是要求HTML 5能够符合这些原则，今后也将以其为前提来支持HTML 5。下面针对这些原则进行介绍。

首先是兼容性问题。虽然到了HTML 5时代，但并不代表现在用HTML 4创建出来的网站必须全部要重建，只会要求各Web浏览器今后能正常运行用HTML 5开发出来的功能。“非革命性的发展”这一点正是通过兼容性体现出来的。正是因为保障了兼容性才能让人毫不犹豫地用HTML 5来开发网站。

接着是实用性。实用性是指要求能够解决实际问题。HTML 5内只封装了切实有用的功

能，不封装复杂而没有实际意义的功能。

通过以上列举的HTML设计原则，尤其是与HTML 4相兼容的部分，基本上可以让人放下心来，大胆地使用HTML 5。

## 1.4　HTML 5要解决的三个问题

HTML 5的出现，对于Web来说意义是非常重大的。因为它的意图是想要把目前Web上存在的各种问题一并解决掉，它是一个企图心比较强的HTML版本。

那么，到底Web上存在哪些问题，HTML 5又打算怎么解决呢？

❑ Web浏览器之间的兼容性很低。

首先要提到的就是，Web浏览器之间的兼容性是非常低的。在某个Web浏览器上可以正常运行的HTML/CSS/JavaScript等Web程序，在另一个Web浏览器上就不正常了的事情是非常多的。

如果用一句话来描述这个问题的原因，可以说是“规范不统一”。规范不统一，没有被标准化，是这个问题的主要原因。

在HTML 5中，这个问题将得到解决。HTML 5的使命是详细分析各Web浏览器所具有的功能，然后以此为基础，要求这些浏览器所有内部功能都要符合一个通用标准。

如果各浏览器都符合通用标准，然后以该标准为基础来书写程序，那么程序在各浏览器都能正常运行的可能性就大大提高了，这对于Web开发者和Web设计者都是一件令人可喜的事情。而且，今后开发者开发出来的Web功能只要符合通用标准，Web浏览器也都是很愿意封装该功能的。

❑ 文档结构不够明确。

第二个问题是，在之前的HTML版本中，文档的结构不够清晰、明确。例如，为了要表示“标题”，“正文”，之前一般都是用 元素。但是，严格说来， 不是一个能把文档结构表达得很清楚的元素，使用了过多的 要素的文章，阅读时不仔细研究，是很难看出文档结构的。而且，对于搜索引擎或屏幕阅读器等程序来说，过多使用了div元素，那么这些程序就连“从哪到哪算是重要的正文”，“这个<ul>要素是表示导航菜单，还是表示项目列表”等对于结构分析来说最基本的问题的答案也都不知道。

在HTML 5中，为了解决这个问题，追加了很多跟结构相关的元素。不仅如此，还结合了包括微格式、无障碍应用在内的各种各样的周边技术。

❑ Web应用程序的功能受到了限制。

最后一个问题是，HTML与Web应用程序的关系十分薄弱。Web应用程序的特征是先从网络下载，然后忠实运行，因此应该对会威胁到用户安全的功能进行限制。

目前安全性的保障这方面已做到了，但对于Web应用程序来说，一直以来HTML真正所做出的贡献是很少的，譬如说就连上传文件时想同时选择一个以上的文件都做不到。

为了弥补这方面的不足，HTML 5已经开始提供各种各样Web应用上的新API，各浏览器也在快速地封装着这些API，HTML 5已经使富Web应用的实现变成了可能。

# 第2章
# HTML 5与HTML 4的区别

2.1 语法的改变
2.2 新增的元素和废除的元素
2.3 新增的属性和废除的属性
2.4 全局属性

HTML 5以HTML 4为基础，对HTML 4进行了大量的修改。本章从总体上概要介绍到底HTML 5对HTML 4进行了哪些修改，HTML 5与HTML 4之间比较大的区别是什么。

学习内容：

- 掌握HTML 5与HTML 4在基本语法上有什么区别，这个基本语法区别包括DOCTYPE声明、内容类型（ContentType）、字符编码的指定方法、元素标记的省略、具有布尔值的属性、引号的省略等几个方面。
- 了解在HTML 5中新增了哪些元素，删除了哪些HTML 4中的元素，为什么要删除这些元素，用什么元素或方法来取代这些被删除的元素。
- 了解在HTML 5中新增了哪些属性，删除了哪些HTML 4中的属性，在HTML 5中用什么方法来取代这些被删除的属性。
- 掌握什么是全局属性，掌握本章中介绍的几个常用全局属性，它们是contentEditable属性、designMode属性、hidden属性、spellcheck属性，以及tabindex属性。

## 2.1　语法的改变

### 2.1.1　HTML 5的语法变化

与HTML 4相比，HTML 5在语法上发生了很大的变化。可能有很多人会有疑问，“之前的HTML已经相当普及了！”，“如果改变基础语法，会产生什么影响？”等。

但是，HTML 5中的语法变化，与其他开发语言中的语法变化在根本意义上有所不同。它的变化，正是因为在HTML 5之前几乎没有符合标准规范的Web浏览器！

HTML的语法是在SGML（Standard Generalized Markup Language）语言的基础上建立起来的。但是SGML语法非常复杂，要开发能够解析SGML语法的程序也很不容易，所以很多浏览器都不包含SGML的分析器。因此，虽然HTML基本上遵从SGML的语法，但是对于HTML的执行在各浏览器之间并没有一个统一的标准。

在这种情况下，各浏览器之间的互兼容性和互操作性在很大程度上取决于网站或网络应用程序的开发者们在开发上所做的共同努力，而浏览器本身始终是存在缺陷的。

如上所述，在HTML 5中提高Web浏览器之间的兼容性是它的一个很大的目标，为了确保兼容性，就要有一个统一的标准。因此，在HTML 5中，就围绕着这个Web标准，重新定义了一套在现有的HTML的基础上修改而来的语法，使它运行在各浏览器时各浏览器都能够符合这个通用标准。

因为关于HTML 5语法解析的算法也都提供了详细的记载，所以各Web浏览器的供应商们可以把HTML 5分析器集中封装在自己的浏览器中。最新的Firefox（默认为4.0以后的版本）与WebKit浏览器引擎中都迅速地封装了供HTML 5使用的分析器，IE（Internet Explorer）与Opera也在努力加快对于HTML 5的支持——浏览器兼容性的提高指日可待。

接下来，让我们具体看一下在HTML 5中，到底对语法进行了哪些改变。

### 2.1.2 HTML 5中的标记方法

首先，让我们来看一下在HTML 5中的标记方法。

1. 内容类型（ContentType）

首先，HTML 5的文件扩展符与内容类型保持不变。也就是说，扩展符仍然为“.html”或“.htm”，内容类型（ContentType）仍然为“text/html”。

2. DOCTYPE声明

DOCTYPE声明是HTML文件中必不可少的，它位于文件第一行。在HTML 4中，它的声明方法如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

在HTML 5中，刻意不使用版本声明，一份文档将会适用于所有版本的HTML。HTML 5中的DOCTYPE声明方法（不区分大小写）如下：

```
<!DOCTYPE html>
```

另外，当使用工具时，也可以在DOCTYPE声明方式中加入SYSTEM识别符，声明方法如下面的代码所示：

```
<!DOCTYPE HTML SYSTEM "about:legacy-compat">
```

在HTML 5中像这样的DOCTYPE声明方式是允许的（不区分大小写,引号不区分是单引号还是双引号）。

3. 指定字符编码

在HTML 4中，使用meta元素的形式指定文件中的字符编码，如下所示：

```
<meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
```

在HTML 5中，可以使用对<meta>元素直接追加charset属性的方式来指定字符编码，如下所示：

```
<meta charset="UTF-8">
```

两种方法都有效，可以继续使用前面一种方式（通过content元素的属性来指定），但是不能同时混合使用两种方式。在以前的网站代码中可能会存在下面代码所示的标记方式，但在HTML 5中，这种字符编码方式将被认为是错误的，这一点请注意：

```
<meta charset="UTF-8" http-equiv="Content-Type" content="text/html;charset=UTF-8">
```

从HTML 5开始，对于文件的字符编码推荐使用UTF-8。

### 2.1.3 HTML 5确保了与之前HTML版本的兼容性

HTML 5的语法是为了保证与之前的HTML语法达到最大程度的兼容而设计的。例如，符合“没有<p>的结束标记”的HTML代码随处可见，HTML 5中并没有把这种情况作为错误来处理，而是允许存在这种情况，但明确地规定了这种情况应该怎么处理。

那么，针对这个问题，让我们从元素标记的省略、具有boolean值的属性、引号的省略这

几方面来详细看一下在HTML 5中是如何确保与之前版本的HTML达到兼容的。

1. 可以省略标记的元素

在HTML 5中，元素的标记可以省略。具体来说，元素的标记分为“不允许写结束标记”、“可以省略结束标记”和“开始标记和结束标记全部可以省略”三种类型。让我们来针对这三类情况列举一个元素清单，其中包括HTML 5中的新元素（关于这些新元素，2.2节将进行介绍）。

- ❑ 不允许写结束标记的元素有：area、base、br、col、command、embed、hr、img、input、keygen、link、meta、param、source、track、wbr。
- ❑ 可以省略结束标记的元素有：li、dt、dd、p、rt、rp、optgroup、option、colgroup、thead、tbody、tfoot、tr、td、th。
- ❑ 可以省略全部标记的元素有：html、head、body、colgroup、tbody。

---

说明：“不允许写结束标记的元素”是指，不允许使用开始标记与结束标记将元素括起来的形式，只允许使用“<元素/>”的形式进行书写。例如“<br>...</br>”的书写方式是错误的，正确的书写方式为“<br/>”。当然，HTML 5之前的版本中<br>这种写法可以被沿用。

“可以省略全部标记的元素”是指，该元素可以完全被省略。请注意，即使标记被省略了，该元素还是以隐式的方式存在的。例如将body元素省略不写时，但它在文档结构中还是存在的，可以使用document.body进行访问。

---

2. 具有boolean值的属性

对于具有boolean值的属性，例如disabled与readonly等，当只写属性而不指定属性值时，表示属性值为true；如果想要将属性值设为false,可以不使用该属性。另外，要想将属性值设定为true时，也可以将属性名设定为属性值，或将空字符串设定为属性值。

属性值的设定方法可以参考下面的代码示例：

```
<!--只写属性不写属性值代表属性为true-->
<input type="checkbox" checked>
<!--不写属性代表属性为false-->
<input type="checkbox">
<!--属性值=属性名，代表属性为true-->
<input type="checkbox" checked="checked">
<!--属性值=空字符串，代表属性为true-->
<input type="checkbox" checked="">
```

3. 省略引号

大家已经知道，指定属性值的时候，属性值两边既可以用双引号，也可以用单引号。

HTML 5在此基础上做了一些改进，当属性值不包括空字符串、“<”、“>”、“=”、单引号、双引号等字符时，属性值两边的引号可以省略。如下面的代码所示：

```
<!-- 请注意type的属性值两边的引号 -->
<input type="text">
<input type='text'>
<input type=text>
```

### 2.1.4 标记示例

现在，让我们通过前面学到的HTML 5的语法知识来看一个关于HTML 5标记的示例。

代码清单2-1完全是用HTML 5写成的，省略了<html>、<head>、<body>等元素。可以通过这个示例复习一下HTML 5的DOCTYPE声明、用<meta>元素的charset属性指定字符编码、<p>元素的结束标记的省略、使用<元素/>的方式来结束<meta>元素，以及<br>元素等本节中所介绍到的知识要点。

代码清单2-1 HTML 5标记示例

```
<!DOCTYPE html>
<meta charset="UTF-8">
<title>HTML 5标记示例</title>
<p>这段代码是根据HTML 5语法
<br/>编写出来的。
```

这段代码在Firefox 4浏览器中的运行结果如图2-1所示，另外，本书中如果没有特别说明使用什么浏览器的时候，默认使用的都是Firefox 4浏览器。


图2-1 HTML 5标记示例

## 2.2 新增的元素和废除的元素

本节将详细介绍HTML 5中新增和废除了哪些元素[⊖]。

### 2.2.1 新增的结构元素

在HTML 5中，新增了以下与结构相关的元素。

❑ section元素

section元素表示页面中的一个内容区块，比如章节、页眉、页脚或页面中的其他部分。它可以与h1、h2、h3、h4、h5、h6等元素结合起来使用，标示文档结构。

HTML 5中代码示例：

```
<section>...</section>
```

⊖ 其他资料介绍的新增元素可能会比下面要介绍的多，这是因为HTML 5在最新发布的版本中，又把这些本来想新增的元素给删除了。

HTML 4中代码示例：

```
<div>...</div>
```

❑ article元素

article元素表示页面中的一块与上下文不相关的独立内容，譬如博客中的一篇文章或报纸中的一篇文章。

HTML 5中代码示例：

```
<article>...</article>
```

HTML 4中代码示例：

```
<div>...</div>
```

❑ aside元素

aside元素表示article元素的内容之外的、与article元素的内容相关的辅助信息。

HTML 5中代码示例：

```
<aside>…</aside>
```

HTML 4中代码示例：

```
<div>...</div>
```

❑ header元素

header元素表示页面中一个内容区块或整个页面的标题。

HTML 5中代码示例：

```
<header>...</header>
```

HTML 4中代码示例：

```
<div>...</div>
```

❑ hgroup元素

hgroup元素用于对整个页面或页面中一个内容区块的标题进行组合。

HTML 5中代码示例：

```
<hgroup>...</hgroup>
```

HTML 4中代码示例：

```
<div>...</div>
```

❑ footer元素

footer元素表示整个页面或页面中一个内容区块的脚注。一般来说，它会包含创作者的姓名、创作日期以及创作者联系信息。

HTML 5中代码示例：

```
<footer></footer>
```

HTML 4中代码示例：

```
<div>...</div>
```

❑ nav元素

nav元素表示页面中导航链接的部分。

HTML 5中代码示例：

```
<nav></nav>
```

HTML 4中代码示例：

```
<ul></ul>
```

❑ figure元素

figure元素表示一段独立的流内容，一般表示文档主体流内容中的一个独立单元。使用figcaption元素为figure元素组添加标题。

HTML 5中代码示例：

```
<figure>
<figcaption>PRC</figcaption>
<p>The People's Republic of China was born in 1949...</p>
</figure>
```

HTML 4中代码示例：

```
<dl>
<h1>PRC</h1>
<p>The People's Republic of China was born in 1949...</p>
</dl>
```

### 2.2.2 新增的其他元素

除了结构元素外，在HTML 5中，还新增了以下元素：

❑ video元素

video元素定义视频，比如电影片段或其他视频流。

HTML 5中代码示例：

```
<video src="movie.ogg" controls="controls">video元素</video>
```

HTML 4中代码示例：

```
<object type="video/ogg" data="movie.ogv">
     <param name="src" value="movie.ogv">
</object>
```

❑ audio元素

audio元素定义音频，比如音乐或其他音频流。

HTML 5中代码示例：

```
<audio src="someaudio.wav">audio元素</audio>
```

HTML 4中代码示例：

```
<object type="application/ogg" data="someaudio.wav">
<param name="src" value="someaudio.wav">
</object>
```

❑ embed元素

embed元素用来插入各种多媒体，格式可以是 Midi、Wav、AIFF、AU、MP3等。

HTML 5中代码示例：

```
<embed src="horse.wav" />
```

HTML 4中代码示例：

```
<object data="flash.swf"  type="application/x-shockwave-flash"></object>
```

❑ mark元素

mark元素主要用来在视觉上向用户呈现那些需要突出显示或高亮显示的文字。mark元素的一个比较典型的应用就是在搜索结果中向用户高亮显示搜索关键词。

HTML 5中代码示例：

```
<mark></mark>
```

HTML 4中代码示例：

```
<span></span>
```

❑ progress元素

progress元素表示运行中的进程，可以使用 progress元素来显示 JavaScript 中耗费时间的函数的进程。

HTML5中代码示例：

```
<progress></progress>
```

这是HTML5中新增功能，故无法用HTML 4代码来实现。

❑ meter元素

meter元素表示度量衡。仅用于已知最大值和最小值的度量。必须定义度量的范围，既可以在元素的文本中，也可以在min/max属性中定义。

HTML5中代码示例：

```
<meter></meter>
```

这是HTML5中新增功能，故无法用HTML 4代码来实现。

❑ time元素

time元素表示日期或时间，也可以同时表示两者。

HTML 5中代码示例：

```
<time></time>
```

HTML 4中代码示例：

```
<span></span>
```

❑ ruby元素

ruby元素表示ruby注释（中文注音或字符）。

HTML 5中代码示例：

```
<ruby>汉 <rt><rp>(</rp>ㄏㄢˋ<rp>)</rp></rt></ruby>
```

这也是HTML 5中新增的功能。

❑ rt元素

rt元素表示字符（中文注音或字符）的解释或发音。

HTML 5中代码示例：

```
<ruby>汉 <rt> ㄏㄢˋ</rt></ruby>
```

这是HTML 5中新增的功能。

❑ rp元素

rp元素在ruby注释中使用，以定义不支持ruby元素的浏览器所显示的内容。

HTML 5中代码示例：

```
<ruby>汉 <rt><rp>(</rp>ㄏㄢˋ<rp>)</rp></rt></ruby>
```

这是HTML 5中新增的功能。

❑ wbr元素

wbr元素表示软换行。wbr元素与br元素的区别是：br元素表示此处必须换行；而wbr元素的意思是浏览器窗口或父级元素的宽度足够宽时（没必要换行时），不进行换行，而当宽度不够时，主动在此处进行换行。wbr元素好像对字符型的语言作用挺大，但是对于中文，貌似没多大用处。

HTML 5中代码示例：

```
<p> To learn AJAX, you must be fami<wbr>liar with the XMLHttp<wbr>
Request Object. </p>
```

这是HTML 5中新增的功能。

❑ canvas元素

canvas元素表示图形，比如图表和其他图像。这个元素本身没有行为，仅提供一块画布，但它把一个绘图API展现给客户端JavaScript，以使脚本能够把想绘制的东西绘制到这块画布上。

HTML 5中代码示例：

```
<canvas id="myCanvas" width="200" height="200"></canvas>
```

HTML 4中代码示例：

```
<object data="inc/hdr.svg" type="image/svg+xml" width="200" height="200">
</object>
```

❑ command元素

command元素表示命令按钮，比如单选按钮、复选框或按钮。

HTML 5中代码示例：

```
<command onclick=cut()" label="cut">
```

这是HTML 5中新增的功能。

❑ details元素

details元素表示用户要求得到并且可以得到的细节信息。它可以与summary元素配合使用。summary元素提供标题或图例。标题是可见的，用户点击标题时，会显示出细节信息。summary元素应该是details元素的第一个子元素。

HTML 5中代码示例：

```
<details>
   <summary>HTML 5</summary>
   This document teaches you everything you have to learn about HTML 5.
</details>
```

这是HTML 5中新增的功能。

❑ datalist元素

datalist元素表示可选数据的列表，与input元素配合使用，可以制作出输入值的下拉列表。

HTML 5中代码示例：

```
<datalist></datalist>
```

这是HTML 5中新增的功能。

❑ datagrid元素

datagrid元素表示可选数据的列表，它以树形列表的形式来显示。

HTML 5中代码示例：

```
<datagrid></datagrid>
```

这是HTML 5中新增的功能。

❑ keygen元素

keygen元素表示生成密钥。

HTML 5中代码示例：

```
<keygen>
```

这是HTML 5中新增的功能。

❑ output元素

output元素表示不同类型的输出，比如脚本的输出。

HTML 5中代码示例：

```
<output></output>
```

HTML 4中代码示例：

```
<span></span>
```

❑ source元素

source元素为媒介元素（比如<video>和<audio>）定义媒介资源。

HTML 5中代码示例：

```
<source>
```

HTML 4中代码示例：

```
<param>
```

❑ menu元素

menu元素表示菜单列表。当希望列出表单控件时使用该标签。

HTML 5中代码示例：

```
<menu>
```

```
    <li><input type="checkbox" />Red</li>
    <li><input type="checkbox" />blue</li>
</menu>
```

在HTML 4中，menu元素不被推荐使用。

### 2.2.3 新增的input元素的类型

HTML 5中新增了很多input元素的类型，现列举如下：

❑ email

email类型表示必须输入E-mail地址的文本输入框。

❑ url

url类型表示必须输入URL地址的文本输入框。

❑ number

number类型表示必须输入数值的文本输入框。

❑ range

range类型表示必须输入一定范围内数字值的文本输入框。

❑ Date Pickers

HTML 5拥有多个可供选取日期和时间的新型输入文本框：

date —— 选取日、月、年

month —— 选取月、年

week —— 选取周和年

time —— 选取时间（小时和分钟）

datetime —— 选取时间、日、月、年（UTC时间）

datetime-local —— 选取时间、日、月、年（本地时间）

### 2.2.4 废除的元素

由于各种原因，在HTML 5中废除了很多元素，简单介绍如下。

1. 能使用CSS替代的元素

对于basefont、big、center、font、s、strike、tt、u这些元素，由于它们的功能都是纯粹为画面展示服务的，而HTML 5中提倡把画面展示性功能放在CSS样式表中统一编辑，所以将这些元素废除了，并使用编辑CSS、添加CSS样式表的方式进行替代。其中font元素允许由“所见即所得”的编辑器来插入，s元素、strike元素可以由del元素替代，tt元素可以由CSS的font-family属性替代。

2. 不再使用frame框架

对于frameset元素、frame元素与noframes元素，由于frame框架对网页可用性存在负面影响，在HTML 5中已不支持frame框架，只支持iframe框架，或者用服务器方创建的由多个页面组成的复合页面的形式，同时将以上这三个元素废除。

3. 只有部分浏览器支持的元素

对于applet、bgsound、blink、marquee等元素，由于只有部分浏览器支持这些元素，特别是bgsound元素以及marquee元素，只被Internet Explorer所支持，所以在HTML 5中被废除。其中applet元素可由embed元素或object元素替代，bgsound元素可由audio元素替代，marquee可以由JavaScript编程的方式所替代。

4. 其他被废除的元素

其他被废除元素还有：

- 废除rb元素，使用ruby元素替代
- 废除acronym元素，使用abbr元素替代
- 废除dir元素，使用ul元素替代
- 废除isindex元素，使用form元素与input元素相结合的方式替代
- 废除listing元素，使用pre元素替代
- 废除xmp元素，使用code元素替代
- 废除nextid元素，使用GUIDS替代
- 废除plaintext元素，使用“ text/plian ”MIME类型替代

## 2.3　新增的属性和废除的属性

在HTML 5中，在增加和废除了很多元素的同时，也增加和废除了很多属性，本节对于这些增加和废除的属性进行简单介绍[⊖]。

### 2.3.1　新增的属性

1. 表单相关的属性

新增的与表单相关的元素如下：

- 可以对input（type=text）、select、textarea与button元素指定autofocus属性。它以指定属性的方式让元素在画面打开时自动获得焦点。
- 可以对input元素（type=text）与textarea元素指定placeholder属性，它会对用户的输入进行提示，提示用户可以输入的内容。
- 可以对input、output、select、textarea、button与fieldset指定form属性，声明它属于哪个表单，然后将其放置在页面上任何位置，而不是表单之内。
- 可以对input元素（type=text）与textarea元素指定required属性。该属性表示在用户提交的时候进行检查，检查该元素内一定要有输入内容。
- 为input元素增加了几个新的属性：autocomplete、min、max、multiple、pattern与step。同时还有一个新的list元素与datalist元素配合使用。datalist元素与autocomlete属性配合使用。multiple属性允许在上传文件时一次上传多个文件。

⊖ 其他资料介绍的新增属性可能会比下面要介绍得多，这是因为HTML 5在最新发布的版本中，又把这些本来想新增的属性给删除了。

- ❑ 为input元素与button元素增加了新属性formaction、formenctype、formmethod、formnovalidate与formtarget，他们可以重载form元素的action、enctype、method、novalidate与target属性。为fieldset元素增加了disabled属性，可以把它的子元素设为disabled（无效）状态。
- ❑ 为input元素、button元素、form元素增加了novalidate属性，该属性可以取消提交时进行的有关检查，表单可以被无条件地提交。
- ❑ 为所有可使用标签（label元素）的表单元素[包括非隐藏的input元素（type属性值不等于hidden)、button元素、select元素、textarea元素、meter元素、output元素、progress元素及keygen元素]定义一个labels属性，属性值为一个NodeList对象，代表该元素所绑定的标签元素所构成的集合。
- ❑ 可以在标签（label元素）内部放置一个表单元素，并且通过该标签的control属性来访问该表单元素。
- ❑ 针对input元素与textarea元素在HTML 5中增加SelectionDirection属性。当用户在这两个元素中用鼠标选取部分文字时，可以使用该属性来获取选取方向。当用户正向选取文字时，该属性值为“ forward ”；当用户反向选取文字时，该属性值为“ backward ”。当用户没有选取任何文字时，该属性值为“ forward ”。
- ❑ 对复选框（checkbox元素）添加indeterminate属性，以说明复选框处于“ 尚未明确是否选取 ”状态。
- ❑ 对类型为image的input元素添加，用于指定图片按钮中图片高度的height属性与用于指定图片宽度的width属性。
- ❑ 对textarea元素新增用于限定可输入文字个数的maxlength属性，与用于指定表单提交时是否在文字换行处添加换行符的wrap属性。
- ❑ 对iframe元素新增sandbox属性，其作用是出于安全性方面的原因，对iframe元素内的内容是否允许显示，表单是否允许被提交，以及脚本是否允许被执行等方面进行一些限制。
- ❑ 对script元素新增async属性与defer属性，用于加快页面的加载速度，使脚本代码的读取不再妨碍页面上其他元素的加载。

2. 链接相关属性

新增的与链接相关的属性如下：

- ❑ 为a与area元素增加了media属性，该属性规定目标URL是为什么类型的媒介/设备进行优化的，只能在href属性存在时使用。
- ❑ 为area元素增加了hreflang属性与rel属性，以保持与a元素、link元素的一致。
- ❑ 为link元素增加了新属性sizes。该属性可以与icon元素结合使用（通过rel属性），该属性指定关联图标（icon元素）的大小。
- ❑ 为base元素增加了target属性，主要目的是保持与a元素的一致性。

3. 其他属性

除了上面介绍的与表单和链接相关的属性外，HTML 5还增加了下面的属性：

- ❑ 为ol元素增加属性reversed，它指定列表倒序显示。

- 为meta元素增加charset属性，因为这个属性已经被广泛支持了，而且为文档的字符编码的指定提供了一种比较良好的方式。
- 为menu元素增加了两个新的属性——type与label。label属性为菜单定义一个可见的标注，type属性让菜单可以以上下文菜单、工具条与列表菜单的三种形式出现。
- 为style元素增加scoped属性，用来规定样式的作用范围，譬如只对页面上某个树起作用。
- 为script元素增加async属性，它定义脚本是否异步执行。
- 为html元素增加属性manifest，开发离线Web应用程序时它与API结合使用，定义一个URL，在这个URL上描述文档的缓存信息。
- 为iframe元素增加三个属性sandbox、seamless与srcdoc，用来提高页面安全性，防止不信任的Web页面执行某些操作。

### 2.3.2　废除的属性

HTML 4中的一些属性在HTML 5中不再被使用，而是采用其他属性或其他方案进行替代，具体如表2-1所示。

表2-1　在HTML 5中被废除了的属性

| 在HTML 4中使用的属性 | 使用该属性的元素 | 在HTML 5中的替代方案 |
|---|---|---|
| rev | link、a | rel |
| charset | link、a | 在被链接的资源的中使用HTTP Content-type头元素 |
| shape、coords | a | 使用area元素代替a元素 |
| longdesc | img、iframe | 使用a元素链接到较长描述 |
| target | link | 多余属性，被省略 |
| nohref | area | 多余属性，被省略 |
| profile | head | 多余属性，被省略 |
| version | html | 多余属性，被省略 |
| name | img | id |
| scheme | meta | 只为某个表单域使用scheme |
| archive、classid、codebase、codetype、declare、standby | object | 使用data与type属性类调用插件。需要使用这些属性来设置参数时，使用param属性 |
| valuetype、type | param | 使用name与value属性，不声明值的MIME类型 |
| axis、abbr | td、th | 使用以明确简洁的文字开头、后跟详述文字的形式。可以对更详细内容使用title属性，来使单元格的内容变得简短 |
| scope | td | 在被链接的资源的中使用HTTP Content-type头元素 |
| align | caption、input legend、div、h1、h2、h3、h4、h5、h6、p | 使用CSS样式表替代 |

（续）

| 在HTML 4中使用的属性 | 使用该属性的元素 | 在HTML 5中的替代方案 |
| --- | --- | --- |
| alink、link、text、vlink、background、bgcolor | body | 使用CSS样式表替代 |
| align、bgcolor、border、cellpadding、cellspacing、frame、rules、width | table | 使用CSS样式表替代 |
| align、char、charoff、height、nowrap、valign | tbody、thead、tfoot | 使用CSS样式表替代 |
| align、bgcolor、char、charoff、height、nowrap、valign、width | td、th | 使用CSS样式表替代 |
| align、bgcolor、char、charoff、valign | tr | 使用CSS样式表替代 |
| align、char、charoff、valign、width | col、colgroup | 使用CSS样式表替代 |
| align、border、hspace、vspace | object | 使用CSS样式表替代 |
| clear | br | 使用CSS样式表替代 |
| compact、type | ol、ul、li | 使用CSS样式表替代 |
| compact | dl | 使用CSS样式表替代 |
| compact | menu | 使用CSS样式表替代 |
| width | pre | 使用CSS样式表替代 |
| align、hspace、vspace | img | 使用CSS样式表替代 |
| align、noshade、size、width | hr | 使用CSS样式表替代 |
| align、frameborder、scrolling、marginheight、marginwidth | iframe | 使用CSS样式表替代 |
| autosubmit | menu | |

## 2.4 全局属性

在HTML 5中，新增了一个“全局属性”的概念。所谓全局属性，是指可以对任何元素都使用的属性，本节将详细介绍几种常用的全局属性。

### 2.4.1 contentEditable属性

contentEditable是由微软开发、被其他浏览器反编译并投入应用的一个全局属性。该属性的主要功能是允许用户编辑元素中的内容，所以该元素必须是可以获得鼠标焦点的元素，而且在点击鼠标后要向用户提供一个插入符号，提示用户该元素中的内容允许编辑。contentEditable属性是一个布尔值属性，可以被指定为true或false。

除此之外，该属性还有个隐藏的inherit（继承）状态，属性为true时，元素被指定为允许编辑；属性为false时，元素被指定为不允许编辑；未指定true或false时，则由inherit状态来决定，如果元素的父元素是可编辑的，则该元素就是可编辑的。

另外，除了contentEditable属性外，元素还具有一个isContentEditable属性，当元素可编辑时，该属性为true；当元素不可编辑时，该属性为false。

代码清单2-2中给出了一个使用contentEditable属性的示例，当列表元素被加上contentEditable属性后，该元素就变成可编辑的了，读者可自行在浏览器中对该示例进行试验。

代码清单2-2　contentEditable属性示例

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8">
<title>contentEditalbe属性示例</title>
</head>
<h2>可编辑列表</h2>
<ul contentEditable="true">
<li>列表元素1</li>
<li>列表元素2</li>
<li>列表元素3</li>
</ul>
```

这段代码运行后的结果如图2-2所示。


图2-2　可编辑列表示例

在编辑完元素中的内容后，如果想要保存其中内容，只能把该元素的innerHTML发送到服务器端进行保存，因为改变元素内容后该元素的innerHTML内容也会随之改变，目前还没有特别的API来保存编辑后元素中的内容。

最后，在这里列举一下支持contentEditable属性的元素：defaults、A、ABBR、ACRONYM、ADDRESS、B、BDO、BIG、BLOCKQUOTE、BODY、BUTTON、CENTER、CITE、CODE、CUSTOM、DD、DEL、DFN、DIR、DIV、DL、DT、EM、FIELDSET、FONT、FORM、hn、I、INPUT type=button、INPUT type=password、INPUT type=radio、INPUT type=reset、INPUT type=submit、INPUT type=text、INS、ISINDEX、KBD、LABEL。

### 2.4.2 designMode属性

designMode属性用来指定整个页面是否可编辑，当页面可编辑时，页面中任何支持上文所述的contentEditable属性的元素都变成了可编辑状态。designMode属性只能在JavaScript脚本里被编辑修改。该属性有两个值——“ on ”与“ off ”。属性被指定为“ on ”时，页面可编辑；被指定为“ off ”时，页面不可编辑。使用JavaScript脚本来指定designMode属性的方法如下所示：

```
document.designMode="on"
```

针对designMode属性，各浏览器的支持情况也各不相同：

- ❑ IE8：出于安全考虑，不允许使用designMode属性让页面进入编辑状态。
- ❑ IE9：允许使用designMode属性让页面进入编辑状态。
- ❑ Chrome 3和Safari：使用内嵌frame的方式,该内嵌frame是可编辑的。
- ❑ Firefox和Opera：允许使用designMode属性让页面进入编辑状态。

### 2.4.3 hidden属性

在HTML 5中，所有的元素都允许使用一个hidden属性。该属性类似于input元素中的hidden元素，功能是通知浏览器不渲染该元素，使该元素处于不可见状态。但是元素中的内容还是浏览器创建的，也就是说页面装载后允许使用JavaScript脚本将该属性取消，取消后该元素变为可见状态，同时元素中的内容也即时显示出来。Hidden属性是一个布尔值的属性，当设为true时，元素处于不可见状态；当设为false时，元素处于可见状态。

### 2.4.4 spellcheck属性

spellcheck属性是HTML 5针对input元素（type=text）与textarea这两个文本输入框提供的一个新属性，它的功能为对用户输入的文本内容进行拼写和语法检查。spellcheck属性是一个布尔值的属性，具有true或false两种值。但是它在书写时有一个特殊的地方，就是必须明确声明属性值为true或false，书写方法如下所示：

```
<!—以下两种书写方法正确-->
<textarea spellcheck="true" >
<input type=text spellcheck=false>
<!—以下书写方法为错误-->
<textarea spellcheck >
```

需要注意的是，如果元素的readOnly属性或disabled属性设为true，则不执行拼写检查。

目前除了IE之外，Firefox、Chrome、Safari、Opera等浏览器都对该属性提供了支持。图2-3为Opera浏览器中spellcheck属性的表现形式。


图2-3 Opera浏览器中spellcheck的属性示例

### 2.4.5 tabindex属性

tabindex是开发中的一个基本概念，当不断敲击Tab键让窗口或页面中的控件获得焦点，对窗口或页面中的所有控件进行遍历的时候，每一个控件的tabindex表示该控件是第几个被访问到的。

过去这个属性在编辑网页时是非常有用的，但如今控件的遍历顺序是由元素在页面上所处的位置决定的，所以就不再需要了。

但是tabindex还有另外一个作用，在默认情况下，只有链接元素与表单元素可以通过按键获得焦点。如果对其他元素使用tabindex属性后，也能让该元素获得焦点，那么当脚本中执行focus()语句的时候，就可以让该元素获得焦点了。但这样做会产生一个副作用：该元素也可以通过按Tab键获得焦点，而这有时可能不是开发者想要的结果。

把元素的tabindex值设为负数（通常为 - 1）后就解决这个问题了。tabindex的值为负数后，仍然可以通过编程的方式让元素获得焦点，但按下Tab键时该元素就不能获得焦点了，这在复杂的页面中或复杂的Web应用程序中是十分有用的。在HTML 4中， - 1是一个无用的属性值，但到了HTML 5中，通过巧妙地运用让该属性值得到了极大的应用。

# 第3章
# HTML 5的结构

3.1 新增的主体结构元素
3.2 新增的非主体结构元素
3.3 HTML 5结构

在HTML 5对HTML 4所做的各种修改中，一个比较重大的修改就是为了使文档结构更加清晰明确，容易阅读，增加了很多新的结构元素。本章将详细介绍这些新增的结构元素，包括它们的定义、使用方法以及使用示例，最后再介绍一下在HTML 5中，究竟怎样将这些新增的结构元素结合起来综合使用。

学习内容：

- 掌握HTML 5中新增的主体结构元素的定义、使用方法，以及使用场合。新增的主体结构元素包括article元素、section元素、nav元素及aside元素。
- 掌握HTML 5中新增的非主体结构元素的定义、使用方法，以及使用场合。新增的非主体结构元素包括header元素、hgroup元素、footer元素及address元素。
- 掌握HTML 5中应该怎样结合运用这些新增结构元素来合理编排页面总体布局，掌握什么是显式编排，什么是隐式编排，HTML 5分析器是按什么原则来分析页面结构的，以及怎样对这些新增元素使用CSS样式。

## 3.1　新增的主体结构元素

在HTML 5中，为了使文档的结构更加清晰明确，追加了几个与页眉、页脚、内容区块等文档结构相关联的结构元素。需要说明的是，本章所讲的内容区块是指将HTML页面按逻辑进行分割后的单位。例如对于书籍来说，章、节都可以称为内容区块；对于博客网站来说，导航菜单、文章正文、文章的评论等每一个部分都可称为内容区块。

接下来将详细讲解HTML 5中在页面的主体结构方面新增加的结构元素。

### 3.1.1　article元素

article元素代表文档、页面或应用程序中独立的、完整的、可以独自被外部引用的内容。它可以是一篇博客或报刊中的文章、一篇论坛帖子、一段用户评论或独立的插件，或其他任何独立的内容。

除了内容部分，一个article元素通常有它自己的标题（一般放在一个header元素里面），有时还有自己的脚注。

现在，让我们以博客为例来看一段关于article元素的代码示例，如代码清单3-1所示。

代码清单3-1　article元素示例

```
<article>
    <header>
        <h1>苹果</h1>
        <p>发表日期: <time pubdate="pubdate">2010/10/09</time></p>
    </header>
    <p><b>苹果</b> ，植物类水果，多次花果...（"苹果"文章正文）</p>
    <footer>
        <p><small>著作权归***公司所有。</small></p>
    </footer>
</article>
```

这个示例是一篇讲述苹果的博客文章，在header元素中嵌入了文章的标题部分，在这部分中，文章的标题“苹果”被镶嵌在h1元素中，文章的发表日期镶嵌在p元素中。在标题下部的p元素中，嵌入了一大段该博客文章的正文，在结尾处的footer元素中，嵌入了文章的著作权，作为脚注。整个示例的内容相对比较独立、完整，因此，对这部分内容使用了article元素。

article元素是可以嵌套使用的，内层的内容在原则上需要与外层的内容相关联。例如，一篇博客文章中，针对该文章的评论就可以使用嵌套article元素的方式；用来呈现评论的article元素被包含在表示整体内容的article元素里面。

接着，让我们来看一个关于article元素嵌套的代码示例，如代码清单3-2所示。

代码清单3-2　article元素嵌套示例

```
<article>
    <header>
        <h1>苹果</h1>
        <p>发表日期:
                <time pubdate datetime="2010/10/09">2010/10/09</time>
        </p>
    </header>
    <p><b>苹果</b> ，植物类水果，多次花果...（"苹果"文章正文）</p>
    <section>
        <h2>评论</h2>
        <article>
            <header>
               <h3>发表者：陆凌牛</h3>
               <p>
                  <time pubdate datetime="2010-10-10T19:10-08:00">
                      1小时前
                  </time>
               </p>
            </header>
            <p>我喜欢苹果，我最喜爱的品种是红富士。</p>
        </article>
        <article>
            <header>
               <h3>发表者：张玉</h3>
               <p>
                  <time pubdate datetime="2010-10-10T19:15-08:00">
                      1小时前
                  </time>
               </p>
            </header>
            <p>苹果？我不喜欢，我喜欢吃橘子。</p>
        </article>
    </section>
</article>
```

这个示例中的内容比代码清单3-1中的内容更加完整了，它添加了文章读者的评论内容，示例的整体内容还是比较独立、完整的，因此对其使用article元素。具体来说，示例内容又分为几部分，文章标题放在了header元素中，文章正文放在了header元素后面的p元素中，然

后section元素把正文与评论部分进行了区分（section元素马上就要讲到，是一个分块元素，用来把页面中的内容进行分块），在section元素中嵌入了评论的内容，评论中每一个人的评论相对来说又是比较独立、完整的，因此对它们都使用一个article元素，在评论的article元素中，又可以分为标题与评论内容部分，分别放在header元素与p元素中。

关于示例中提到的time元素与pubdate属性，请查看本节结尾处有关time元素与pubdate属性的说明。

另外，article元素也可以用来表示插件，它的作用是使插件看起来好像内嵌在页面中一样。代码清单3-3是它的一个示例。

代码清单3-3 用article元素表示插件

```
<article>
    <h1>My Fruit Spinner</h1>
    <object>
        <param name="allowFullScreen" value="true">
        <embed src="#" width="600" height="395"></embed>
    </object>
</article>
```

## 3.1.2 section元素

section元素用于对网站或应用程序中页面上的内容进行分块。一个section元素通常由内容及其标题组成。但section元素并非一个普通的容器元素；当一个容器需要被直接定义样式或通过脚本定义行为时，推荐使用div而非section元素。

我们可以这样理解：section元素中的内容可以单独存储到数据库中或输出到Word文档中。代码清单3-4是它的一个示例（注意：标题部分位于它的内部，而不是它的前面）。

代码清单3-4 section元素示例

```
<section>
    <h1>苹果</h1>
    <p><b>苹果</b> ，植物类水果，多次花果...（"苹果"文章正文）</p>
</section>
```

通常不推荐为那些没有标题的内容使用section元素，可以使用HTML 5轮廓工具来检查页面中是否有没标题的section，HTML 5轮廓工具的网址为“ http://gsnedders.html5.org/outliner/ ”，如果使用该工具进行检查后，发现某个section的说明中有“ untieled section ”（没有标题的section）文字，这个section就有可能使用不当（但是nav元素或aside元素没有标题是合理的）。

section元素的作用是对页面上的内容进行分块，或者说对文章进行分段，请不要与“ 有着自己的完整的、独立的内容 ”的article元素混淆。

下面，我们来看article元素与section元素结合使用的两个示例，希望能够帮助你更好地理解article元素与section元素的区别。

首先来看一个带有section元素的article元素示例，如代码清单3-5所示。

代码清单3-5 带有section元素的article元素示例

```
<article>
    <h1>苹果</h1>
    <p><b>苹果</b> ，植物类水果，多次花果...</p>
    <section>
        <h2>红富士</h2>
        <p>红富士是从普通富士的芽(枝)变中选育出的着色系富的统称...</p>
    </section>
    <section>
        <h2>国光</h2>
        <p>国光苹果品，又名小国光、万寿。原产美国，1600年发现的偶然实生苗...</p>
    </section>
</article>
```

代码清单3-5中的内容首先是一段独立的、完整的内容，因此使用article元素。该内容是一篇关于苹果的文章，该文章分为3段，每一段都有一个独立的标题，因此使用了两个section元素。请记住，对文章分段的工作也是使用section元素完成的。可能有人会问，为什么没有对第一段使用section元素，这里其实是可以使用section元素的，但是由于其结构比较清晰，分析器可以识别第一段内容在一个section元素里，所以也可以将第一个section元素略，但是如果第一个section元素里还要包含子section元素或子article元素，那么就必须写明第一个section元素了。

接着，我们来看一个包含article元素的section元素示例，如代码清单3-6所示。

代码清单3-6 包含article元素的section元素示例

```
<section>
    <h1>水果</h1>
    <article>
        <h2>苹果</h2>
        <p>苹果，植物类水果，多次花果...</p>
    </article>
    <article>
        <h2>橘子</h2>
        <p>橘子，是芸香科柑橘属的一种水果...</p>
    </article>
    <article>
        <h2>香蕉</h2>
        <p>香蕉,属于芭蕉科芭蕉属植物，又指其果实，热带地区广泛栽培食用...</p>
    </article>
</section>
```

这个示例比前面的示例复杂了一些，首先，它是一篇文章中的一段，因此最初没有使用article元素。但是，在这一段中有几块独立的内容，因此，嵌入了几个独立的article元素。

看到这里，你可能又会糊涂了，这两个元素可以互换使用吗？它们的区别到底是什么呢？事实上，在HTML 5中，article元素可以看成是一种特殊种类的section元素，它比section元素更强调独立性。即section元素强调分段或分块，而article强调独立性。具体来说，如果一块内容相对来说比较独立、完整的时候，应该使用article元素，但是如果你想将一块内容分成几段的时候，应该使用section元素。另外，在HTML 5中，div元素变成了一种容器，当

使用CSS样式的时候，可以对这个容器进行一个总体的CSS样式的套用。

另外再补充一点，在HTML 5中，你可以将所有页面的从属部分，譬如导航条、菜单、版权说明等包含在一个统一的页面中，以便统一使用CSS样式来进行装饰。

最后，关于section元素的使用禁忌总结如下：

1）不要将section元素用作设置样式的页面容器，那是div元素的工作。

2）如果article元素、aside元素或nav元素更符合使用条件，不要使用section元素。

3）不要为没有标题的内容区块使用section元素。

### 3.1.3　nav元素

nav元素是一个可以用作页面导航的链接组，其中的导航元素链接到其他页面或当前页面的其他部分。并不是所有的链接组都要被放进nav元素，只需要将主要的、基本的链接组放进nav元素即可。例如，在页脚中通常会有一组链接，包括服务条款、首页、版权声明等，这时使用footer元素是最恰当。一个页面中可以拥有多个nav元素，作为页面整体或不同部分的导航。

接着让我们来看一个nav元素的使用示例，在这个示例中，一个页面由几部分组成，每个部分都带有链接，但只将最主要的链接放入了nav元素中，如代码清单3-7所示。

代码清单3-7　nav元素示例

```
<body>
<h1>技术资料</h1>
<nav>
    <ul>
        <li><a href="/">主页</a></li>
        <li><a href="/events">开发文档</a></li>
        ...more...
    </ul>
</nav>
<article>
   <header>
        <h1>HTML 5与CSS 3的历史</h1>
        <nav>
            <ul>
                <li><a href="#HTML 5">HTML 5的历史</a></li>
                <li><a href="#CSS 3">CSS 3的历史</a></li>
                ...more...
            </ul>
        </nav>
    </header>
    <section id="HTML 5">
        <h1>HTML 5的历史</h1>
        <p>讲述HTML 5的历史的正文</p>
    </section>
    <section id="CSS 3">
        <h1>CSS 3的历史</h1>
        <p>讲述CSS 3的历史的正文</p>
    </section>
```

```
        ...more...
        <footer>
            <p>
                <a href="?edit">编辑</a> |
                <a href="?delete">删除</a> |
                <a href="?rename">重命名</a>
            </p>
        </footer>
    </article>
    <footer>
        <p><small>版权所有：陆凌牛</small></p>
    </footer>
    </body>
```

在这个例子中，第一个nav元素用于页面导航，将页面跳转到其他页面上去（跳转到网站主页或开发文档目录页面）；第二个nav元素放置在article元素中，用作这篇文章中两个组成部分的页内导航。

具体来说，nav元素可以用于以下这些场合：

- ❑ 传统导航条。现在主流网站上都有不同层级的导航条，其作用是将当前画面跳转到网站的其他主要页面上去。
- ❑ 侧边栏导航。现在主流博客网站及商品网站上都有侧边栏导航，其作用是将页面从当前文章或当前商品跳转到其他文章或其他商品页面上去。
- ❑ 页内导航。页内导航的作用是在本页面几个主要的组成部分之间进行跳转。
- ❑ 翻页操作。翻页操作是指在多个页面的前后页或博客网站的前后篇文章滚动。

除此之外，nav元素也可以用于其他所有你觉得是重要的、基本的导航链接组中。

请注意：在HTML 5中不要用menu元素代替nav元素。过去有很多Web应用程序的开发员喜欢用menu元素进行导航，我想有必要再次强调，menu元素是用在一系列发出命令的菜单上的，是一种交互性的元素，或者更确切地说是使用在Web应用程序中的。

### 3.1.4 aside元素

aside元素用来表示当前页面或文章的附属信息部分，它可以包含与当前页面或主要内容相关的引用、侧边栏、广告、导航条，以及其他类似的有别于主要内容的部分。

aside元素主要有以下两种使用方法。

1）被包含在article元素中作为主要内容的附属信息部分，其中的内容可以是与当前文章有关的参考资料、名词解释，等等。这部分代码请看代码清单3-8。

代码清单3-8 文章内部的aside元素示例

```
<!DOCTYPE html>
<head>
<meta charset="utf-8">
<title>aside元素示例</title>
</head>
<body>
<header>
```

```
    <h1>F#入门</h1>
</header>
<article>
    <h1>第四节  词法闭包</h1>
    <p>lambda表达式可以创建词法闭包...(文章正文）</p>
    <aside>
        <!-- 因为这个aside元素被放置在一个article元素内部,
        所以分析器将这个aside元素的内容理解成是和article元素的内容相关联的。 -->
        <h1>名词解释</h1>
        <dl>
            <dt>F#</dt>
            <dd>F#为.Net2010中引入的新型函数型编程语言</dd>
        </dl>
        <dl>
            <dt>词法闭包</dt>
            <dd>词法闭包是指，将创建lambda表达式时的环境保存起来...(详细解释）</dd>
       </dl>
    </aside>
</article>
</body>
```

程序运行结果如图3-1所示。


图3-1　aside元素示例

这是笔者博客网页中的一篇文章，网页的标题放在了header元素中，在header元素的后面将所有关于文章的部分放在了一个article元素中，将文章的正文部分放在了一个p元素中，但是该文章还有一个名词解释的附属部分，用来解释该文章中的一些名词，因此，在p元素的下部又放置了一个aside元素，用来存放名词解释部分的内容。

2）在article元素之外使用，作为页面或站点全局的附属信息部分。最典型的形式是侧边栏，其中的内容可以是友情链接，博客中其他文章列表、广告单元等。下面这个示例为标准博客网页中一个侧边栏的示例，示例中的“ IT新技术 ”为博客的名称，如代码清单3-9所示。

代码清单3-9 侧边栏示例

```
<aside>
    <nav>
        <h2>评论</h2>
        <ul>
            <li>
            <a href="http://blog.sina.com.cn/1683">erway</a>    10-24 14:25
            </li>
            <li>
            <a href="http://blog.sina.com.cn/u/1345">太阳雨</a>10-22 23:48<br/>
            <a href="http://blog.sina.com.cn/s/blog_6a9kv8f.html#comment">
               顶，拜读一下老牛的文章
             </a>
            </li>
            <li>
            <a href="http://blog.sina.com.cn/u/1259295385">新浪官博</a>
            08-12 08:50<br/>
            <a href="#">恭喜！您已经成功开通了博客</a>
            </li>
       </ul>
    </nav>
</aside>
```

如果对这部分再加上CSS样式，在浏览器中的显示效果如图3-2所示。

该示例为一个典型的博客网站中的侧边栏部分，因此放在了aside元素中，但是该侧边栏又是具有导航作用的，因此放置在nav元素中，该侧边栏的标题是“评论”，放在了h2元素中，在标题之后使用了一个ul列表，用来存放具体的导航链接中。


图3-2 用aside元素实现的侧边栏示例

### 3.1.5 time元素与微格式

首先来说一下微格式，它是一种利用HTML的class属性来对网页添加附加信息的方法，附加信息例如新闻事件发生的日期和时间、个人电话号码、企业邮箱等。

微格式并不是在HTML 5之后才有的，在HTML 5之前它就和HTML结合使用了，但是在使用过程中发现在日期和时间的机器编码上出现了一些问题，编码过程中会产生一些歧义。HTML 5增加了一种新的元素来无歧义地、明确地对机器的日期和时间进行编码，并且以让人易读的方式来展现它。这个元素就是time元素。

time元素代表24小时中的某个时刻或某个日期，表示时刻时允许带时差。它可以定义很多格式的日期和时间，如下所示：

```
<time datetime="2010-11-13">2010年11月13日</time>
<time datetime="2010-11-13">11月13日</time>
<time datetime="2010-11-13">我的生日</time>
<time datetime="2010-11-13T20:00">我生日的晚上8点</time>
<time datetime="2010-11-13T20:00Z">我生日的晚上8点</time>
<time datetime="2010-11-13T20:00+09:00">我生日的晚上8点的美国时间</time>
```

编码时机器读到的部分在datetime属性里，而元素的开始标记与结束标记中间的部分是显示在网页上的。datetime属性中日期与时间之间要用“T”文字分隔，“T”表示时间。请注意倒数第二行，时间加上Z文字表示给机器编码时使用UTC标准时间，倒数第一行则加上了时差，表示向机器编码另一地区时间，如果是编码本地时间，则不需要添加时差。

### 3.1.6 pubdate属性

pubdate属性是一个可选的、boolean值的属性，它可以用到article元素中的time元素上，意思是time元素代表了文章（artilce元素的内容）或整个网页的发布日期，pubdate属性的具体使用方法如代码清单3-10所示。

代码清单3-10 pubdate与time结合使用

```
<article>
    <header>
        <h1>苹果</h1>
        <p>发布日期
           <time datetime="2010-10-29" pubdate>2010年10月29日</time>
        </p>
    </header>
   <p>苹果，植物类水果，多次花果...（"苹果"文章正文）</p>
  ...
</article>
```

你也许会疑惑为什么需要用到pubdate属性，为什么不能认为time元素就直接表示了文章或网页的发布日期呢？请看代码清单3-11。

代码清单3-11 pubdate与time结合使用

```
<article>
    <header>
        <h1>关于<time datetime=2010-10-29>10月29日</time>的舞会通知</h1>
        <p>发布日期:
           <time datetime=2010-10-11 pubdate>2010年10月11日</time>
        </p>
    </header>
    <p>大家好:我是法律系3年级学生代表,.....(关于舞会的通知)</p>
</article>
```

在这个例子中，有两个time元素，分别定义了两个日期——一个是舞会日期，另一个是通知发布日期。由于都使用了time元素，所以需要使用pubdate属性表明哪个time元素代表了通知的发布日期。

## 3.2 新增的非主体结构元素

除了以上几个主要的结构元素之外，HTML 5内还增加了一些表示逻辑结构或附加信息的非主体结构元素。下面分别来介绍。

### 3.2.1 header元素

header元素是一种具有引导和导航作用的结构元素，通常用来放置整个页面或页面内的一个内容区块的标题，但也可以包含其他内容，例如数据表格、搜索表单或相关的logo图片。

很明显，整个页面的标题应该放在页面的开头，我们可以用如下所示的形式书写页面的标题：

```
<header><h1>页面标题</h1></header>
```

需要强调的一点是：一个网页内并未限制header元素的个数，可以拥有多个，可以为每个内容区块加一个header元素，如代码清单3-12所示。

代码清单3-12 多个header元素示例

```
<header>
    <h1>网页标题</h1>
</header>
<article>
    <header>
        <h1>文章标题</h1>
    </header>
    <p>文章正文</p>
</article>
```

在HTML 5中，一个header元素通常包括至少一个heading元素（h1-h6），也可以包括我们后面将要讨论的hgroup元素，还可以包括其他元素（譬如table或form），根据最新的W3C HTML 5标准，还可以包括nav元素。最后，让我们看一下博客网页中header元素的一个应用示例。示例中header元素处于页面顶部。详见代码清单3-13。

代码清单3-13 博客网页中header元素的示例

```
<header>
<hgroup>
<h1>IT新技术</h1>
<a href="http://blog.sina.com.cn/itnewtech">
http://blog.sina.com.cn/itnewtech
</a>
<a href="#">[订阅]</a>
<a href="#">[手机订阅]</a>
</hgroup>
<nav>
<ul>
<li>首页</li>
<li><a href="http://blog.sina.com.cn/articlelist1.html">博文目录</a></li>
<li><a href="http://photo.blog.sina.com.cn/itnewtechl">图片</a></li>
<li><a href="http://photo.blog.sina.com.cn/itnewtech">关于我</a></li>
</nav>
</header>
```

如果对这段代码使用CSS样式，显示界面如图3-3所示。


图3-3 博客网页中header元素示例

## 3.2.2 hgroup元素

hgroup元素是将标题及其子标题进行分组的元素。hgroup元素通常会将h1～h6元素进行分组，譬如一个内容区块的标题及其子标题算一组。

通常，如果文章只有一个主标题，是不需要hgroup元素的，如代码清单3-14所示。

代码清单3-14 只有header元素的示例

```
<article>
    <header>
        <h1>文章标题</h1>
        <p><time datetime="2010-03-20">2010年10月29日</time></p>
    </header>
    <p>文章正文</p>
</article>
```

但是，如果文章有主标题，主标题下有子标题，就需要使用hgroup元素了，如代码清单3-15所示。

代码清单3-15 hgroup元素示例

```
<article>
    <header>
        <hgroup>
            <h1>文章主标题</h1>
            <h2>文章子标题</h2>
        </hgroup>
        <p><time datetime="2010-03-20">2010年10月29日</time></p>
    </header>
    <p>文章正文</p>
</article>
```

## 3.2.3 footer元素

footer元素可以作为其上层父级内容区块或是一个根区块的脚注。footer通常包括其相关区块的脚注信息，如作者、相关阅读链接及版权信息等。

在HTML 5出现之前，我们使用下面的方式编写页脚，如代码清单3-16所示。

代码清单3-16 HTML 5之前的页脚示例

```
<div id="footer">
    <ul>
```

```
            <li>版权信息</li>
            <li>站点地图</li>
            <li>联系方式</li>
        </ul>
    <div>
```

但是到了HTML 5之后，这种方式将不再使用，而是使用更加语义化的footer元素来替代，如代码清单3-17所示。

代码清单3-17　footer元素示例

```
    <footer>
        <ul>
            <li>版权信息</li>
            <li>站点地图</li>
            <li>联系方式</li>
        </ul>
    </footer>
```

与header元素一样，一个页面中也未限制footer元素的个数。同时，可以为article元素或section元素添加footer元素，请看下面两个示例。

代码清单3-18为一个在article元素中添加footer元素的示例。

代码清单3-18　在article元素中添加footer元素

```
    <article>
        文章内容
        <footer>
            文章的脚注
        </footer>
    </article>
```

代码清单3-19为一个在section元素中添加footer元素的示例。

代码清单3-19　在section元素中添加footer元素

```
    <section>
        分段内容
        <footer>
            分段内容的脚注
        </footer>
    </section>
```

### 3.2.4　address元素

address元素用来在文档中呈现联系信息，包括文档作者或文档维护者的名字、他们的网站链接、电子邮箱、真实地址、电话号码等。address应该不只是用来呈现电子邮箱或真实地址，还应用来展示跟文档相关的联系人的所有联系信息。譬如，在代码清单3-20中，展示了一些博客中某篇文章评论者的名字及其在博客中的网址链接。

代码清单3-20　address元素示例

```
<address>
    <a href=http://blog.sina.com.cn/itnewtech>陆凌牛</a>
    <a href=http://blog.sina.com.cn/zhangyu>张玉</a>
    <a href=http://blog.sina.com.cn/baiquanli>白权立</a>
</address>
```

下面我们通过代码清单3-21来看如何把footer元素、time元素与address元素结合起来使用。

代码清单3-21　footer、time与address结合使用的例子

```
<footer>
    <div>
        <address>
           <a title="文章作者：陆凌牛" href="http://blog.sina.com.cn/itnewtech">
           陆凌牛</a>
        </address>
        发表于<time datetime="2010-10-04">2010年10月4日</time>
    </div>
</footer>
```

在这个示例中，把博客文章的作者、博客的主页链接作为作者信息放在了address元素中，把文章发表日期放在了time元素中，把这个address元素与time元素中的总体内容作为脚注信息放在了footer元素中。

## 3.3　HTML 5结构

前面两节中详细介绍了在HTML 5中具体新增了哪些结构元素，以及这些元素的定义和使用方法。在HTML 5中，可以使用这些结构元素来编排一份网页大纲，通过该网页大纲可以对网页中具有哪些内容，网页中以什么样的结构形式来组织这些内容有更清楚的认识。

### 3.3.1　大纲

在Word文档中，一份文档的大纲如图3-4所示。

HTML 5网页文档中的大纲也基本如此，只不过使用不同的结构元素进行表达而已。换句话说，在HTML 5中，使用各种结构元素所描述出来的整个网页的层次结构就是该网页的大纲。因此，在组织这份大纲时，不能使用div元素，因为div元素只能作为容器，在需要对网页中某个局部使用整体样式时才使用。

1. HTML 5的基础知识
1. 1 HTML 5概述
（第一章中第一节的正文）
1. 1. 1 HTML 5是什么？
（第一章中第一节第一小节的正文）
1. 1. 2 HTML 5中的新增API
（第一章中第一节第二小节的正文）

图3-4　Word中的文档大纲

可以通过许多工具对HTML 5的网页文档进行分析，然后将该文档中的大纲以可视化的形式展现出来。前面提到的“http://gsnedders.html5.org/outliner/”网站中就有一个文档大纲分析器，利用该分析器对代码清单3-22中的文档进行分析，结果如图3-5所示。

代码清单3-22　大纲分析工具测试用代码

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8">
<title>大纲分析</title>
</head>
<section>
    <h1>HTML5的基础知识</h1>
    <section>
        <h2>HTML5概述</h2>
        （正文）
        <section>
            <h3>HTML5是什么？</h3>
            （正文）
        </section>
        <section>
            <h3>HTML5中的新增API</h3>
            （正文）
        </section>
    </section>
    <section>
        <h2>HTML5快速入门</h2>
        （正文）
        <section>
            <h3>HTML与XHTML</h3>
            （正文）
        </section>
    </section>
</section>
```

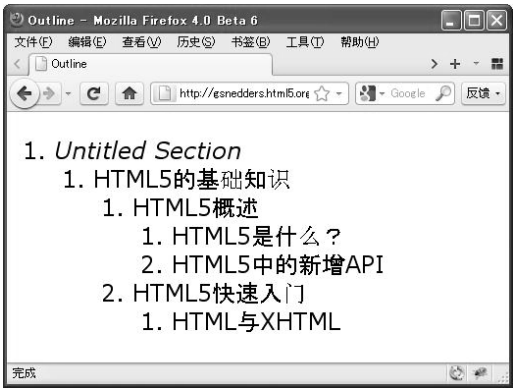


图3-5　在线大纲分析器的分析结果

图3-5中出现“ 1.Untitled Section ”，是由于该网页文档中第一个元素即为section元素，缺乏整个网页标题元素。加入标题元素（<h1>元素），将代码清单3-22中的代码修改为代码清单3-23所示的代码，分析出来的大纲如图3-6所示。

代码清单3-23　添加了标题元素后的大纲分析工具测试用代码

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8">
<title>大纲分析</title>
</head>
    <h1>HTML5的基础知识</h1>
<section>
    <h2>HTML5概述</h2>
     （正文）
    <section>
        <h3>HTML5是什么？</h3>
         （正文）
    </section>
    <section>
        <h3>HTML5中的新增API</h3>
         （正文）
    </section>
</section>
<section>
    <h2>HTML5快速入门</h2>
     （正文）
    <section>
        <h3>HTML与XHTML</h3>
         （正文）
    </section>
</section>
```

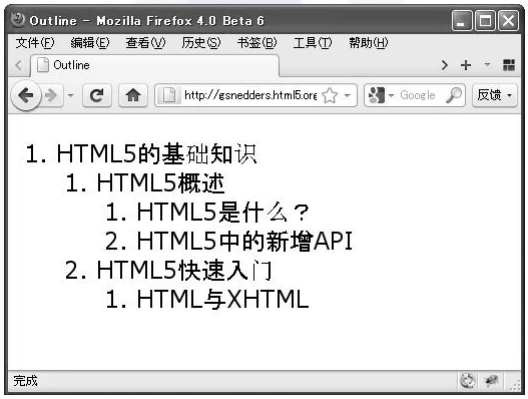


图3-6　加入网页标题后的页面大纲

看到这里，有读者也许会问：如果只加一个h1元素就可以分析出标题来，那还需要header元素干什么呢？答案是h1元素一般用来显示文字标题。事实上，在要定义为网页标题的整个内容中，往往不只显示一个文字而已，其中包括了大量的导航条、企业logo图片，广告Flash等，这些内容都可以放在一个header元素中作为整个网页标题的内容，而标题文字为h1元素中的文字，在大纲中显示该标题文字。但是，这里要说明的是，header元素本身的作用不是用来生成大纲的，大纲是依靠header元素中的h1～h6元素来生成的，如针对代码清单

3-24中的代码，生成的大纲如图3-7所示。

代码清单3-24　在企业网站中使用图片来显示企业名称

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8">
<title>企业网站</title>
</head>
<header>
    <img src="title.jpg" alt="用来显示企业名称的图片">
</header>
<section>
    <h2>企业描述</h2>
    （正文）
</section>
```

这里会有这样一个问题，在很多企业网站（或其他网站）中，企业的标题并不是以文字来显示的，而是为了提升视觉效果，放在图片中显示的。这种情况就不能生成大纲了吗？笔者在这里提供个小技巧，在header元素中，使用如下代码，既可以用图片来显示企业名称，又可以生成大纲。

```
<header>
    <h1><img src="title.jpg" alt="企业名称"></h1>
</header>
```

在header元素中使用这段代码后，生成的大纲如图3-8所示。


图3-7　header元素本身并不用来生成大纲


图3-8　在header元素中使用图片来生成大纲

与header元素相同，footer元素中如果没有标题元素（h1～h6元素）也不用来生成大纲。在代码清单3-22或代码清单3-24的代码底部追加如下代码，生成的大纲将不会有任何变化。

```
<footer>
    版权所有：陆凌牛
</footer>
```

另外，对于nav元素与aside元素来说，如果不在元素内部加入标题元素（h1～h6元素），那么在生成大纲时会在该元素所在位置处添加一个“Untitled Section”的说明文字。根据HTML 5开发文档中的记载，nav元素的作用为存放一组链接组，aside元素的作用为表示当

前页面或文章的附属信息部分，允许不在这两个元素中添加标题，也就是说，大纲中存在这两个元素的内容为“ Untitled Section ”的说明文字是合理的。

在代码清单3-22的代码底部添加如下代码，生成的大纲如图3-9所示。

```
<nav>
    <ul>
        <li><a href="#">链接测试1</a></li>
        <li><a href="#">链接测试2</a></li>
    </ul>
</nav>
<aside>
    侧边栏中的内容
</aside>
```

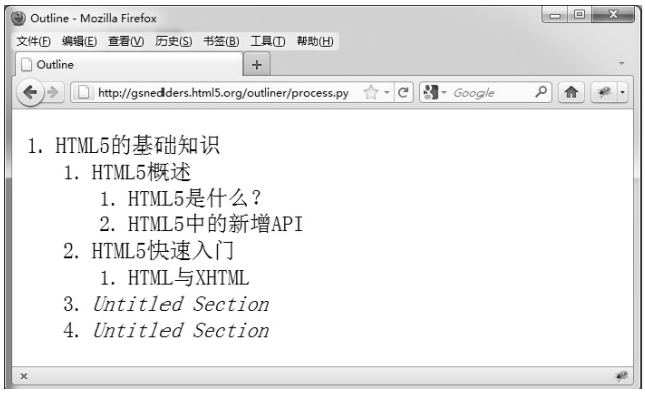


图3-9　在文档的底部添加nav元素与aside元素

另外，在HTML 5中，body元素、blockquote元素、fieldset元素、td元素、details元素及figure元素被称为节根（sectioning roots）元素。这些元素的共同特征是拥有自己独立的大纲，并且这些元素内的section元素、article元素、标题元素（h1～h6元素）、nav元素及aside元素等，只被用在生成其父元素的大纲时，而不被用在生成父元素的上层祖先元素的大纲时。

在代码清单3-25中，blockquote元素内部有一个h1元素，正是因为这个h1元素是位于节根元素blockquote元素内部的，所以在针对blockquote元素的父元素body元素生成页面大纲时，该h1元素并没有显示在大纲中，如图3-10所示。

代码清单3-25　针对body元素生成大纲时节根元素中的子元素不起作用

```
<!DOCTYPE html>
<meta charset="UTF-8">
<body>
<h1>网页标题</h1>
<blockquote>
   <h1>节根元素内部标题</h1>
</blockquote>
</body>
```

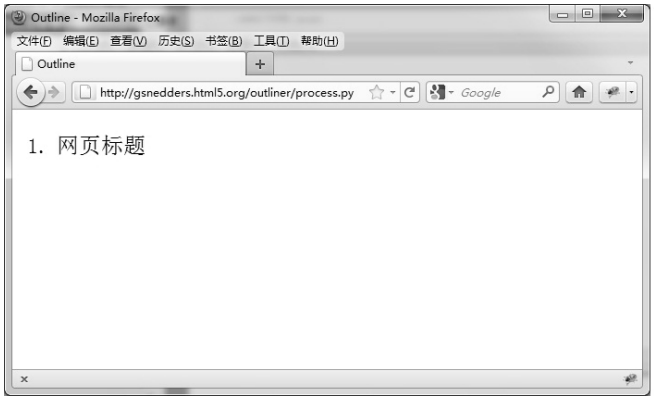


图3-10 针对body元素生成大纲时节根元素中的子元素不起作用

### 3.3.2 大纲的编排规则

关于内容区块的编排，可以分为“显式编排”与“隐式编排”两种方式。

1. 显式编排内容区块

显式编排是指明确使用section等元素创建文档结构，在每个内容区块内使用标题（h1～h6、hgroup等），如代码清单3-26所示。

代码清单3-26 显式编排内容区块示例

```
<body>
    <h1>网页级内容区块标题</h1>
    <p>网页级内容区块的正文</p>
    <section>
        <h2>section级内容区块的标题</h2>
        <p>section级内容区块的正文</p>
    </section>
</body>
```

2. 隐式编排内容区块

所谓隐式编排，是指不明确使用section等元素，而是根据页面中所书写的各级标题（h1～h6、hgroup等）把各级内容区块自动创建出来。因为HTML 5分析器只要看到书写了某个级别的标题，就会判断存在相对应的内容区块。代码清单3-27为一个隐式编排内容区块的示例。

代码清单3-27 隐式编排内容区块示例

```
<body>
    <h1>网页级内容区块标题</h1>
    <p>网页级内容区块的正文</p>
    <!--分析器根据h2等元素判断生成内容区块-->
    <h2>section级内容区块的标题</h2>
    <p>section级内容区块的正文</p>
</body>
```

将这两种编排方式进行对比，很明显，显式编排更加清晰、明确、易读。

3. 标题分级

不同的标题有不同的级别，h1的级别最高，h6的级别最低。在进行隐式编排时按如下规则自动生成内容区块。

❑ 如果新出现的标题比上一个标题级别低，将生成下级内容区块。

❑ 如果新出现的标题比上一个标题级别高，或者两者的级别相等，将生成新的内容区块。

第一条规则的示例与前面一样，现在来看关于第二条规则的示例，如代码清单3-28所示。

代码清单3-28　第二条规则的示例

```
<body>
<section>
    <h2>section级别的内容区块的标题</h2>
    <p>section级别的内容区块的正文</p>
    <!--因为下面的标题级别比上一个标题级别高，所以自动创建新的内容区块 -->
    <h1>新的section级别的内容区块的标题</h1>
    <p>新的section级别的内容区块的正文</p>
</section>
</body>
```

如果把上一个示例改成显式编排，如代码清单3-29所示。

代码清单3-29　第二条规则的显式编排示例

```
<body>
<section>
    <h2>section级别的内容区块的标题</h2>
    <p>section级别的内容区块的正文</p>
</section>
<section>
    <h1>新的section级别的内容区块的标题</h1>
    <p>新的section级别的内容区块的正文</p>
</section>
</body>
```

因为隐式编排容易使自动生成的整个文档结构与所想要的文档结构不一样，而且也容易引起文档结构的混乱，所以尽量使用显式编排。

4. 不同的内容区块可以使用相同级别的标题

另外，不同的内容区块可以使用相同级别的标题。例如，父内容区块与子内容区块可以使用相同级别的标题h1。这样做的好处是：每个级别的标题都可以单独设计，如果既需要“整个网页的标题”，又需要“文章的标题”（譬如书写文档时），这样做将会带来很大的便利性，如代码清单3-30所示。

代码清单3-30　不同的内容区块可以使用相同级别的标题

```
<body>
<h1>网页的标题</h1>
<article>
    <header>
        <hgroup>
```

```
            <h1>文章标题</h1>
            <h2>文章子标题</h2>
        </hgroup>
        <p>文章正文</p>
    </header>
</article>
</body>
```

5. 网页编排示例

基于以上讲解过的知识点，下面来看看应该怎样编排网页的内容。代码清单3-31为一个标准博客网页的示例，这个示例中有一个标准博客网页应具备的基本要素，只缺少为了使用样式而补充添加的div元素。

代码清单3-31　网页编排示例

```
<!DOCTYPE html>
<head>
  <title>网页编排示例</title>
  <meta charset="UTF-8">
</head>
<body>
<!-- 网页标题 -->
<header>
    <h1>网页标题</h1>
    <!-- 网站导航链接 -->
    <nav>
        <ul>
            <li><a href="index.html">首页</a></li>
            <li><a href="help.html">帮助</a></li>
        </ul>
    </nav>
</header>
<!-- 文章正文 -->
<article>
    <hgroup>
        <h1>文章主标题</h1>
        <h2>文章子标题</h2>
    </hgroup>
    <p>文章正文</p>
    <!--文章评论 -->
    <section class="comments">
        <article>
            <h1>评论标题</h1>
            <p>评论正文</p>
        </article>
    </section>
</article>
<!-- 版权信息 -->
<footer>
    <small>版权所有：陆凌牛</small>
</footer>
</body>
```

在这个示例中，使用了嵌套article元素的方式，将关于评论的article元素嵌套在了主

article元素中。在HTML 5中，推荐使用这种嵌套article元素的方式。

### 3.3.3 对新的结构元素使用样式

因为很多浏览器尚未对HTML 5中新增的结构元素提供支持，我们无法知道客户端使用的浏览器是否支持这些元素，所以需要使用CSS追加如下声明，目的是通知浏览器页面中使用的HTML 5中新增元素都以块方式显示。

```
//追加block声明
article, aside, dialog, figure, footer, header, legend, nav, section
{   display: block; }
//正常使用样式
nav{float;left;width:20%;}
article{float:right;width:79%;}
```

另外，IE 8及之前的浏览器是不支持用CSS的方法来使用这些尚未支持的结构元素的，为了在IE浏览器中正常使用这些结构元素，需要使用JavaScript脚本，如下所示：

```
//在脚本中创建元素
<script>
document.createElement("header");
document.createElement("nav");
document.createElement("article");
document.createElement("footer");
</script>
<style>
//正常使用样式
nav{float;left;width:20%;}
article{float:right;width:79%;}
</style>
```

虽然这段JavaScript脚本在其他浏览器中是不需要的，但是它不会对这些浏览器造成不良影响。另外，在IE 9之后，就不需要这段脚本了。