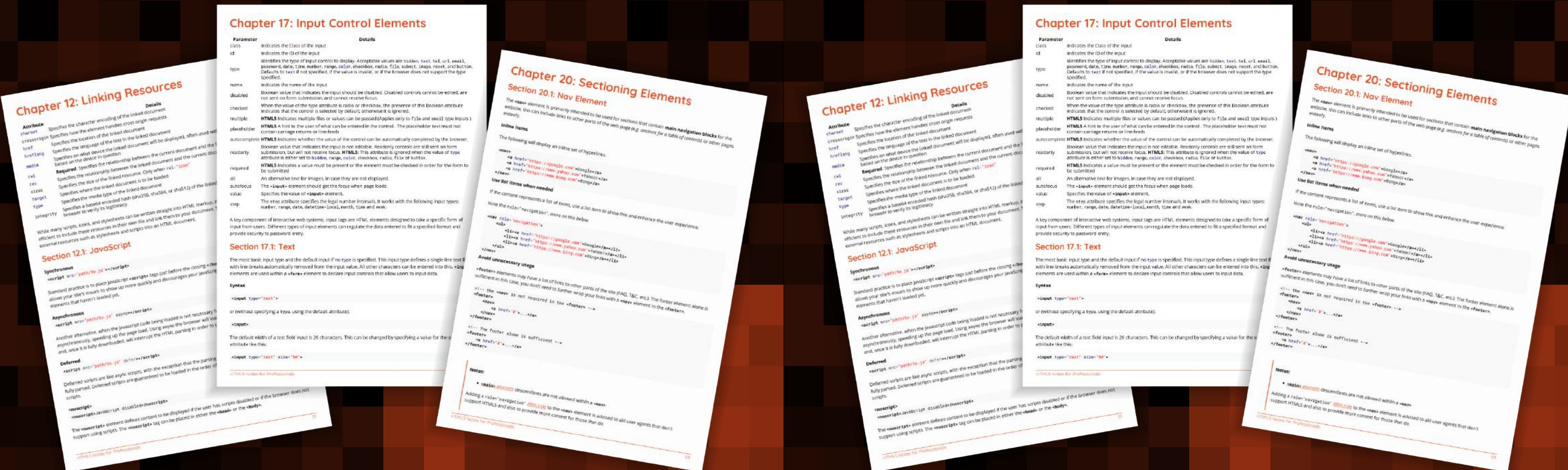


HTML5

专业人士笔记

HTML5

Notes for Professionals



100多页

专业提示和技巧

100+ pages

of professional hints and tricks

目录

关于	1
第1章：HTML入门	2
第1.1节：你好，世界	2
第2章：文档类型声明	5
第2.1节：添加文档类型声明	5
第2.2节：HTML 5文档类型声明	5
第3章：标题	6
第3.1节：使用标题	6
第4章：段落	7
第4.1节：HTML段落	7
第5章：文本格式	8
第5.1节：高亮显示	8
第5.2节：加粗、斜体和下划线	8
第5.3节：缩写	9
第5.4节：插入、删除或删除线	9
第5.5节：上标和下标	9
第6章：锚点和超链接	11
第6.1节：链接到另一个网站	11
第6.2节：链接到锚点	12
第6.3节：链接到同一网站的页面	12
第6.4节：拨打电话号码的链接	12
第6.5节：在新标签页/窗口中打开链接	13
第6.6节：运行JavaScript的链接	13
第6.7节：运行电子邮件客户端的链接	14
第七章：列表	15
第7.1节：有序列表	15
第7.2节：无序列表	16
第7.3节：嵌套列表	17
第7.4节：描述列表	17
第8章：表格	19
第8.1节：简单表格	19
第8.2节：跨列或跨行	19
第8.3节：列组	20
第8.4节：带thead、tbody、tfoot和标题的表格	21
第8.5节：标题范围	22
第9章：注释	24
第9.1节：创建注释	24
第9.2节：注释内联元素之间的空白	24
第10章：类和ID	26
第10.1节：给元素赋予类	26
第10.2节：给元素赋予ID	27
第10.3节：可接受的值	27
第10.4节：与重复ID相关的问题	29
第11章：数据属性	30
第11.1节：旧版浏览器支持	30
第11.2节：数据属性的使用	30

Contents

About	1
Chapter 1: Getting started with HTML	2
Section 1.1: Hello World	2
Chapter 2: Doctypes	5
Section 2.1: Adding the Doctype	5
Section 2.2: HTML 5 Doctype	5
Chapter 3: Headings	6
Section 3.1: Using Headings	6
Chapter 4: Paragraphs	7
Section 4.1: HTML Paragraphs	7
Chapter 5: Text Formatting	8
Section 5.1: Highlighting	8
Section 5.2: Bold, Italic, and Underline	8
Section 5.3: Abbreviation	9
Section 5.4: Inserted, Deleted, or Stricken	9
Section 5.5: Superscript and Subscript	9
Chapter 6: Anchors and Hyperlinks	11
Section 6.1: Link to another site	11
Section 6.2: Link to an anchor	12
Section 6.3: Link to a page on the same site	12
Section 6.4: Link that dials a number	12
Section 6.5: Open link in new tab/window	13
Section 6.6: Link that runs JavaScript	13
Section 6.7: Link that runs email client	14
Chapter 7: Lists	15
Section 7.1: Ordered List	15
Section 7.2: Unordered List	16
Section 7.3: Nested lists	17
Section 7.4: Description List	17
Chapter 8: Tables	19
Section 8.1: Simple Table	19
Section 8.2: Spanning columns or rows	19
Section 8.3: Column Groups	20
Section 8.4: Table with thead, tbody, tfoot, and caption	21
Section 8.5: Heading scope	22
Chapter 9: Comments	24
Section 9.1: Creating comments	24
Section 9.2: Commenting out whitespace between inline elements	24
Chapter 10: Classes and IDs	26
Section 10.1: Giving an element a class	26
Section 10.2: Giving an element an ID	27
Section 10.3: Acceptable Values	27
Section 10.4: Problems related to duplicated IDs	29
Chapter 11: Data Attributes	30
Section 11.1: Older browsers support	30
Section 11.2: Data Attribute Use	30

第12章：链接资源	31
第12.1节：JavaScript	31
第12.2节：外部CSS样式表	32
第12.3节：网站图标（Favicon）	32
第12.4节：替代CSS	32
第12.5节：资源提示：dns-prefetch、prefetch、prerender	33
第12.6节：链接的“media”属性	33
第12.7节：上一页和下一页	33
第12.8节：网络订阅	33
第13章：在HTML中包含JavaScript代码	35
第13.1节：处理禁用的JavaScript	35
第13.2节：链接到外部JavaScript文件	35
第13.3节：直接包含JavaScript代码	35
第13.4节：异步执行的JavaScript文件包含	35
第14章：使用带有CSS的HTML	36
第14.1节：外部样式表的使用	36
第14.2节：内部样式表	36
第14.3节：内联样式	37
第14.4节：多重样式表	37
第15章：图像	38
第15.1节：创建图像	38
第15.2节：选择替代文本	38
第15.3节：使用srcset属性的响应式图像	39
第15.4节：使用picture元素的响应式图像	40
第16章：图像映射	41
第16.1节：图像映射简介	41
第17章：输入控件元素	43
第17.1节：文本	43
第17.2节：复选框和单选按钮	44
第17.3节：输入验证	46
第17.4节：颜色	47
第17.5节：密码	48
第17.6节：文件	48
第17.7节：按钮	49
第17.8节：提交	50
第17.9节：重置	50
第17.10节：隐藏	50
第17.11节：电话	50
第17.12节：电子邮件	51
第17.13节：数字	51
第17.14节：范围	51
第17.15节：搜索	51
第17.16节：图像	51
第17.17节：周	52
第17.18节：网址	52
第17.19节：本地日期时间	52
第17.20节：月份	52
第17.21节：时间	52
第17.22节：日期时间（全球）	53
第17.23节：日期	53

Chapter 12: Linking Resources	31
Section 12.1: JavaScript	31
Section 12.2: External CSS Stylesheet	32
Section 12.3: Favicon	32
Section 12.4: Alternative CSS	32
Section 12.5: Resource Hint: dns-prefetch, prefetch, prerender	33
Section 12.6: Link 'media' attribute	33
Section 12.7: Prev and Next	33
Section 12.8: Web Feed	33
Chapter 13: Include JavaScript Code in HTML	35
Section 13.1: Handling disabled Javascript	35
Section 13.2: Linking to an external JavaScript file	35
Section 13.3: Directly including JavaScript code	35
Section 13.4: Including a JavaScript file executing asynchronously	35
Chapter 14: Using HTML with CSS	36
Section 14.1: External Stylesheet Use	36
Section 14.2: Internal Stylesheet	36
Section 14.3: Inline Style	37
Section 14.4: Multiple Stylesheets	37
Chapter 15: Images	38
Section 15.1: Creating an image	38
Section 15.2: Choosing alt text	38
Section 15.3: Responsive image using the srcset attribute	39
Section 15.4: Responsive image using picture element	40
Chapter 16: Image Maps	41
Section 16.1: Introduction to Image Maps	41
Chapter 17: Input Control Elements	43
Section 17.1: Text	43
Section 17.2: Checkbox and Radio Buttons	44
Section 17.3: Input Validation	46
Section 17.4: Color	47
Section 17.5: Password	48
Section 17.6: File	48
Section 17.7: Button	49
Section 17.8: Submit	50
Section 17.9: Reset	50
Section 17.10: Hidden	50
Section 17.11: Tel	50
Section 17.12: Email	51
Section 17.13: Number	51
Section 17.14: Range	51
Section 17.15: Search	51
Section 17.16: Image	51
Section 17.17: Week	52
Section 17.18: Url	52
Section 17.19: DateTime-Local	52
Section 17.20: Month	52
Section 17.21: Time	52
Section 17.22: DateTime (Global)	53
Section 17.23: Date	53

第18章：表单	54
第18.1节：提交	54
第18.2节：表单标签中的target属性	55
第18.3节：上传文件	55
第18.4节：分组若干输入字段	55
第19章：Div元素	57
第19.1节：基本用法	57
第19.2节：嵌套	57
第20章：分区元素	59
第20.1节：导航元素	59
第20.2节：文章元素	60
第20.3节：主体元素	61
第20.4节：页眉元素	62
第20.5节：页脚元素	63
第20.6节：区块元素	63
第21章：导航栏	64
第21.1节：基础导航栏	64
第21.2节：HTML5导航栏	64
第22章：标签元素	65
第22.1节：关于标签	65
第22.2节：基本用法	65
第23章：输出元素	67
第23.1节：使用For和Form属性的输出元素	67
第23.2节：带属性的输出元素	67
第24章：空元素	68
第24.1节：空元素	68
第25章：媒体元素	69
第25.1节：音频	69
第25.2节：视频	69
第25.3节：使用`<video>`和`<audio>`元素显示音频/视频内容	69
第25.4节：视频头部或背景	70
第26章：进度元素	71
第26.1节：进度	71
第26.2节：更改进度条颜色	71
第26.3节：HTML回退	72
第27章：选择菜单控件	73
第27.1节：选择菜单	73
第27.2节：选项	73
第27.3节：选项组	74
第27.4节：数据列表	74
第28章：嵌入	76
第28.1节：基本用法	76
第28.2节：定义MIME类型	76
第29章：内联框架（IFrame）	77
第29.1节：内联框架基础	77
第29.2节：沙箱	77
第29.3节：设置框架大小	77
第29.4节：使用“srcdoc”属性	78

Chapter 18: Forms	54
Section 18.1: Submitting	54
Section 18.2: Target attribute in form tag	55
Section 18.3: Uploading Files	55
Section 18.4: Grouping a few input fields	55
Chapter 19: Div Element	57
Section 19.1: Basic usage	57
Section 19.2: Nesting	57
Chapter 20: Sectioning Elements	59
Section 20.1: Nav Element	59
Section 20.2: Article Element	60
Section 20.3: Main Element	61
Section 20.4: Header Element	62
Section 20.5: Footer Element	63
Section 20.6: Section Element	63
Chapter 21: Navigation Bars	64
Section 21.1: Basic Navigation Bar	64
Section 21.2: HTML5 Navigation Bar	64
Chapter 22: Label Element	65
Section 22.1: About Label	65
Section 22.2: Basic Use	65
Chapter 23: Output Element	67
Section 23.1: Output Element Using For and Form Attributes	67
Section 23.2: Output Element with Attributes	67
Chapter 24: Void Elements	68
Section 24.1: Void elements	68
Chapter 25: Media Elements	69
Section 25.1: Audio	69
Section 25.2: Video	69
Section 25.3: Using `<video>` and `<audio>` element to display audio/video content	69
Section 25.4: Video header or background	70
Chapter 26: Progress Element	71
Section 26.1: Progress	71
Section 26.2: Changing the color of a progress bar	71
Section 26.3: HTML Fallback	72
Chapter 27: Selection Menu Controls	73
Section 27.1: Select Menu	73
Section 27.2: Options	73
Section 27.3: Option Groups	74
Section 27.4: Datalist	74
Chapter 28: Embed	76
Section 28.1: Basic usage	76
Section 28.2: Defining the MIME type	76
Chapter 29: IFrames	77
Section 29.1: Basics of an Inline Frame	77
Section 29.2: Sandboxing	77
Section 29.3: Setting the Frame Size	77
Section 29.4: Using the "srcdoc" Attribute	78

第29.5节：使用带锚点的内联框架	78
第30章：内容语言	79
第30.1节：基础文档语言	79
第30.2节：元素语言	79
第30.3节：多语言元素	79
第30.4节：区域性网址	79
第30.5节：处理不同语言的属性	79
第31章：SVG	81
第31.1节：内联SVG	81
第31.2节：在HTML中嵌入外部SVG文件	81
第31.3节：使用CSS嵌入SVG	82
第32章：画布（Canvas）	83
第32.1节：基本示例	83
第32.2节：在<canvas>上绘制两个矩形	83
第33章：元信息	85
第33.1节：页面信息	85
第33.2节：字符编码	85
第33.3节：机器人	86
第33.4节：社交媒体	86
第33.5节：移动布局控制	87
第33.6节：自动刷新	88
第33.7节：电话号码识别	88
第33.8节：自动重定向	88
第33.9节：网页应用	89
第34章：计算机代码的标记	90
第34.1节：带有<pre>和<code>的块	90
第34.2节：带有<code>的行内	90
第35章：引用标记	91
第35.1节：带有<q>的行内	91
第35.2节：带有<blockquote>的块	91
第36章：Tabindex	93
第36.1节：向制表顺序中添加元素	93
第36.2节：从制表顺序中移除元素	93
第36.3节：定义自定义制表顺序（不推荐）	93
第37章：全局属性	94
第37.1节：Contenteditable属性	94
第38章：HTML 5缓存	95
第38.1节：HTML5缓存的基本示例	95
第39章：HTML事件属性	96
第39.1节：HTML表单事件	96
第39.2节：键盘事件	96
第40章：字符实体	97
第40.1节：HTML中的字符实体	97
第40.2节：常见特殊字符	97
第41章：ARIA	98
第41.1节：role="presentation"	98
第41.2节：role="alert"	98
第41.3节：role="alertdialog"	98

Section 29.5: Using Anchors with IFrames	78
Chapter 30: Content Languages	79
Section 30.1: Base Document Language	79
Section 30.2: Element Language	79
Section 30.3: Elements with Multiple Languages	79
Section 30.4: Regional URLs	79
Section 30.5: Handling Attributes with Different Languages	79
Chapter 31: SVG	81
Section 31.1: Inline SVG	81
Section 31.2: Embedding external SVG files in HTML	81
Section 31.3: Embedding SVG using CSS	82
Chapter 32: Canvas	83
Section 32.1: Basic Example	83
Section 32.2: Drawing two rectangles on a <canvas>	83
Chapter 33: Meta Information	85
Section 33.1: Page Information	85
Section 33.2: Character Encoding	85
Section 33.3: Robots	86
Section 33.4: Social Media	86
Section 33.5: Mobile Layout Control	87
Section 33.6: Automatic Refresh	88
Section 33.7: Phone Number Recognition	88
Section 33.8: Automatic redirect	88
Section 33.9: Web App	89
Chapter 34: Marking up computer code	90
Section 34.1: Block with <pre> and <code>	90
Section 34.2: Inline with <code>	90
Chapter 35: Marking-up Quotes	91
Section 35.1: Inline with <q>	91
Section 35.2: Block with <blockquote>	91
Chapter 36: Tabindex	93
Section 36.1: Add an element to the tabbing order	93
Section 36.2: Remove an element from the tabbing order	93
Section 36.3: Define a custom tabbing order (not recommended)	93
Chapter 37: Global Attributes	94
Section 37.1: Contenteditable Attribute	94
Chapter 38: HTML 5 Cache	95
Section 38.1: Basic Example of HTML5 cache	95
Chapter 39: HTML Event Attributes	96
Section 39.1: HTML Form Events	96
Section 39.2: Keyboard Events	96
Chapter 40: Character Entities	97
Section 40.1: Character Entities in HTML	97
Section 40.2: Common Special Characters	97
Chapter 41: ARIA	98
Section 41.1: role="presentation"	98
Section 41.2: role="alert"	98
Section 41.3: role="alertdialog"	98

第41.4节：role="application"	98
第41.5节：role="article"	98
第41.6节：role="banner"	99
第41.7节：role="button"	99
第41.8节：role="cell"	99
第41.9节：role="checkbox"	99
第41.10节：role="columnheader"	100
第41.11节：role="combobox"	100
第41.12节：role="complementary"	100
第41.13节：role="contentinfo"	100
第41.14节：role="definition"	100
第41.15节：role="dialog"	101
第41.16节：role="directory"	101
第41.17节：role="document"	101
第41.18节：role="form"	101
第41.19节：role="grid"	102
第41.20节：role="gridcell"	102
第41.21节：role="group"	102
第41.22节：role="heading"	102
第41.23节：role="img"	103
第41.24节：角色="link"	103
第41.25节：角色="list"	103
第41.26节：角色="listbox"	103
第41.27节：角色="listitem"	103
第41.28节：角色="log"	104
第41.29节：角色="main"	104
第41.30节：角色="marquee"	104
第41.31节：角色="math"	104
第41.32节：role="menu"	104
第41.33节：role="menubar"	104
第41.34节：role="menuitem"	105
第41.35节：role="menuitemcheckbox"	105
第41.36节：role="menuitemradio"	105
第41.37节：role="navigation"	105
第41.38节：role="note"	105
第41.39节：role="option"	105
第41.40节：role="progressbar"	106
第41.41节：role="radio"	106
第41.42节：role="region"	106
第41.43节：role="radiogroup"	106
第41.44节：role="row"	106
第41.45节：role="rowgroup"	107
第41.46节：role="rowheader"	107
第41.47节：role="scrollbar"	107
第41.48节：role="search"	107
第41.49节：role="searchbox"	108
第41.50节：角色="分隔符"	108
第41.51节：角色="滑块"	108
第41.52节：角色="微调按钮"	108
第41.53节：角色="状态"	108
第41.54节：角色="开关"	108

Section 41.4: role="application"	98
Section 41.5: role="article"	98
Section 41.6: role="banner"	99
Section 41.7: role="button"	99
Section 41.8: role="cell"	99
Section 41.9: role="checkbox"	99
Section 41.10: role="columnheader"	100
Section 41.11: role="combobox"	100
Section 41.12: role="complementary"	100
Section 41.13: role="contentinfo"	100
Section 41.14: role="definition"	100
Section 41.15: role="dialog"	101
Section 41.16: role="directory"	101
Section 41.17: role="document"	101
Section 41.18: role="form"	101
Section 41.19: role="grid"	102
Section 41.20: role="gridcell"	102
Section 41.21: role="group"	102
Section 41.22: role="heading"	102
Section 41.23: role="img"	103
Section 41.24: role="link"	103
Section 41.25: role="list"	103
Section 41.26: role="listbox"	103
Section 41.27: role="listitem"	103
Section 41.28: role="log"	104
Section 41.29: role="main"	104
Section 41.30: role="marquee"	104
Section 41.31: role="math"	104
Section 41.32: role="menu"	104
Section 41.33: role="menubar"	104
Section 41.34: role="menuitem"	105
Section 41.35: role="menuitemcheckbox"	105
Section 41.36: role="menuitemradio"	105
Section 41.37: role="navigation"	105
Section 41.38: role="note"	105
Section 41.39: role="option"	105
Section 41.40: role="progressbar"	106
Section 41.41: role="radio"	106
Section 41.42: role="region"	106
Section 41.43: role="radiogroup"	106
Section 41.44: role="row"	106
Section 41.45: role="rowgroup"	107
Section 41.46: role="rowheader"	107
Section 41.47: role="scrollbar"	107
Section 41.48: role="search"	107
Section 41.49: role="searchbox"	108
Section 41.50: role="separator"	108
Section 41.51: role="slider"	108
Section 41.52: role="spinbutton"	108
Section 41.53: role="status"	108
Section 41.54: role="switch"	108

第41.55节：角色="标签"	109
第41.56节：角色="表格"	109
第41.57节：角色="标签列表"	109
第41.58节：role="tabpanel"	109
第41.59节：角色="文本框"	109
第41.60节：角色="计时器"	110
第41.61节：角色="工具栏"	110
第41.62节：角色="工具提示"	110
第41.63节：角色="树形结构"	110
第41.64节：角色="树形网格"	111
第41.65节：角色="树形项目"	111
鸣谢	112
你可能也喜欢	117

Section 41.55: role="tab"	109
Section 41.56: role="table"	109
Section 41.57: role="tablist"	109
Section 41.58: role="tabpanel"	109
Section 41.59: role="textbox"	109
Section 41.60: role="timer"	110
Section 41.61: role="toolbar"	110
Section 41.62: role="tooltip"	110
Section 41.63: role="tree"	110
Section 41.64: role="treegrid"	111
Section 41.65: role="treeitem"	111
Credits	112
You may also like	117

欢迎随意免费分享此PDF，
本书的最新版本可从以下网址下载：
<https://goalkicker.com/HTML5Book>

这本《专业人士的HTML5笔记》汇编自[Stack Overflow Documentation](#)，内容由Stack Overflow的优秀人士撰写。
文本内容采用知识共享署名-相同方式共享许可协议发布，详见本书末尾对各章节贡献者的致谢。图片版权归各自所有者所有，除非另有说明。

这是一本非官方的免费书籍，旨在用于教育目的，与官方HTML5组织或公司以及Stack Overflow无关。所有商标和注册商标均为其各自公司所有者的财产。

本书中提供的信息不保证正确或准确，使用风险自负

请将反馈和更正发送至web@petercv.com

Please feel free to share this PDF with anyone for free,
latest version of this book can be downloaded from:
<https://goalkicker.com/HTML5Book>

This *HTML5 Notes for Professionals* book is compiled from [Stack Overflow Documentation](#), the content is written by the beautiful people at Stack Overflow. Text content is released under Creative Commons BY-SA, see credits at the end of this book whom contributed to the various chapters. Images may be copyright of their respective owners unless otherwise specified

This is an unofficial free book created for educational purposes and is not affiliated with official HTML5 group(s) or company(s) nor Stack Overflow. All trademarks and registered trademarks are the property of their respective company owners

The information presented in this book is not guaranteed to be correct nor accurate, use at your own risk

Please send feedback and corrections to web@petercv.com

第1章：HTML入门

版本	规范	发布日期
1.0	不适用	1994-01-01
2.0	RFC 1866	1995-11-24
3.2	W3C：HTML 3.2 规范	1997-01-14
4.0	W3C：HTML 4.0 规范	1998-04-24
4.01	W3C：HTML 4.01 规范	1999-12-24
5	WHATWG：HTML Living Standard	2014-10-28
5.1	W3C：HTML 5.1 规范	2016-11-01

第1.1节：你好，世界

介绍

[HTML](#)（超文本标记语言）使用由元素组成的标记系统，这些元素表示特定的内容。标记意味着使用HTML你声明什么呈现给观众，而不是如何呈现。视觉表现由层叠样式表（CSS）定义，并由浏览器实现。仍然存在允许此类的元素，例如font，“完全过时，作者不得使用”[1]。

HTML有时被称为编程语言，但它没有逻辑，因此是**标记语言**。HTML标签为页面中的内容提供语义意义和机器可读性。

一个元素通常由一个起始标签（<element_name>）、一个结束标签（</element_name>）组成，标签中包含元素名称，名称被尖括号包围，中间是内容：
<element_name>...内容...</element_name>

有些HTML元素没有结束标签或任何内容。这些称为空元素。空元素包括、<meta>、<link>和<input>。

元素名称可以被视为其包含内容的描述性关键词，例如video、audio、table、footer。

一个HTML页面可能包含数百个元素，随后由网页浏览器读取、解释并渲染为屏幕上人类可读或可听的内容。

对于本文档，重要的是要注意元素和标签之间的区别：

元素：视频，音频，表格，页脚

标签：<video>, <audio>, <table>, <footer>, </html>, </body>

元素解析

让我们分解一个标签.....

<p> 标签表示一个普通段落。

元素通常有一个开始标签和一个结束标签。开始标签包含元素名称，位于尖括号内

Chapter 1: Getting started with HTML

Version	Specification	Release Date
1.0	N/A	1994-01-01
2.0	RFC 1866	1995-11-24
3.2	W3C: HTML 3.2 Specification	1997-01-14
4.0	W3C: HTML 4.0 Specification	1998-04-24
4.01	W3C: HTML 4.01 Specification	1999-12-24
5	WHATWG: HTML Living Standard	2014-10-28
5.1	W3C: HTML 5.1 Specification	2016-11-01

Section 1.1: Hello World

Introduction

[HTML](#) (**H**ypertext **M**arkup **L**anguage) uses a markup system composed of elements which represent specific content. *Markup* means that with HTML you declare *what* is presented to a viewer, not *how* it is presented. Visual representations are defined by [Cascading Style Sheets \(CSS\)](#) and realized by browsers. [Still existing elements that allow for such](#), like e.g. [font](#), "are entirely obsolete, and must not be used by authors"[1].

HTML is sometimes called a programming language but it has no logic, so is a **markup language**. HTML tags provide semantic meaning and machine-readability to the content in the page.

An element usually consists of an opening tag (<**element_name**>), a closing tag (</**element_name**>), which contain the element's name surrounded by angle brackets, and the content in between:
<element_name>...content...</element_name>

There are some HTML elements that don't have a closing tag or any contents. These are called void elements. Void elements include ****, **<meta>**, **<link>** and **<input>**.

Element names can be thought of as descriptive keywords for the content they contain, such as video, audio, table, footer.

A HTML page may consist of potentially hundreds of elements which are then read by a web browser, interpreted and rendered into human readable or audible content on the screen.

For this document it is important to note the difference between elements and tags:

Elements: video, audio, table, footer

Tags: <video>, <audio>, <table>, <footer>, </html>, </body>

Element insight

Let's break down a tag...

The **<p>** tag represents a common paragraph.

Elements commonly have an opening tag and a closing tag. The opening tag contains the element's name in angle

（<p>）。结束标签与开始标签相同，只是在开始括号和元素名称之间添加了一个斜杠（/）（</p>）。

内容可以放在这两个标签之间：<p>这是一个简单的段落。</p>。

创建一个简单页面

下面的HTML示例创建了一个简单的“Hello World”网页。

HTML 文件可以使用任何文本编辑器创建。文件必须保存为.html或.htm[2]扩展名，才能被识别为 HTML 文件。

一旦创建，该文件可以在任何网页浏览器中打开。

```
<!DOCTYPE html>
<html lang="en">

  <head>
    <meta charset="UTF-8">
    <title>你好！</title>
  </head>

  <body>
    <h1>你好，世界！</h1>
    <p>这是一个简单的段落。</p>
  </body>

</html>
```

简单的页面拆解

以下是示例中使用的标签：

标签	含义
<!DOCTYPE>	定义文档中使用的HTML版本。本例中为HTML5。 有关更多信息，请参见文档类型主题。
<html>	打开页面。关闭标签（</html>）后不应有任何标记。lang属性使用ISO语言代码声明页面的主要语言（例如英语为en）。 有关更多信息，请参见内容语言主题。
<head>	打开头部部分，该部分不会显示在主浏览器窗口中，主要包含关于HTML文档的信息，称为元数据。它还可以包含外部样式表和脚本的导入。关闭标签为</head>。
<meta >	为浏览器提供有关文档的一些元数据。charset属性声明字符编码。现代HTML文档应始终使用UTF-8，尽管这不是强制要求。在HTML中，<meta>标签不需要关闭标签。 有关更多信息，请参见Meta主题。
<title>	页面标题。在此开始标签和结束标签（</title>）之间书写的文本将显示在页面的标签页或浏览器的标题栏中。
<body>	打开显示给用户的文档部分，即页面所有可见或可听内容。关闭标签</body>之后不应添加任何内容。

brackets (<p>). The closing tag is identical to the opening tag with the addition of a forward slash (/) between the opening bracket and the element's name (</p>).

Content can then go between these two tags: <p>This is a simple paragraph.</p>.

Creating a simple page

The following HTML example creates a simple "Hello World" web page.

HTML files can be created using any text editor. The files must be saved with a .html or .htm[2] extension in order to be recognized as HTML files.

Once created, this file can be opened in any web browser.

```
<!DOCTYPE html>
<html lang="en">

  <head>
    <meta charset="UTF-8">
    <title>Hello!</title>
  </head>

  <body>
    <h1>Hello World!</h1>
    <p>This is a simple paragraph.</p>
  </body>

</html>
```

Simple page break down

These are the tags used in the example:

Tag	Meaning
<!DOCTYPE>	Defines the HTML version used in the document. In this case it is HTML5. See the doctypes topic for more information.
<html>	Opens the page. No markup should come after the closing tag (</html>). The lang attribute declares the primary language of the page using the ISO language codes (en for English). See the Content Language topic for more information.
<head>	Opens the head section, which does not appear in the main browser window but mainly contains information about the HTML document, called metadata. It can also contain imports from external stylesheets and scripts. The closing tag is </head>.
<meta>	Gives the browser some metadata about the document. The charset attribute declares the character encoding. Modern HTML documents should always use UTF-8, even though it is not a requirement. In HTML, the <meta> tag does not require a closing tag. See the Meta topic for more information.
<title>	The title of the page. Text written between this opening and the closing tag (</title>) will be displayed on the tab of the page or in the title bar of the browser.
<body>	Opens the part of the document displayed to users, i.e. all the visible or audible content of a page. No content should be added after the closing tag </body>.

<h1> 页面的一级标题。
有关更多信息，请参见标题。

<p> 表示普通文本段落。

- 1. ↑ [HTML5，第11.2节 不符合规范的特性](#)
- 2. ↑ .htm继承自传统DOS的三字符文件扩展名限制。

<h1> A level 1 heading for the page.
See headings for more information.

<p> Represents a common paragraph of text.

- 1. ↑ [HTML5, 11.2 Non-conforming features](#)
- 2. ↑ .htm is inherited from the legacy [DOS](#) three character file extension limit.

第2章：文档类型

文档类型（Doctypes）是“document type”的缩写，帮助浏览器理解文档所使用的HTML版本，以便更好地解析。文档类型声明不是HTML标签，应放在文档的最顶部。本节介绍HTML中各种文档类型的结构和声明方式。

第2.1节：添加文档类型

<!DOCTYPE>声明应始终包含在HTML文档的顶部，位于<html>标签之前。

版本 ≥ 5

有关HTML 5文档类型的详细信息，请参见HTML 5 Doctype。

```
<!DOCTYPE html>
```

第2.2节：HTML 5文档类型

HTML5不基于SGML（标准通用标记语言），因此不需要引用DTD（文档类型定义）。

HTML 5 文档类型声明：

```
<!DOCTYPE html>
```

大小写不敏感

根据[W3.org HTML 5DOCTYPE规范](#)：

DOCTYPE必须由以下组件组成，且顺序如下：

1. 一个字符串，该字符串与"<!DOCTYPE"字符串进行ASCII大小写不敏感匹配。

因此，以下DOCTYPE也是有效的：

```
<!doctype html>
<!dOcTyPe html>
<!DocTYpe html>
```

这篇SO文章对此话题进行了详细讨论：[大写还是小写doctype？](#)

Chapter 2: Doctypes

Doctypes - short for 'document type' - help browsers to understand the version of HTML the document is written in for better interpretability. Doctype declarations are not HTML tags and belong at the very top of a document. This topic explains the structure and declaration of various doctypes in HTML.

Section 2.1: Adding the Doctype

The `<!DOCTYPE>` declaration should always be included at the top of the HTML document, before the `<html>` tag.

Version ≥ 5

See HTML 5 Doctype for details on the HTML 5 Doctype.

```
<!DOCTYPE html>
```

Section 2.2: HTML 5 Doctype

HTML5 is not based on SGML (Standard Generalized Markup Language), and therefore does not require a reference to a DTD (Document Type Definition).

HTML 5 Doctype declaration:

```
<!DOCTYPE html>
```

Case Insensitivity

Per the [W3.org HTML 5 DOCTYPE Spec](#):

A DOCTYPE must consist of the following components, in this order:

1. A string that is an ASCII **case-insensitive** match for the string `"<!DOCTYPE"`.

therefore the following DOCTYPEs are also valid:

```
<!doctype html>
<!dOcTyPe html>
<!DocTYpe html>
```

This SO article discusses the topic extensively: [Uppercase or lowercase doctype?](#)

第3章：标题

HTML不仅提供了普通的段落标签，还提供了六个不同的标题标签，用于表示不同大小和粗细的标题。编号为标题1到标题6，标题1的文字最大且最粗，而标题6则最小最细，接近段落级别。本文详细说明了这些标签的正确用法。

第3.1节：使用标题

标题可用于描述其前面的主题，使用<h1>到<h6>标签定义。标题支持所有全局属性。

- <h1>定义最重要的标题。
- <h6>定义最不重要的标题。

定义标题：

```
<h1>标题1</h1>
<h2>标题2</h2>
<h3>标题3</h3>
<h4>标题4</h4>
<h5>标题5</h5>
<h6>标题6</h6>
```

正确的结构很重要

搜索引擎和其他用户代理通常基于标题元素索引页面内容，例如用于创建目录，因此使用正确的标题结构很重要。

一般来说，一篇文章应有一个h1元素作为主标题，后面跟随h2子标题-如有必要，可逐层向下。如果在更高层级使用了h1元素，则不应将其用于描述任何较低层级的内容。

示例文档（额外缩进以示层级关系）：

```
<h1>主标题</h1>
<p>介绍</p>

  <h2>原因</h2>

    <h3>原因 1</h3>
    <p>段落</p>

    <h3>原因 2</h3>
    <p>段落</p>

  <h2>总结</h2>
  <p>段落</p>
```

Chapter 3: Headings

HTML provides not only plain paragraph tags, but six separate header tags to indicate headings of various sizes and thicknesses. Enumerated as heading 1 through heading 6, heading 1 has the largest and thickest text while heading 6 is the smallest and thinnest, down to the paragraph level. This topic details proper usage of these tags.

Section 3.1: Using Headings

Headings can be used to describe the topic they precede and they are defined with the <h1> to <h6> tags. Headings support all the global attributes.

- <h1> defines the most important heading.
- <h6> defines the least important heading.

Defining a heading:

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

Correct structure matters

Search engines and other user agents usually index page content based on heading elements, for example to create a table of contents, so using the correct structure for headings is important.

In general, an article should have one h1 element for the main title followed by h2 subtitles – going down a layer if necessary. If there are h1 elements on a higher level they shoudn't be used to describe any lower level content.

Example document (extra intendation to illustrate hierarchy):

```
<h1>Main title</h1>
<p>Introduction</p>

  <h2>Reasons</h2>

    <h3>Reason 1</h3>
    <p>Paragraph</p>

    <h3>Reason 2</h3>
    <p>Paragraph</p>

  <h2>In conclusion</h2>
  <p>Paragraph</p>
```

第4章：段落

列	列
<p>	定义段落
 	插入单行换行符
<pre>	定义预格式化文本

段落是最基本的HTML元素。本主题解释并演示了HTML中段落元素的使用法。

第4.1节：HTML段落

HTML<p>元素定义了一个段落：

```
<p>这是一个段落。</p>
<p>这是另一个段落。</p>
```

显示-

你无法确定HTML将如何显示。

大屏幕或小屏幕，以及调整大小的窗口会产生不同的结果。

在HTML中，不能通过在HTML代码中添加额外的空格或额外的换行来改变输出效果。

浏览器在显示页面时会删除所有多余的空格和多余的换行：

```
<p>这是      另一个      段落，浏览器会删除多余的空格</p>
```

Chapter 4: Paragraphs

Column	Column
<p>	Defines a paragraph
 	Inserts a single line break
<pre>	Defines pre-formatted text

Paragraphs are the most basic HTML element. This topic explains and demonstrates the usage of the paragraph element in HTML.

Section 4.1: HTML Paragraphs

The HTML <p> element defines a **paragraph**:

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Display-

You cannot be sure how HTML will be displayed.

Large or small screens, and resized windows will create different results.

With HTML, you cannot change the output by adding extra spaces or extra lines in your HTML code.

The browser will remove any extra spaces and extra lines when the page is displayed:

```
<p>This is      another      paragraph, extra spaces      will be      removed      by browsers</p>
```

第5章：文本格式

虽然大多数HTML标签用于创建元素，但HTML也提供了文本内格式化标签，用于对部分文本应用特定的文本相关样式。本主题包括HTML文本格式化的示例，如高亮、加粗、下划线、下标和删除线文本。

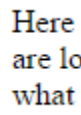
第5.1节：高亮

`<mark>`元素是HTML5中新引入的，用于标记或高亮文档中的文本，“因为其在另一个上下文中的相关性”。1

最常见的例子是在搜索结果中，用户输入了搜索查询，结果显示时高亮显示所需的查询内容。

```
<p>这里是一篇文章中的部分内容，包含我们正在查找的<mark>搜索查询</mark>。高亮显示文本将使用户更容易找到他们想要的内容。</p>
```

输出：



一种常见的标准格式是黑色文字配黄色背景，但这可以通过CSS进行更改。

第5.2节：加粗、斜体和下划线

加粗文本

要加粗文本，使用``或``标签：

```
<strong>加粗文本</strong>
```

或者

```
<b>加粗文本</b>
```

有什么区别？语义。`` 用于表示文本在语义上对周围文本具有根本性或重要性，而`` 则表示这种重要性，仅表示文本应加粗。

如果使用``，文本转语音程序不会以不同方式朗读该词——你只是强调它们，而没有增加额外的重要性。通过使用

``，程序会用不同的语调朗读这些词，以传达文本在某种程度上的重要性。

斜体文本

要将文本斜体化，使用``或`<i>`标签：

Chapter 5: Text Formatting

While most HTML tags are used to create elements, HTML also provides in-text formatting tags to apply specific text-related styles to portions of text. This topic includes examples of HTML text formatting such as highlighting, bolding, underlining, subscript, and stricken text.

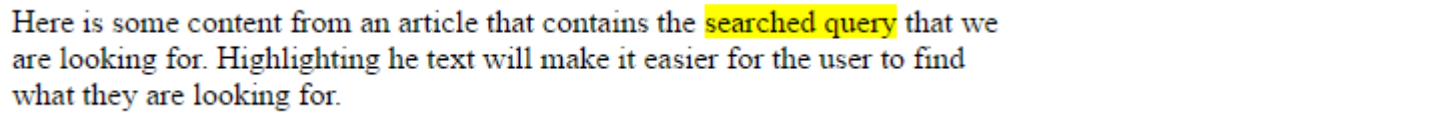
Section 5.1: Highlighting

The `<mark>` element is new in HTML5 and is used to mark or highlight text in a document "due to its relevance in another context".1

The most common example would be in the results of a search where the user has entered a search query and results are shown highlighting the desired query.

```
<p>Here is some content from an article that contains the <mark>searched query</mark> that we are looking for. Highlighting the text will make it easier for the user to find what they are looking for.</p>
```

Output:



A common standard formatting is black text on a yellow background, but this can be changed with CSS.

Section 5.2: Bold, Italic, and Underline

Bold Text

To bold text, use the `` or `` tags:

```
<strong>Bold Text Here</strong>
```

or

```
<b>Bold Text Here</b>
```

What's the difference? Semantics. `` is used to indicate that the text is fundamentally or semantically *important* to the surrounding text, while `` indicates no such importance and simply represents text that should be bolded.

If you were to use `` a text-to-speech program would not say the word(s) any differently than any of the other words around it - you are simply drawing attention to them without adding any additional importance. By using ``, though, the same program would want to speak those word(s) with a different tone of voice to convey that the text is important in some way.

Italic Text

To italicize text, use the `` or `<i>` tags:

```
<em>斜体文本</em>
```

或者

```
<i>斜体文本</i>
```

有什么区别？语义。 用于表示文本应有额外强调，需要突出显示，而<i> 仅表示应与周围正常文本区分开的文本。

例如，如果你想强调句子中的动作，可以通过斜体来强调它，方式是: "你能不能快点提交编辑？"

但如果你是在标识一本通常会用斜体表示的书名或报纸名，你只需使用<i>: "我在高中时被迫读了《罗密欧与朱丽叶》。"

下划线文本

虽然<u>元素本身在HTML 4中被弃用，但在HTML 5中以另一种语义含义重新引入——表示未明确说明的、非文本的注释。你可能会用这种方式来标示页面上的拼写错误，或者用于中文专有名词标记。

```
<p>这段文字包含一些<u>拼写错误的</u>文本。</p>
```

第5.3节：缩写

要标记某个表达为缩写，使用<abbr>标签：

```
<p>我喜欢写<abbr title="超文本标记语言">HTML</abbr>！</p>
```

如果存在，title 属性用于显示该缩写的完整描述。

第5.4节：插入、删除或删除线

要标记为插入的文本，请使用 <ins> 标签：

```
<ins>新文本</ins>
```

要标记为删除的文本，请使用 标签：

```
<del>已删除文本</del>
```

要添加删除线的文本，请使用 <s> 标签：

```
<s>此处为删除线文本</s>
```

第5.5节：上标和下标

要将文本向上或向下偏移，可以使用 <sup> 和 <sub> 标签。

创建上标：

```
<sup>上标内容</sup>
```

```
<em>Italicized Text Here</em>
```

or

```
<i>Italicized Text Here</i>
```

What's the difference? Semantics. is used to indicate that the text should have extra emphasis that should be stressed, while <i> simply represents text which should be set off from the normal text around it.

For example, if you wanted to stress the action inside a sentence, one might do so by emphasizing it in italics via : "Would you just *submit* the edit already?"

But if you were identifying a book or newspaper that you would normally italicize stylistically, you would simply use <i>: "I was forced to read *Romeo and Juliet* in high school.

Underlined Text

While the <u> element itself was deprecated in HTML 4, it was reintroduced with alternate semantic meaning in HTML 5 - to represent an unarticulated, non-textual annotation. You might use such a rendering to indicate misspelled text on the page, or for a Chinese proper name mark.

```
<p>This paragraph contains some <u>misspelled</u> text.</p>
```

Section 5.3: Abbreviation

To mark some expression as an abbreviation, use <abbr> tag:

```
<p>I like to write <abbr title="Hypertext Markup Language">HTML</abbr>!</p>
```

If present, the title attribute is used to present the full description of such abbreviation.

Section 5.4: Inserted, Deleted, or Stricken

To mark text as inserted, use the <ins> tag:

```
<ins>New Text</ins>
```

To mark text as deleted, use the tag:

```
<del>Deleted Text</del>
```

To strike through text, use the <s> tag:

```
<s>Struck-through text here</s>
```

Section 5.5: Superscript and Subscript

To offset text either upward or downward you can use the tags <sup> and <sub>.

To create superscript:

```
<sup>superscript here</sup>
```


创建下标：

_{下标内容}

To create subscript:

_{subscript here}

第6章：锚点和超链接

参数	详情
href	指定目标地址。它可以是绝对或相对URL，或者锚点的名称。绝对URL是网站的完整URL，如http://example.com/。相对URL指向同一网站内的另一个目录和/或文档，例如/about-us/指向目录“about-us”位于根目录（/）内。当指向另一个目录但未明确指定文档时，Web服务器通常返回该目录内的文档“index.html”。
hreflang	指定由 href 属性链接的资源的语言（该属性必须与此属性一起存在）。HTML5 使用 BCP 47中的语言值，HTML 4 使用 RFC 1766中的语言值。
rel	指定当前文档与链接文档之间的关系。对于 HTML5，值必须在规范中定义或在 Microformats 维基中注册。
target	指定链接打开的位置，例如在新标签页或窗口中。可能的值有_blank、_self、_parent、_top和frameName（已弃用）。不建议强制此类行为，因为这会侵犯用户对网站的控制权。
title	指定链接的额外信息。当光标移动到链接上时，这些信息通常以工具提示文本的形式显示。该属性不限于链接，几乎所有 HTML 标签都可以使用。
download	指定当用户点击超链接时，目标将被下载。该属性的值将作为下载文件的名称。允许的值没有限制，浏览器会自动检测正确的文件扩展名并添加到文件名后（如 .img、.pdf 等）。如果省略该值，则使用原始文件名。

锚点标签通常用于链接不同的网页，但它们也可以用于链接单个文档中的不同位置，通常是在目录中，甚至可以启动外部应用程序。本文介绍了HTML锚点标签在各种角色中的实现和应用。

第6.1节：链接到另一个网站

这是<a>（anchor元素）元素的基本用法：

```
<a href="http://example.com/">链接到example.com</a>
```

它创建了一个超链接，链接到由href（超文本引用）属性指定的URL http://example.com/，锚文本为“链接到example.com”。它看起来大致如下：



为了表示该链接指向外部网站，你可以使用external链接类型：

```
<a href="http://example.com/" rel="external">example站点</a>
```

你可以链接到使用HTTP以外协议的网站。例如，要链接到FTP站点，可以这样做，

```
<a href="ftp://example.com/">这可能是一个指向FTP站点的链接</a>
```

在这种情况下，区别在于该锚标签请求用户的浏览器使用文件传输协议（FTP）而非超文本传输协议（HTTP）连接到example.com。

这可能是一个指向FTP站点的链接

Chapter 6: Anchors and Hyperlinks

Parameter	Details
href	Specifies the destination address. It can be an absolute or relative URL, or the name of an anchor. An absolute URL is the complete URL of a website like http://example.com/ . A relative URL points to another directory and/or document inside the same website, e.g. /about-us/ points to the directory “about-us” inside the root directory (/). When pointing to another directory without explicitly specifying the document, web servers typically return the document “index.html” inside that directory.
hreflang	Specifies the language of the resource linked by the href attribute (which must be present with this one). Use language values from BCP 47 for HTML5 and RFC 1766 for HTML 4.
rel	Specifies the relationship between the current document and the linked document. For HTML5, the values must be defined in the specification or registered in the Microformats wiki .
target	Specifies where to open the link, e.g. in a new tab or window. Possible values are _blank, _self, _parent, _top, and frameName (deprecated). Forcing such behaviour is not recommended since it violates the control of the user over a website.
title	Specifies extra information about a link. The information is most often shown as a tooltip text when the cursor moves over the link. This attribute is not restricted to links, it can be used on almost all HTML tags.
download	Specifies that the target will be downloaded when a user clicks on the hyperlink. The value of the attribute will be the name of the downloaded file. There are no restrictions on allowed values, and the browser will automatically detect the correct file extension and add it to the file (.img, .pdf, etc.). If the value is omitted, the original filename is used.

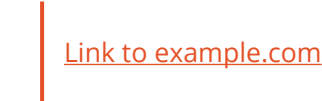
Anchor tags are commonly used to link separate webpages, but they can also be used to link between different places in a single document, often within table of contents or even launch external applications. This topic explains the implementation and application of HTML anchor tags in various roles.

Section 6.1: Link to another site

This is the basic use of the <a> ([anchor element](#)) element:

```
<a href="http://example.com/">Link to example.com</a>
```

It creates a hyperlink, to the URL http://example.com/ as specified by the href (hypertext reference) attribute, with the anchor text "Link to example.com". It would look something like the following:



To denote that this link leads to an external website, you can use the external link type:

```
<a href="http://example.com/" rel="external">example site</a>
```

You can link to a site that uses a protocol other than HTTP. For example, to link to an FTP site, you can do,

```
<a href="ftp://example.com/">This could be a link to a FTP site</a>
```

In this case, the difference is that this anchor tag is requesting that the user's browser connect to example.com using the File Transfer Protocol (FTP) rather than the Hypertext Transfer Protocol (HTTP).

This could be a link to a FTP site

第6.2节：链接到锚点

锚点可用于跳转到HTML页面上的特定标签。<a>标签可以指向任何具有id属性的元素。要了解更多关于ID的信息，请访问[关于类和ID的文档](#)。锚点主要用于跳转到页面的子部分，通常与标题标签一起使用。

假设你创建了一个包含多个主题 的页面（page1.html）：

```
<h2>第一个主题</h2>
<p>关于第一个主题的内容</p>
<h2>第二个主题</h2>
<p>关于第二个主题的内容</p>
```

当你有多个部分时，可能想在页面顶部创建一个目录，包含指向特定部分的快速链接（或书签）。

如果你给主题添加了id属性，就可以链接到它们

```
<h2 id="Topic1">第一个主题</h2>
<p>关于第一个主题的内容</p>
<h2 id="Topic2">第二个主题</h2>
<p>关于第二个主题的内容</p>
```

现在你可以在目录中使用锚点：

```
<h1>目录</h1>
  <a href='#Topic1'>点击跳转到第一个主题</a>
  <a href='#Topic2'>点击跳转到第二个主题</a>
```

这些锚点也附加在它们所在的网页（page1.html）上。因此，你可以通过引用页面和锚点名称，在网站内从一个页面链接到另一个页面。

请记住，你总是可以回到第一个主题查找支持信息。

第6.3节：链接到同一网站上的页面

你可以使用相对路径链接到同一网站上的页面。

```
<a href="/example">这里是文本</a>
```

上述示例将跳转到服务器根目录（/）下的文件example。

如果此链接位于<http://example.com>，以下两个链接都会将用户带到相同的位置

```
<a href="/page">这里是文本</a>
<a href="http://example.com/page">这里是文本</a>
```

以上两个链接都会跳转到example.com根目录下的page文件。

第6.4节：拨打号码的链接

如果 href 属性的值以 tel: 开头，当你点击它时，设备会拨打该号码。这适用于移动设备或运行类似Skype或FaceTime等可以拨打电话的软件的电脑/平板电脑。

Section 6.2: Link to an anchor

Anchors can be used to jump to specific tags on an HTML page. The <a> tag can point to any element that has an id attribute. To learn more about IDs, visit the [documentation about Classes and IDs](#). Anchors are mostly used to jump to a subsection of a page and are used in conjunction with header tags.

Suppose you've created a page (page1.html) on many topics:

```
<h2>First topic</h2>
<p>Content about the first topic</p>
<h2>Second topic</h2>
<p>Content about the second topic</p>
```

Once you have several sections, you may want to create a Table of Contents at the top of the page with quick-links (or bookmarks) to specific sections.

If you gave an id attribute to your topics, you could then link to them

```
<h2 id="Topic1">First topic</h2>
<p>Content about the first topic</p>
<h2 id="Topic2">Second topic</h2>
<p>Content about the second topic</p>
```

Now you can use the anchor in your table of contents:

```
<h1>Table of Contents</h1>
  <a href='#Topic1'>Click to jump to the First Topic</a>
  <a href='#Topic2'>Click to jump to the Second Topic</a>
```

These anchors are also attached to the web page they're on (page1.html). So you can link across the site from one page to the other by referencing the page *and* anchor name.

Remember, you can always look back in the First Topic for supporting information.

Section 6.3: Link to a page on the same site

You can use a [relative path](#) to link to pages on the same website.

```
<a href="/example">Text Here</a>
```

The above example would go to the file example at the root directory (/) of the server.

If this link was on <http://example.com>, the following two links would bring the user to the same location

```
<a href="/page">Text Here</a>
<a href="http://example.com/page">Text Here</a>
```

Both of the above would go to the page file at the root directory of example.com.

Section 6.4: Link that dials a number

If the value of the href-attribute begins with tel:, your device will dial the number when you click it. This works on mobile devices or on computers/tablets running software – like Skype or FaceTime – that can make phone calls.

给我们打电话

大多数设备和程序都会以某种方式提示用户确认他们即将拨打的号码。

第6.5节：在新标签页/窗口中打开链接

这里是文本

target 属性指定链接的打开位置。将其设置为 _blank，表示告诉浏览器在新标签页或窗口中打开（根据用户偏好）。

安全漏洞警告！

使用 target="_blank" 会通过JavaScript使打开页面部分访问 window.opener 对象，这允许该页面访问并更改 window.opener.location ，从而可能将用户重定向到恶意软件或钓鱼网站。

每当对你不控制的页面使用此功能时，请在链接中添加 rel="noopener"，以防止 window.opener 对象随请求发送。

目前，Firefox 不支持 noopener，因此您需要使用 rel="noopener noreferrer" 以获得最大效果。

第6.6节：运行JavaScript的链接

只需使用javascript:协议，将文本作为JavaScript运行，而不是作为普通链接打开：

运行代码

你也可以使用onclick属性实现同样的效果：

运行代码

需要return false;来防止点击指向#的链接时页面滚动到顶部。确保在它之前包含所有你想运行的代码，因为return会停止后续代码的执行。

另外值得注意的是，你可以在井号后面加一个感叹号!来防止页面滚动到顶部。这是因为任何无效的slug都会导致链接不会在页面上滚动到任何地方，因为它找不到它引用的元素（id为"!"的元素）。你也可以使用任何无效的slug（例如#scrollsNowhere）来达到同样的效果。在这种情况下，不需要return false;：

运行代码

你应该使用这些方法吗？

答案几乎肯定是不。像这样在元素内联运行JavaScript是相当不好的做法。考虑使用纯JavaScript解决方案，在页面中查找元素并绑定函数。监听事件

还要考虑该元素是否真的是一个按钮而不是一个链接。如果是的话，应该使用<button>。

Call us

Most devices and programs will prompt the user in some way to confirm the number they are about to dial.

Section 6.5: Open link in new tab/window

Text Here

The target attribute specifies where to open the link. By setting it to _blank, you tell the browser to open it in a new tab or window (per user preference).

SECURITY VULNERABILITY WARNING!

Using target="_blank" gives the opening site partial access to the window.opener object via JavaScript, which allows that page to then access and change the window.opener.location of *your* page and potentially redirect users to malware or phishing sites.

Whenever using this for pages you do not control, add rel="noopener" to your link to prevent the window.opener object from being sent with the request.

Currently, Firefox does not support noopener, so you will need to use rel="noopener noreferrer" for maximum effect.

Section 6.6: Link that runs JavaScript

Simply use the javascript: protocol to run the text as JavaScript instead of opening it as a normal link:

Run Code

You can also achieve the same thing using the onclick attribute:

Run Code

The return false; is necessary to prevent your page from scrolling to the top when the link to # is clicked. Make sure to include all code you'd like to run before it, as returning will stop execution of further code.

Also noteworthy, you can include an exclamation mark ! after the hashtag in order to prevent the page from scrolling to the top. This works because any invalid slug will cause the link to not scroll *anywhere* on the page, because it couldn't locate the element it references (an element with id="!"). You could also just use any invalid slug (such as #scrollsNowhere) to achieve the same effect. In this case, return false; is not required:

Run Code

Should you be using any of this?

The answer is almost certainly **no**. Running JavaScript inline with the element like this is fairly bad practice. Consider using pure JavaScript solutions that look for the element in the page and bind a function to it instead. Listening for an event

Also consider whether this element is really a *button* instead of a *link*. If so, you should use <button>.

第6.7节：运行电子邮件客户端的链接

基本用法

如果href属性的值以mailto:开头，点击时会尝试打开电子邮件客户端：

```
<a href="mailto:example@example.com">发送电子邮件</a>
```

这会将电子邮件地址example@example.com作为新建邮件的收件人。

抄送和密送

你也可以使用以下语法添加抄送（cc）或密送（bcc）收件人地址：

```
<a href="mailto:example@example.com?cc=john@example.com&bcc=jane@example.com">发送电子邮件</a>
```

主题和正文文本

您还可以填写新邮件的主题和正文：

```
<a href="mailto:example@example.com?subject=Example+subject&body=Message+text">发送邮件</a>
```

这些值必须进行URL编码。

点击带有mailto:的链接将尝试打开操作系统指定的默认邮件客户端，或者会询问您选择要使用的客户端。并非所有邮件客户端都支持收件人地址后指定的所有选项。

Section 6.7: Link that runs email client

Basic usage

If the value of the href-attribute begins with `mailto:` it will try to open an email client on click:

```
<a href="mailto:example@example.com">Send email</a>
```

This will put the email address `example@example.com` as the recipient for the newly created email.

Cc and Bcc

You can also add addresses for cc- or bcc-recipients using the following syntax:

```
<a href="mailto:example@example.com?cc=john@example.com&bcc=jane@example.com">Send email</a>
```

Subject and body text

You can populate the subject and body for the new email as well:

```
<a href="mailto:example@example.com?subject=Example+subject&body=Message+text">Send email</a>
```

Those values must be [URL encoded](#).

Clicking on a link with `mailto:` will try to open the default email client specified by your operating system or it will ask you to choose what client you want to use. Not all options specified after the recipient's address are supported in all email clients.

第7章：列表

HTML提供了三种指定列表的方式：有序列表、无序列表和描述列表。有序列表使用序数序列来表示列表元素的顺序，无序列表使用定义的符号（如项目符号）来列出无特定顺序的元素，描述列表使用缩进来列出元素及其子项。本节介绍这些列表在HTML标记中的实现和组合方式。

第7.1节：有序列表

可以使用标签创建有序列表，每个列表项可以使用标签创建，如下面的示例所示：

```
<ol>
  <li>项目</li>
  <li>另一个项目</li>
  <li>再一个项目</li>
</ol>
```

这将生成一个编号列表（这是默认样式）：

- 1.项目
- 2.另一个项目
- 3.再一个项目

手动更改数字

有几种方法可以调整有序列表中项目的编号。第一种方法是使用start属性设置起始数字。列表将从该定义的数字开始，并按通常方式递增1。

```
<ol start="3">
  <li>项目</li>
  <li>另一个项目</li>
  <li>再一个项目</li>
</ol>
```

这将生成一个编号列表（这是默认样式）：

- 3.项目
- 4.另一个项目
- 5.再一个项目

你也可以显式地将某个列表项设置为特定数字。指定值之后的列表项将从该列表项的值开始递增1，忽略父列表的当前位置。

```
<li value="7"></li>
```

同样值得注意的是，通过直接在列表项上使用value属性，可以通过从较低的值重新开始编号来覆盖有序列表的现有编号系统。因此，如果父列表已经编号到值7，而遇到一个值为4的列表项，那么该列表项仍将显示为4并从该点开始继续计数

Chapter 7: Lists

HTML offers three ways for specifying lists: ordered lists, unordered lists, and description lists. Ordered lists use ordinal sequences to indicate the order of list elements, unordered lists use a defined symbol such as a bullet to list elements in no designated order, and description lists use indents to list elements with their children. This topic explains the implementation and combination of these lists in HTML markup.

Section 7.1: Ordered List

An ordered list can be created with the tag and each list item can be created with the tag as in the example below:

```
<ol>
  <li>Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ol>
```

This will produce a numbered list (which is the default style):

1. Item
2. Another Item
3. Yet Another Item

Manually changing the numbers

There are a couple of ways you can play with which numbers appear on the list items in an ordered list. The first way is to set a starting number, using the start attribute. The list will start at this defined number, and continue incrementing by one as usual.

```
<ol start="3">
  <li>Item</li>
  <li>Some Other Item</li>
  <li>Yet Another Item</li>
</ol>
```

This will produce a numbered list (which is the default style):

3. Item
4. Some Other Item
5. Yet Another Item

You can also explicitly set a certain list item to a specific number. Further list items after one with a specified value will continue incrementing by one from that list item's value, ignoring where the parent list was at.

```
<li value="7"></li>
```

It is also worth noting that, by using the value attribute directly on a list item, you can override an ordered list's existing numbering system by restarting the numbering at a lower value. So if the parent list was already up to value 7, and encountered a list item at value 4, then that list item would still display as 4 and continue counting

从那一点重新开始。

```
<ol start="5">
  <li>项目</li>
  <li>其他项目</li>
  <li value="4">重置项目</li>
  <li>另一个项目</li>
  <li>又一个项目</li>
</ol>
```

因此，上述示例将生成一个编号模式为5、6、4、5、6的列表——从一个比之前更低的数字重新开始，并在列表中重复数字6。

注意：start和value属性只接受数字——即使有序列表设置为显示罗马数字或字母。

版本 ≥ 5

你可以通过在ol元素中添加reversed来反转编号：

```
<ol reversed>
  <li>项目</li>
  <li>其他项目</li>
  <li value="4">重置项目</li>
  <li>另一个项目</li>
  <li>又一个项目</li>
</ol>
```

反向编号在你不断向列表中添加内容时非常有用，比如新的播客剧集或演示文稿，并且你希望最新的项目出现在最前面。

更改数字类型

您可以通过使用 type 属性轻松更改列表项标记中显示的数字类型

<ol type="1 a A i I">		
类型	描述	示例
1	默认值 - 十进制数字	1,2,3,4
a	按字母顺序排列（小写） a,b,c,d	
A	按字母顺序排列（大写） A,B,C,D	
i	罗马数字（小写）	i,ii,iii,iv
I	罗马数字（大写）	I,II,III,IV

你应该使用ol来显示一个项目列表，其中项目是有意排序的，且应强调顺序。如果改变项目顺序不会使列表错误，则应使用。

第7.2节：无序列表

可以使用标签创建无序列表，每个列表项可以使用标签创建，如下面的示例所示：

from that point again.

```
<ol start="5">
  <li>Item</li>
  <li>Some Other Item</li>
  <li value="4">A Reset Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ol>
```

So the example above will produce a list that follows the numbering pattern of 5, 6, 4, 5, 6 - starting again at a number lower than the previous and duplicating the number 6 in the list.

Note: The start and value attributes only accept a number - even if the ordered list is set to display as Roman numerals or letters.

Version ≥ 5

You can reverse the numbering by adding reversed in your ol element:

```
<ol reversed>
  <li>Item</li>
  <li>Some Other Item</li>
  <li value="4">A Reset Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ol>
```

Reverse numbering is helpful if you're continually adding to a list, such as with new podcast episodes or presentations, and you want the most recent items to appear first.

Changing the type of numeral

You can easily change the type of numeral shown in the list item marker by using the type attribute

<ol type="1 a A i I">		
Type	Description	Examples
1	Default value - Decimal numbers	1,2,3,4
a	Alphabetically ordered (lowercase)	a,b,c,d
A	Alphabetically ordered (uppercase)	A,B,C,D
i	Roman Numerals (lowercase)	i,ii,iii,iv
I	Roman Numerals (uppercase)	I,II,III,IV

You should use ol to display a list of items, where the items have been intentionally ordered and order should be emphasized. If changing the order of the items does NOT make the list incorrect, you should use .

Section 7.2: Unordered List

An unordered list can be created with the tag and each list item can be created with the tag as shown by the example below:

```
<ul>
  <li>项目</li>
  <li>另一个项目</li>
  <li>又一个项目</li>
</ul>
```

这将生成一个带项目符号的列表（这是默认样式）：

- 项目
- 另一个项目
- 又一个项目

你应该使用ul来显示项目列表，其中项目的顺序不重要。如果更改项目的顺序会导致列表错误，则应使用。

第7.3节：嵌套列表

你可以嵌套列表来表示列表项的子项。

```
<ul>
  <li>项目1</li>
  <li>项目2
    <ul>
      <li>子项目2.1</li>
      <li>子项目2.2</li>
    </ul>
  </li>
  <li>项目3</li>
</ul>
```

- 项目1
- 项目2
 - 子项 2.1
 - 子项 2.2
- 项目 3

嵌套列表必须是li元素的子元素。

你也可以嵌套不同类型的列表：

```
<ol>
  <li>你好，列表！</li>
  <li>
    <ul>
      <li>你好，嵌套列表！</li>
    </ul>
  </li>
</ol>
```

第7.4节：描述列表

描述列表（或称为定义列表，HTML5之前的叫法）可以用dl元素创建。它由

```
<ul>
  <li>Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ul>
```

This will produce a bulleted list (which is the default style):

- Item
- Another Item
- Yet Another Item

You should use ul to display a list of items, where the order of the items is not important. If changing the order of the items makes the list incorrect, you should use .

Section 7.3: Nested lists

You can nest lists to represent sub-items of a list item.

```
<ul>
  <li>item 1</li>
  <li>item 2
    <ul>
      <li>sub-item 2.1</li>
      <li>sub-item 2.2</li>
    </ul>
  </li>
  <li>item 3</li>
</ul>
```

- item 1
- item 2
 - sub-item 2.1
 - sub-item 2.2
- item 3

The nested list has to be a child of the li element.

You can nest different types of list, too:

```
<ol>
  <li>Hello, list!</li>
  <li>
    <ul>
      <li>Hello, nested list!</li>
    </ul>
  </li>
</ol>
```

Section 7.4: Description List

A description list (or *definition list*, as it was called before HTML5) can be created with the dl element. It consists of

名称-值组组成，其中名称放在dt元素中，值放在dd元素中。

```
<dl>
  <dt>名称 1</dt>
  <dd>值 1</dd>
  <dt>名称 2</dt>
  <dd>值 2</dd>
</dl>
```

[实时演示](#)

一个名称-值组可以有多个名称和/或多个值（表示备选项）：

```
<dl>

  <dt>名称 1</dt>
  <dt>名称 2</dt>
  <dd>名称 1 和 2 的值</dd>

  <dt>名称 3</dt>
  <dd>名称 3 的值</dd>
  <dd>名称 3 的值</dd>

</dl>
```

[实时演示](#)

name-value groups, where the name is given in the dt element, and the value is given in the dd element.

```
<dl>
  <dt>name 1</dt>
  <dd>value for 1</dd>
  <dt>name 2</dt>
  <dd>value for 2</dd>
</dl>
```

[Live demo](#)

A name-value group can have more than one name and/or more than one value (which represent alternatives):

```
<dl>

  <dt>name 1</dt>
  <dt>name 2</dt>
  <dd>value for 1 and 2</dd>

  <dt>name 3</dt>
  <dd>value for 3</dd>
  <dd>value for 3</dd>

</dl>
```

[Live demo](#)

第8章：表格

HTML `<table>` 元素允许网页作者以二维表格的形式显示表格数据（例如文本、图像、链接、其他表格等），表格由行和列的单元格组成。

第8.1节：简单表格

```
<table>
  <tr>
    <th>标题 1/列 1</th>
    <th>标题 2/列 2</th>
  </tr>
  <tr>
    <td>第 1 行数据 列 1</td>
    <td>第 1 行数据 列 2</td>
  </tr>
  <tr>
    <td>第 2 行数据 列 1</td>
    <td>第 2 行数据 列 2</td>
  </tr>
</table>
```

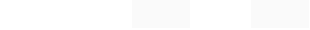
这将渲染一个`<table>`，包含总共三行（`<tr>`）：一行表头单元格（`<th>`）和两行内容单元格（`<td>`）。`<th>`元素是表格头部，`<td>`元素是表格数据。你可以在`<td>`或`<th>`中放入任何内容。



标题 1/列 1 标题 2/列 2
第 1 行数据 列 1 第 1 行数据 列 2
第 2 行数据 列 1 第 2 行数据 列 2

第 8.2 节：跨列或跨行

表格单元格可以使用`colspan`和`rowspan`属性跨越多列或多行。这些属性可以应用于`<th>`和`<td>`元素。



```
<table>
  <tr>
    <td>第1行第1列</td>
    <td>第1行第2列</td>
    <td>第1行第3列</td>
  </tr>
  <tr>
    <td colspan="3">第二行跨越所有三列</td>
  </tr>
  <tr>
    <td rowspan="2">此单元格跨越两行</td>
    <td>第3行第2列</td>
    <td>第3行第3列</td>
  </tr>
  <tr>
    <td>第4行第2列</td>
    <td>第4行第3列</td>
  </tr>
</table>
```

Chapter 8: Tables

The HTML `<table>` element allows web authors to display tabular data (such as text, images, links, other tables, etc.) in a two dimensional table with rows and columns of cells.

Section 8.1: Simple Table

```
<table>
  <tr>
    <th>Heading 1/Column 1</th>
    <th>Heading 2/Column 2</th>
  </tr>
  <tr>
    <td>Row 1 Data Column 1</td>
    <td>Row 1 Data Column 2</td>
  </tr>
  <tr>
    <td>Row 2 Data Column 1</td>
    <td>Row 2 Data Column 2</td>
  </tr>
</table>
```

This will render a `<table>` consisting of three total rows (`<tr>`): one row of header cells (`<th>`) and two rows of content cells (`<td>`). `<th>` elements are *tabular headers* and `<td>` elements are *tabular data*. You can put whatever you want inside a `<td>` or `<th>`.

Heading 1/Column 1 Heading 2/Column 2
Row 1 Data Column 1 Row 1 Data Column 2
Row 2 Data Column 1 Row 2 Data Column 2

Section 8.2: Spanning columns or rows

Table cells can span multiple columns or rows using the `colspan` and `rowspan` attributes. These attributes can be applied to `<th>` and `<td>` elements.

```
<table>
  <tr>
    <td>row 1 col 1</td>
    <td>row 1 col 2</td>
    <td>row 1 col 3</td>
  </tr>
  <tr>
    <td colspan="3">This second row spans all three columns</td>
  </tr>
  <tr>
    <td rowspan="2">This cell spans two rows</td>
    <td>row 3 col 2</td>
    <td>row 3 col 3</td>
  </tr>
  <tr>
    <td>row 4 col 2</td>
    <td>row 4 col 3</td>
  </tr>
</table>
```

将导致

row 1 col 1	row 1 col 2	row 1 col 3
This second row spans all three columns		
This cell spans two rows	row 3 col 2	row 3 col 3
	row 4 col 2	row 4 col 3

请注意，不应设计行和列都重叠的表格，因为这是无效的HTML，不同的网页浏览器对此的处理方式不同。

rowspan = 一个非负整数，指定单元格跨越的行数。该属性的默认值为一（1）。值为零（0）表示单元格将从当前行延伸到表格的最后一行（<thead>、<tbody>或<tfoot>）。

colspan = 一个非负整数，指定当前单元格跨越的列数。该属性的默认值为一（1）。值为零（0）表示单元格将从当前列延伸到定义该单元格的列组<colgroup>的最后一列。

第8.3节：列组

有时你可能想对某一列或一组列应用样式。或者出于语义目的，你可能想将列分组。为此，请使用<colgroup>和<col>元素。

可选的<colgroup>标签允许你将列分组。 <colgroup>元素必须是<table>的子元素，且必须位于任何<caption>元素之后，任何表格内容（例如<tr>、<thead>、<tbody>等）之前。

```
<table>
  <colgroup span="2"></colgroup>
  <colgroup span="2"></colgroup>
  ...
</table>
```

可选的<col>标签允许你引用单个列或一系列列，而不应用逻辑分组。 <col>元素是可选的，但如果存在，必须位于<colgroup>元素内。

```
<table>
  <colgroup>
    <col id="MySpecialColumn" />
    <col />
  </colgroup>
  <colgroup>
    <col class="CoolColumn" />
    <col class="NeatColumn" span="2" />
  </colgroup>
  ...
</table>
```

以下CSS样式可以应用于<colgroup>和<col>元素：

- 边框
- 背景

Will result in

row 1 col 1	row 1 col 2	row 1 col 3
This second row spans all three columns		
This cell spans two rows	row 3 col 2	row 3 col 3
	row 4 col 2	row 4 col 3

Note that you should not design a table where both rows and columns overlap as this is invalid HTML and the result is handled differently by different web browsers.

rowspan = A non-negative integer that specifies the number of rows spanned by a cell. The default value of this attribute is one (1). A value of zero (0) means that the cell will extend from the current row until the last row of the table (<thead>, <tbody>, or <tfoot>).

colspan = A non-negative integer that specifies the number of columns spanned by the current cell. The default value of this attribute is one (1). A value of zero (0) means that the cell will extend from the current to the last column of the column group <colgroup> in which the cell is defined.

Section 8.3: Column Groups

Sometimes you may want to apply styling to a column or group of columns. Or for semantic purposes, you may want to group columns together. To do this, use <colgroup> and <col> elements.

The optional <colgroup> tag allows you to group columns together. <colgroup> elements must be child elements of a <table> and must come after any <caption> elements and before any table content (e.g., <tr>, <thead>, <tbody>, etc.).

```
<table>
  <colgroup span="2"></colgroup>
  <colgroup span="2"></colgroup>
  ...
</table>
```

The optional <col> tag allows you to reference individual columns or a range of columns without applying a logical grouping. <col> elements are optional, but if present, they must be inside a <colgroup> element.

```
<table>
  <colgroup>
    <col id="MySpecialColumn" />
    <col />
  </colgroup>
  <colgroup>
    <col class="CoolColumn" />
    <col class="NeatColumn" span="2" />
  </colgroup>
  ...
</table>
```

The following CSS styles can be applied to <colgroup> and <col> elements:

- border
- background

- 宽度
- 可见性
- 显示（如display: none）
 - display: none;实际上会将列从显示中移除，使表格渲染时仿佛这些单元格不存在

更多信息，请参见HTML5表格数据。

第8.4节：带有thead、tbody、tfoot和caption的表格

HTML还为表格提供了<thead>、<tbody>、<tfoot>和<caption>元素。这些附加元素有助于为表格添加语义价值，并为单独的CSS样式提供位置。

当打印一张无法完整显示在一页（纸张）上的表格时，大多数浏览器会在每一页上重复<thead>的内容。

必须遵守特定的顺序，我们应当意识到并非所有元素都会如预期那样排列。以下示例展示了我们的4个元素应如何放置。

```
<table>
  <caption>表格标题</caption> <!--| caption 是表格的第一个子元素 |-->
  <thead> <!--=====| thead 位于 caption 之后 |-->
    <tr>
      <th>表头内容 1</th>
      <th>表头内容 2</th>
    </tr>
  </thead>

  <tbody> <!--=====| tbody 位于 thead 之后 |-->
    <tr>
      <td>主体内容 1</td>
      <td>主体内容 2</td>
    </tr>
  </tbody>

  <tfoot><!--| tfoot 可以放在 tbody 之前或之后，但不能与 tbody 组合在一起。|-->
  <!--| 无论 tfoot 在标记中的位置如何，渲染时都会显示在底部。|-->

  <tr>
    <td>页脚内容 1</td>
    <td>页脚内容 2</td>
  </tr>
</tfoot>

</table>
```

以下示例的结果展示了两张——第一个表格没有任何样式，第二个表格应用了一些CSS属性：background-color、color和border*。这些样式仅作为视觉参考，并非本主题的核心内容。

- width
- visibility
- display (as in display: none)
 - display: none; will actually remove the columns from the display, causing the table to render as if those cells don't exist

For more information, see [HTML5 Tabular data](#).

Section 8.4: Table with thead, tbody, tfoot, and caption

HTML also provides the tables with the <thead>, <tbody>, <tfoot>, and <caption> elements. These additional elements are useful for adding semantic value to your tables and for providing a place for separate CSS styling.

When printing out a table that doesn't fit onto one (paper) page, most browsers repeat the contents of <thead> on every page.

There's a specific order that must be adhered to, and we should be aware that not every element falls into place as one would expect. The following example demonstrates how our 4 elements should be placed.

```
<table>
  <caption>Table Title</caption> <!--| caption is the first child of table |-->
  <thead> <!--=====| thead is after caption |-->
    <tr>
      <th>Header content 1</th>
      <th>Header content 2</th>
    </tr>
  </thead>

  <tbody> <!--=====| tbody is after thead |-->
    <tr>
      <td>Body content 1</td>
      <td>Body content 2</td>
    </tr>
  </tbody>

  <tfoot><!--| tfoot can be placed before or after tbody, but not in a group of tbody. |-->
  <!--| Regardless where tfoot is in markup, it is rendered at the bottom. |-->

  <tr>
    <td>Footer content 1</td>
    <td>Footer content 2</td>
  </tr>
</tfoot>

</table>
```

The following example's results are demonstrated twice--the first table lacks any styles, the second table has a few CSS properties applied: background-color, color, and border*. The styles are provided as a visual guide and is not an essential aspect of the topic at hand.

Table Title	
Header content 1	Header content 2
Body content 1	Body content 2
Footer content 1	Footer content 2

Table Title	
Header content 1	Header content 2
Body content 1	Body content 2
Footer content 1	Footer content 2

元素	应用的样式
<caption>	黑色背景上的黄色文字。
<thead>	紫色背景上的加粗文字。
<tbody>	蓝色背景上的文字。
<tfoot>	绿色背景上的文字。
<th>	橙色边框。
<td>	红色边框。

第8.5节：标题范围

th 元素非常常用于表示表格行和列的标题，如下所示：

```
<table>
  <thead>
    <tr>
      <td></td>
      <th>列标题1</th>
      <th>列标题2</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>行标题1</th>
      <td></td>
      <td></td>
    </tr>
    <tr>
      <th>行标题2</th>
      <td></td>
      <td></td>
    </tr>
  </tbody>
</table>
```

通过使用scope属性，可以提高无障碍性。上述示例将修改为如下：

```
<table>
  <thead>
    <tr>
      <td></td>
      <th scope="col">列标题 1</th>
      <th scope="col">列标题 2</th>
```

Table Title	
Header content 1	Header content 2
Body content 1	Body content 2
Footer content 1	Footer content 2

Table Title	
Header content 1	Header content 2
Body content 1	Body content 2
Footer content 1	Footer content 2

Element	Styles Applies
<caption>	Yellow text on black background.
<thead>	Bold text on purple background.
<tbody>	Text on blue background.
<tfoot>	Text on green background.
<th>	Orange borders.
<td>	Red borders.

Section 8.5: Heading scope

th elements are very commonly used to indicate headings for table rows and columns, like so:

```
<table>
  <thead>
    <tr>
      <td></td>
      <th>Column Heading 1</th>
      <th>Column Heading 2</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>Row Heading 1</th>
      <td></td>
      <td></td>
    </tr>
    <tr>
      <th>Row Heading 2</th>
      <td></td>
      <td></td>
    </tr>
  </tbody>
</table>
```

This can be improved for accessibility by the use of the scope attribute. The above example would be amended as follows:

```
<table>
  <thead>
    <tr>
      <td></td>
      <th scope="col">Column Heading 1</th>
      <th scope="col">Column Heading 2</th>
```



```
        </tr>
    </thead>
    <tbody>
        <tr>
            <th scope="row">行标题 1</th>
            <td></td>
            <td></td>
        </tr>
        <tr>
            <th scope="row">行标题 1</th>
            <td></td>
            <td></td>
        </tr>
    </tbody>
</table>
```

scope 被称为枚举属性，意味着它的值可以是特定可能值集合中的一个。
该集合包括：

- col
- row
- colgroup
- rowgroup

参考文献：

- <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/th#attr-scope>
- <https://www.w3.org/TR/WCAG20-TECHS/H63.html>

```
        </tr>
    </thead>
    <tbody>
        <tr>
            <th scope="row">Row Heading 1</th>
            <td></td>
            <td></td>
        </tr>
        <tr>
            <th scope="row">Row Heading 1</th>
            <td></td>
            <td></td>
        </tr>
    </tbody>
</table>
```

scope is known as an *enumerated attribute*, meaning that it can have a value from a specific set of possible values.
This set includes:

- col
- row
- colgroup
- rowgroup

References:

- <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/th#attr-scope>
- <https://www.w3.org/TR/WCAG20-TECHS/H63.html>

第9章：注释

与其他编程语言、标记语言和降价语言类似，HTML中的注释为其他开发者提供开发相关的信息，而不会影响用户界面。然而，与其他语言不同的是，HTML注释可以用于指定仅供Internet Explorer使用的HTML元素。本章节解释了如何编写HTML注释及其功能应用。

第9.1节：创建注释

HTML注释可用于给自己或其他开发者留下关于代码中特定点的备注。它们可以以<!--开始，以-->结束，如下所示：

```
<!-- 我是一个HTML注释！ -->
```

它们可以内嵌在其他内容中：

```
<h1>这部分将被显示 <!-- 而这部分将不会被显示 -->.</h1>
```

它们也可以跨多行以提供更多信息：

```
<!-- 这是一个多行HTML注释。
无论这里有什么，浏览器都不会渲染。
你可以“注释掉”整段HTML代码。
-->
```

但是，它们不能出现在另一个HTML标签内，比如这样：

```
<h1 <!-- testAttribute="something" -->>这将无法工作</h1>
```

这会产生无效的HTML，因为整个<h1 <!-- testAttribute="something" -->块会被视为一个单一的开始标签 h1，里面包含一些其他无效信息，后面跟着一个单独的>闭合括号但没有任何作用。

为了兼容尝试将HTML解析为XML或SGML的工具，注释内容中不应包含两个连字符--。

第9.2节：注释掉内联元素之间的空白

内联显示元素，通常如span或a，会在文档中包含其前后最多一个空白字符。为了避免标记中出现非常长的行（难以阅读）和无意的空白（影响格式），可以将空白注释掉。

```
<!-- 使用HTML注释来抵消下面的换行符： -->
<a href="#">我希望这之后不会有多余的空白！</a><!--
--><button>Foo</button>
```

尝试在内联元素之间不加注释，这样它们之间会有一个空格。有时需要保留空格字符。

示例代码：

```
<!-- 使用HTML注释来抵消下面的换行符： -->
```

Chapter 9: Comments

Similar to other programming, markup, and markdown languages, comments in HTML provide other developers with development specific information without affecting the user interface. Unlike other languages however, HTML comments can be used to specify HTML elements for Internet Explorer only. This topic explains how to write HTML comments, and their functional applications.

Section 9.1: Creating comments

HTML comments can be used to leave notes to yourself or other developers about a specific point in code. They can be initiated with <!-- and concluded with -->, like so:

```
<!-- I'm an HTML comment! -->
```

They can be incorporated inline within other content:

```
<h1>This part will be displayed <!-- while this will not be displayed -->.</h1>
```

They can also span multiple lines to provide more information:

```
<!-- This is a multiline HTML comment.
Whatever is in here will not be rendered by the browser.
You can "comment out" entire sections of HTML code.
-->
```

However, they **cannot** appear within another HTML tag, like this:

```
<h1 <!-- testAttribute="something" -->>This will not work</h1>
```

This produces invalid HTML as the entire <h1 <!-- testAttribute="something" --> block would be considered a single start tag h1 with some other invalid information contained within it, followed by a single > closing bracket that does nothing.

For compatibility with tools that try to parse HTML as XML or SGML, the body of your comment should not contain two dashes --.

Section 9.2: Commenting out whitespace between inline elements

Inline display elements, usually such as span or a, will include up to one white-space character before and after them in the document. In order to avoid very long lines in the markup (that are hard to read) and unintentional white-space (which affects formatting), the white-space can be commented out.

```
<!-- Use an HTML comment to nullify the newline character below: -->
<a href="#">I hope there will be no extra whitespace after this!</a><!--
--><button>Foo</button>
```

Try it without a comment between the inline elements, and there will be one space between them. Sometimes picking up the space character is desired.

Example code:

```
<!-- Use an HTML comment to nullify the newline character below: -->
```

```
<a href="#">我希望这之后不会有多余的空白！</a><!--
--><button>Foo</button>
<hr>
<!-- 如果没有它，你会注意到一个小的格式差异： -->
<a href="#">我希望这之后不会有多余的空白！</a>
<button>Foo</button>
```

输出：



```
<a href="#">I hope there will be no extra whitespace after this!</a><!--
--><button>Foo</button>
<hr>
<!-- Without it, you can notice a small formatting difference: -->
<a href="#">I hope there will be no extra whitespace after this!</a>
<button>Foo</button>
```

Output:



第10章：类和ID

参数	详情
class	表示元素的类别（非唯一）
id	表示元素的ID（在相同上下文中唯一）

类和ID使得从脚本和样式表中引用HTML元素更加容易。class属性可以用于一个或多个标签，并被CSS用于样式设置。然而，ID旨在引用单个元素，意味着同一个ID绝不应被使用两次。ID通常与JavaScript和内部文档链接一起使用，而在CSS中不推荐使用。本主题包含有关HTML中class和ID属性正确使用的有用解释和示例。

第10.1节：为元素赋予类

类是分配给元素的标识符。使用class属性为元素分配类。

```
<div class="example-class"></div>
```

要为元素分配多个类，请用空格分隔类名。

```
<div class="class1 class2"></div>
```

在CSS中使用类

类可以用于为某些元素设置样式，而不改变所有该类型的元素。例如，这两个span元素可以有完全不同的样式：

```
<span></span>
<span class="special"></span>
```

同名的类可以赋予页面上的任意数量的元素，它们都会获得与该类相关联的样式。除非你在CSS中指定了具体的元素，否则这总是成立。

例如，我们有两个元素，它们都带有类highlight：

```
<div class="highlight">Lorem ipsum</div>
<span class="highlight">Lorem ipsum</span>
```

如果我们的CSS如下，那么绿色将应用于两个元素内的文本：

```
.highlight { color: green; }
```

但是，如果我们只想针对带有类highlight的div元素，则可以像下面这样增加特异性：

```
div.highlight { color: green; }
```

然而，在使用 CSS 进行样式设计时，通常建议只使用类选择器（例如.highlight），而不是带类的元素选择器（例如div.highlight）。

与其他选择器一样，类选择器也可以嵌套：

```
.main .highlight { color: red; } /* 后代组合器 */
```

Chapter 10: Classes and IDs

Parameter	Details
class	Indicates the Class of the element (non-unique)
id	Indicates the ID of the element (unique in the same context)

Classes and IDs make referencing HTML elements from scripts and stylesheets easier. The class attribute can be used on one or more tags and is used by CSS for styling. IDs however are intended to refer to a single element, meaning the same ID should never be used twice. IDs are generally used with JavaScript and internal document links, and are discouraged in CSS. This topic contains helpful explanations and examples regarding proper usage of class and ID attributes in HTML.

Section 10.1: Giving an element a class

Classes are identifiers for the elements that they are assigned to. Use the class attribute to assign a class to an element.

```
<div class="example-class"></div>
```

To assign multiple classes to an element, separate the class names with spaces.

```
<div class="class1 class2"></div>
```

Using classes in CSS

Classes can be used for styling certain elements without changing all elements of that kind. For example, these two span elements can have completely different stylings:

```
<span></span>
<span class="special"></span>
```

Classes of the same name can be given to any number of elements on a page and they will all receive the styling associated with that class. This will always be true unless you specify the element within the CSS.

For example, we have two elements, both with the class highlight:

```
<div class="highlight">Lorem ipsum</div>
<span class="highlight">Lorem ipsum</span>
```

If our CSS is as below, then the color green will be applied to the text within both elements:

```
.highlight { color: green; }
```

However, if we only want to target div's with the class highlight then we can add specificity like below:

```
div.highlight { color: green; }
```

Nevertheless, when styling with CSS, it is generally recommended that only classes (e.g. .highlight) be used rather than elements with classes (e.g. div.highlight).

As with any other selector, classes can can be nested:

```
.main .highlight { color: red; } /* Descendant combinator */
```

```
.footer > .highlight { color: blue; } /* 子元素组合器 */
```

你也可以链式使用类选择器，只选择同时拥有多个类的元素。例如，如果这是我们的 HTML：

```
<div class="special left menu">这段文字将是粉色</div>
```

如果我们想把这段特定的文字设置为粉色，可以在 CSS 中这样写：

```
.special.left.menu { color: pink; }
```

第10.2节：给元素设置ID

元素的 ID 属性是文档中必须唯一的标识符。它的作用是在链接（使用锚点）、脚本编写或样式设计（使用 CSS）时唯一标识该元素。

```
<div id="example-id"></div>
```

在同一文档中不应有两个具有相同ID的元素，即使这些属性附加在两种不同类型的元素上。例如，以下代码是不正确的：

```
<div id="example-id"></div>
<span id="example-id"></span>
```

浏览器会尽力渲染这段代码，但在使用CSS进行样式设置或用JavaScript添加功能时，可能会出现意外的行为。

要在CSS中通过ID引用元素，请在ID前加上#。

```
#example-id { color: green; }
```

要跳转到页面中具有某个ID的元素，请在URL中附加#和元素名称。

```
http://example.com/about#example-id
```

此功能在大多数浏览器中受支持，无需额外的JavaScript或CSS即可生效。

第10.3节：可接受的值

对于ID

版本 ≥ 5

一个id的值唯一限制是：

1. 它在文档中必须唯一
2. 它不能包含任何空格字符
3. 它必须至少包含一个字符

因此，值可以是全数字，仅一个数字，仅标点符号，包含特殊字符，随意。但不能有空白字符。

以下是有效的：

```
<div id="container"> ... </div>
```

```
.footer > .highlight { color: blue; } /* Child combinator */
```

You can also chain the class selector to only select elements that have a combination of several classes. For example, if this is our HTML:

```
<div class="special left menu">This text will be pink</div>
```

And we want to colour this specific piece of text pink, we can do the following in our CSS:

```
.special.left.menu { color: pink; }
```

Section 10.2: Giving an element an ID

The ID attribute of an element is an identifier which must be unique in the whole document. Its purpose is to uniquely identify the element when linking (using an anchor), scripting, or styling (with CSS).

```
<div id="example-id"></div>
```

You should not have two elements with the same ID in the same document, even if the attributes are attached to two different kinds of elements. For example, the following code is incorrect:

```
<div id="example-id"></div>
<span id="example-id"></span>
```

Browsers will do their best to render this code, but unexpected behavior may occur when styling with CSS or adding functionality with JavaScript.

To reference elements by their ID in CSS, prefix the ID with #.

```
#example-id { color: green; }
```

To jump to an element with an ID on a given page, append # with the element name in the URL.

```
http://example.com/about#example-id
```

This feature is supported in most browsers and does not require additional JavaScript or CSS to work.

Section 10.3: Acceptable Values

For an ID

Version ≥ 5

The only restrictions on the value of an id are:

1. it must be unique in the document
2. it must not contain any space characters
3. it must contain at least one character

So the value can be all digits, just one digit, just punctuation characters, include special characters, whatever. Just no whitespace.

So these are valid:

```
<div id="container"> ... </div>
```



```
<div id="999"> ... </div>
<div id="%LV-||"> ... </div>
<div id="----V"> ... </div>
<div id="ꠤ~"> ... </div>
<div id="♥"> ... </div>
<div id="{ }"> ... </div>
<div id="©"> ... </div>
<div id="ⒶWa☆€~¥"> ... </div>
```

这是无效的：

```
<div id=" "> ... </div>
```

当在同一文档中包含时，这也是无效的：

```
<div id="results"> ... </div>
<div id="results"> ... </div>
```

版本 ≤ 4.01

一个 id 值必须以字母开头，后面只能跟随：

- 字母 (A-Z/a-z)
- 数字 (0-9)
- 连字符 ("-")
- 下划线 ("_")
- 冒号 (":")
- 句点 (".")

参考上面 HTML5 部分的第一组示例，只有一个是有有效的：

```
<div id="container"> ... </div>
```

以下这些也是有效的：

```
<div id="sampletext"> ... </div>
<div id="sample-text"> ... </div>
<div id="sample_text"> ... </div>
<div id="sample:text"> ... </div>
<div id="sample.text"> ... </div>
```

同样，如果它不是以字母（大写或小写）开头，则无效。

关于类

类的规则基本上与 id 相同。不同之处在于 class 值 不需要 在文档中唯一。

参考上面的示例，虽然在同一文档中这不是有效的：

```
<div id="results"> ... </div>
<div id="results"> ... </div>
```

这完全没问题：

```
<div class="results"> ... </div>
```

```
<div id="999"> ... </div>
<div id="%LV-||"> ... </div>
<div id="----V"> ... </div>
<div id="ꠤ~"> ... </div>
<div id="♥"> ... </div>
<div id="{ }"> ... </div>
<div id="©"> ... </div>
<div id="ⒶWa☆€~¥"> ... </div>
```

This is invalid:

```
<div id=" "> ... </div>
```

This is also invalid, when included in the same document:

```
<div id="results"> ... </div>
<div id="results"> ... </div>
```

Version ≤ 4.01

An id value must begin with a letter, which can then be followed only by:

- letters (A-Z/a-z)
- digits (0-9)
- hyphens ("-")
- underscores ("_")
- colons (":")
- periods (".")

Referring to the first group of examples in the HTML5 section above, only one is valid:

```
<div id="container"> ... </div>
```

These are also valid:

```
<div id="sampletext"> ... </div>
<div id="sample-text"> ... </div>
<div id="sample_text"> ... </div>
<div id="sample:text"> ... </div>
<div id="sample.text"> ... </div>
```

Again, if it doesn't start with a letter (uppercase or lowercase), it's not valid.

For a Class

The rules for classes are essentially the same as for an id. The difference is that class values *do not* need to be unique in the document.

Referring to the examples above, although this is not valid in the same document:

```
<div id="results"> ... </div>
<div id="results"> ... </div>
```

This is perfectly okay:

```
<div class="results"> ... </div>
```

```
<div class="results"> ... </div>
```

重要提示：ID 和 Class 值在 HTML 之外的处理方式

请记住，上述规则和示例仅适用于 HTML 环境中。

在 ID 或 class 的值中使用数字、标点符号或特殊字符，可能会在其他环境中引发问题，例如 CSS、JavaScript 和正则表达式。

例如，虽然以下 ID 在 HTML5 中是有效的：

```
<div id="9lions"> ... </div>
```

... 但在 CSS 中是无效的：

4.1.3 字符和大小写

在 CSS 中，标识符（包括选择器中的元素名、类名和 ID）只能包含字符 [a-zA-Z0-9] 以及 ISO 10646 字符 U+00A0 及以上，还有连字符 (-) 和下划线 (_)；**它们不能以数字、两个连字符或连字符后跟数字开头。**（加重强调）

在大多数情况下，您可以在字符具有限制或特殊含义的上下文中对其进行转义。

W3C 参考文献

- 3.2.5.1 id 属性
- 3.2.5.7 class 属性
- 6.2 SGML 基本类型

第10.4节：与重复 ID 相关的问题

有多个元素使用相同的 ID 是一个难以排查的问题。HTML 解析器通常会尝试渲染页面。通常不会出现错误，但页面可能会表现异常。

在此示例中：

```
<div id="aDiv">a</div>
<div id="aDiv">b</div>
```

CSS 选择器仍然有效

```
#aDiv {
  color: red;
}
```

但是 JavaScript 无法处理这两个元素：

```
var html = document.getElementById("aDiv").innerHTML;
```

在这种情况下，html 变量只包含第一个 div 的内容 ("a")。

```
<div class="results"> ... </div>
```

Important Note: How ID and Class values are treated outside of HTML

Keep in mind that the rules and examples above apply within the context of HTML.

Using numbers, punctuation or special characters in the value of an id or a class may cause trouble in other contexts, such as CSS, JavaScript and regular expressions.

For example, although the following id is valid in HTML5:

```
<div id="9lions"> ... </div>
```

... it is invalid in CSS:

4.1.3 Characters and case

In CSS, *identifiers* (including element names, classes, and IDs in selectors) can contain only the characters [a-zA-Z0-9] and ISO 10646 characters U+00A0 and higher, plus the hyphen (-) and the underscore (_); **they cannot start with a digit, two hyphens, or a hyphen followed by a digit.** (emphasis added)

In most cases you may be able to escape characters in contexts where they have restrictions or special meaning.

W3C References

- 3.2.5.1 The id attribute
- 3.2.5.7 The class attribute
- 6.2 SGML basic types

Section 10.4: Problems related to duplicated IDs

Having more than one element with the same ID is a hard to troubleshoot problem. The HTML parser will usually try to render the page in any case. Usually no error occurs. But the pace could end up in a mis-behaving web page.

In this example:

```
<div id="aDiv">a</div>
<div id="aDiv">b</div>
```

CSS selectors still work

```
#aDiv {
  color: red;
}
```

But JavaScript fails to handle both elements:

```
var html = document.getElementById("aDiv").innerHTML;
```

In this casehtml variable bears only the first div content ("a").

第11章：数据属性

值	描述
somevalue	指定属性的值（作为字符串）

第11.1节：旧浏览器支持情况

数据属性是在 HTML5 中引入的，所有现代浏览器都支持，但 HTML5 之前的旧浏览器不识别数据属性。

然而，在 HTML 规范中，浏览器不识别的属性必须保持原样，浏览器在渲染页面时会简单地忽略它们。

网页开发者利用这一事实创建了非标准属性，即不属于HTML规范的任何属性。例如，下面代码行中的value属性被视为非标准属性，因为标签的规范中没有value属性，且它也不是全局属性：

```

```

这意味着虽然旧版浏览器不支持数据属性，但它们仍然有效，您可以使用相同的通用JavaScript setAttribute 和 getAttribute 方法来设置和获取它们，但不能使用仅在现代浏览器中支持的新 dataset 属性。

第11.2节：数据属性的使用

HTML5 data-* 属性提供了一种方便的方式在HTML元素中存储数据。存储的数据可以通过JavaScript读取或修改

```
<div data-submitted="yes" class="user_profile">
  ... 一些内容 ...
</div>
```

- 数据属性的结构是 data-*，即数据属性的名称位于 data- 部分之后。通过这个名称，可以访问该属性。
- 可以使用 data-* 属性存储字符串格式的数据（包括 json）。

Chapter 11: Data Attributes

Value	Description
somevalue	Specifies the value of the attribute (as a string)

Section 11.1: Older browsers support

Data attributes were introduced in HTML5 which is supported by all modern browsers, but older browsers before HTML5 don't recognize the data attributes.

However, in HTML specifications, attributes that are not recognized by the browser must be left alone and the browser will simply ignore them when rendering the page.

Web developers have utilized this fact to create non-standard attributes which are any attributes not part of the HTML specifications. For example, the value attribute in the line bellow is considered a non-standard attribute because the specifications for the tag don't have a value attribute and it is not a global attribute:

```

```

This means that although data attributes are not supported in older browsers, they still work and you can set and retrieve them using the same generic JavaScript setAttribute and getAttribute methods, but you cannot use the new dataset property which is only supported in modern browsers.

Section 11.2: Data Attribute Use

HTML5 data-* attributes provide a convenient way to store data in HTML elements. The stored data can be read or modified using JavaScript

```
<div data-submitted="yes" class="user_profile">
  ... some content ...
</div>
```

- Data attribute structure is data-*, i.e. the name of the data attribute comes after the data- part. Using this name, the attribute can be accessed.
- Data in string format (including json) can be stored using data-* attribute.

第12章：链接资源

属性	详情
字符集	指定链接文档的字符编码
crossorigin	指定元素如何处理跨域请求
href	指定链接文档的位置
hreflang	指定链接文档中文本的语言
媒体	指定链接文档将在何种设备上显示，通常用于根据所用设备选择样式表
rel	必需。指定当前文档与链接文档之间的关系
rev	指定链接文档与当前文档之间的关系
尺寸	指定链接资源的大小。仅当 rel="icon" 时有效
target	指定链接文档加载的位置
类型	指定链接文档的媒体类型
完整性	指定链接资源的 base64 编码哈希值（sha256、sha384 或 sha512），允许浏览器验证其合法性。

虽然许多脚本、图标和样式表可以直接写入 HTML 标记中，但最佳实践和更高效的方法是将这些资源放在单独的文件中，并将它们链接到文档中。本主题涵盖如何将外部资源（如样式表和脚本）链接到 HTML 文档中。

第12.1节：JavaScript

同步

```
<script src="path/to.js"></script>
```

标准做法是将 JavaScript `<script>` 标签放置在关闭的 `</body>` 标签之前。最后加载脚本可以让你的网站视觉效果更快显示，并且避免你的 JavaScript 尝试与尚未加载的元素交互。

异步

```
<script src="path/to.js" async></script>
```

另一种选择是，当加载的 JavaScript 代码对页面初始化不是必需时，可以异步加载，从而加快页面加载速度。使用 `async` 时，浏览器会并行加载脚本内容并且一旦完全下载，会中断 HTML 解析以解析 JavaScript 文件。

延迟

```
<script src="path/to.js" defer></script>
```

延迟脚本类似于异步脚本，不同之处在于解析只会在 HTML 完全解析后进行。延迟脚本保证按照声明顺序加载，和同步脚本一样。

```
<noscript>
<noscript>JavaScript 已禁用</noscript>
```

`<noscript>` 元素定义了当用户禁用脚本或浏览器不支持脚本时显示的内容。`<noscript>` 标签可以放置在 `<head>` 或 `<body>` 中。

Chapter 12: Linking Resources

Attribute	Details
charset	Specifies the character encoding of the linked document
crossorigin	Specifies how the element handles cross origin requests
href	Specifies the location of the linked document
hreflang	Specifies the language of the text in the linked document
media	Specifies on what device the linked document will be displayed, often used with selecting stylesheets based on the device in question
rel	Required. Specifies the relationship between the current document and the linked document
rev	Specifies the relationship between the linked document and the current document
sizes	Specifies the size of the linked resource. Only when rel="icon"
target	Specifies where the linked document is to be loaded
type	Specifies the media type of the linked document
integrity	Specifies a base64 encoded hash (sha256, sha384, or sha512) of the linked resource allowing the browser to verify its legitimacy.

While many scripts, icons, and stylesheets can be written straight into HTML markup, it is best practice and more efficient to include these resources in their own file and link them to your document. This topic covers linking external resources such as stylesheets and scripts into an HTML document.

Section 12.1: JavaScript

Synchronous

```
<script src="path/to.js"></script>
```

Standard practice is to place JavaScript `<script>` tags just before the closing `</body>` tag. Loading your scripts last allows your site's visuals to show up more quickly and discourages your JavaScript from trying to interact with elements that haven't loaded yet.

Asynchronous

```
<script src="path/to.js" async></script>
```

Another alternative, when the Javascript code being loaded is not necessary for page initialization, it can be loaded asynchronously, speeding up the page load. Using `async` the browser will load the contents of the script in parallel and, once it is fully downloaded, will interrupt the HTML parsing in order to parse the Javascript file.

Deferred

```
<script src="path/to.js" defer></script>
```

Deferred scripts are like `async` scripts, with the exception that the parsing will only be performed once the HTML is fully parsed. Deferred scripts are guaranteed to be loaded in the order of declaration, same way as synchronous scripts.

```
<noscript>
<noscript>JavaScript disabled</noscript>
```

The `<noscript>` element defines content to be displayed if the user has scripts disabled or if the browser does not support using scripts. The `<noscript>` tag can be placed in either the `<head>` or the `<body>`.

第12.2节：外部CSS样式表

```
<link rel="stylesheet" href="path/to.css" type="text/css">
```

标准做法是在HTML顶部的<head>标签内放置CSS<link>标签。这样CSS会先被加载，并在页面加载时应用，而不是在CSS加载之前显示无样式的HTML。HTML5中不需要type属性，因为HTML5通常支持CSS。

```
<link rel="stylesheet" href="path/to.css" type="text/css">
```

和

```
<link rel="stylesheet" href="path/to.css">
```

... 在HTML5中也做同样的事情。

另一种较少见的做法是在直接的CSS中使用@import语句。如下所示：

```
<style type="text/css">
  @import("path/to.css")
</style>

<style>
  @import("path/to.css")
</style>
```

第12.3节：网站图标（Favicon）

```
<link rel="icon" type="image/png" href="/favicon.png">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
```

对于PNG文件，使用mime类型image/png，对于图标（*.ico）文件，使用image/x-icon。有关区别，请参见[this SO question](#)。

位于网站根目录下名为favicon.ico的文件通常会被自动加载和应用，无需使用<link>标签。如果该文件发生更改，浏览器在更新缓存时可能会比较缓慢且固执。

第12.4节：备用CSS

```
<link rel="alternate stylesheet" href="path/to/style.css" title="yourTitle">
```

某些浏览器允许应用备用样式表（alternate style sheets），如果提供的话。默认情况下它们不会被应用，但通常可以通过浏览器设置进行切换：

Firefox允许用户通过“查看 > 页面样式”子菜单选择样式表，Internet Explorer（从IE 8开始）也支持此功能，同样通过“查看 > 页面样式”访问（至少在IE 11中如此），但Chrome需要通过扩展程序才能使用该功能（截至版本48）。网页也可以提供自己的用户界面，允许用户切换样式。

（来源：[MDN Docs](#)）

Section 12.2: External CSS Stylesheet

```
<link rel="stylesheet" href="path/to.css" type="text/css">
```

The standard practice is to place CSS <link> tags inside the <head> tag at the top of your HTML. This way the CSS will be loaded first and will apply to your page as it is loading, rather than showing unstyled HTML until the CSS is loaded. The typeattribute is not necessary in HTML5, because HTML5 usually supports CSS.

```
<link rel="stylesheet" href="path/to.css" type="text/css">
```

and

```
<link rel="stylesheet" href="path/to.css">
```

... do the same thing in HTML5.

Another, though less common practice, is to use an @import statement inside direct CSS. Like this:

```
<style type="text/css">
  @import("path/to.css")
</style>

<style>
  @import("path/to.css")
</style>
```

Section 12.3: Favicon

```
<link rel="icon" type="image/png" href="/favicon.png">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
```

Use the mime-type image/png for PNG files and image/x-icon for icon (*.ico) files. For the difference, see [this SO question](#).

A file named favicon.ico at the root of your website will typically be loaded and applied automatically, without the need for a <link> tag. If this file ever changes, browsers can be slow and stubborn about updating their cache.

Section 12.4: Alternative CSS

```
<link rel="alternate stylesheet" href="path/to/style.css" title="yourTitle">
```

Some browsers allow alternate style sheets to apply if they are offered. By default they will not be applied, but usually they can be changed through the browser settings:

Firefox lets the user select the stylesheet using the View > Page Style submenu, Internet Explorer also supports this feature (beginning with IE 8), also accessed from View > Page Style (at least as of IE 11), but Chrome requires an extension to use the feature (as of version 48). The web page can also provide its own user interface to let the user switch styles.

(Source: the [MDN Docs](#))

第12.5节：资源提示：dns-prefetch、prefetch、prerender

预连接

预连接（preconnect）关系类似于DNS预取（dns-prefetch），因为它会解析DNS。然而，它还会进行TCP握手，并可选择进行TLS协商。这是一个实验性功能。

```
<link rel="preconnect" href="URL">
```

DNS预取

通知浏览器解析某个URL的DNS，以便该URL的所有资源加载更快。

```
<link rel="dns-prefetch" href="URL">
```

预取

通知浏览器应预取某个资源，以便更快加载。

```
<link rel="prefetch" href="URL">
```

DNS预取仅解析域名，而预取则下载/存储指定的资源。

预渲染

通知浏览器在后台获取并渲染该URL，以便当用户导航到该URL时能够即时呈现给用户。这是一个实验性功能。

```
<link rel="prerender" href="URL">
```

第12.6节：链接的 'media' 属性

```
<link rel="stylesheet" href="test.css" media="print">
```

media 指定了应为哪种类型的媒体使用哪种样式表。使用 print 值时，该样式表仅会用于打印页面的显示。

该属性的值可以是任何 mediatype 值（类似于CSS媒体查询）。

第12.7节：Prev 和 Next

当页面属于一系列文章的一部分时，例如，可以使用 prev 和 next 指向前后相邻的页面。

```
<link rel="prev" href="http://stackoverflow.com/documentation/java/topics">
```

```
<link rel="next" href="http://stackoverflow.com/documentation/css/topics">
```

第12.8节：网络订阅源

使用rel="alternate"属性以便发现您的Atom/RSS订阅源。

```
<link rel="alternate" type="application/atom+xml" href="http://example.com/feed.xml" />
<link rel="alternate" type="application/rss+xml" href="http://example.com/feed.xml" />
```

Section 12.5: Resource Hint: dns-prefetch, prefetch, prerender

Preconnect

The preconnect relationship is similar to dns-prefetch in that it will resolve the DNS. However, it will also make the TCP handshake, and optional TLS negotiation. This is an experimental feature.

```
<link rel="preconnect" href="URL">
```

DNS-Prefetch

Informs browsers to resolve the DNS for a URL, so that all assets from that URL load faster.

```
<link rel="dns-prefetch" href="URL">
```

Prefetch

Informs the browsers that a given resource should be prefetched so it can be loaded more quickly.

```
<link rel="prefetch" href="URL">
```

DNS-Prefetch resolves only the domain name whereas prefetch downloads/stores the specified resources.

Prerender

Informs browsers to fetch and render the URL in the background, so that they can be delivered to the user instantaneously as the user navigates to that URL. This is an experimental feature.

```
<link rel="prerender" href="URL">
```

Section 12.6: Link 'media' attribute

```
<link rel="stylesheet" href="test.css" media="print">
```

Media specifies what style sheet should be used for what type of media. Using the print value would only display that style sheet for print pages.

The value of this attribute can be any of the mediatype values (similar to a CSS media query).

Section 12.7: Prev and Next

When a page is part of a series of articles, for instance, one can use prev and next to point to pages that are coming before and after.

```
<link rel="prev" href="http://stackoverflow.com/documentation/java/topics">
```

```
<link rel="next" href="http://stackoverflow.com/documentation/css/topics">
```

Section 12.8: Web Feed

Use the rel="alternate" attribute to allow discoverability of your Atom/RSS feeds.

```
<link rel="alternate" type="application/atom+xml" href="http://example.com/feed.xml" />
<link rel="alternate" type="application/rss+xml" href="http://example.com/feed.xml" />
```

请参阅MDN文档，了解[RSS订阅源](#)和[Atomic RSS](#)。

See the MDN docs for [RSS feeds](#) and [Atomic RSS](#).

第13章：在

HTML中包含JavaScript代码

属性	详情
src	指定JavaScript文件的路径。可以是相对或绝对URL。
类型	指定MIME类型。此属性在HTML4中是必需的，但在HTML5中是可选的。
async	指定脚本应异步执行（仅适用于外部脚本）。该属性不需要任何值（XHTML除外）。
defer	指定脚本应在页面解析完成后执行（仅适用于外部脚本）。该属性不需要任何值（XHTML除外）。
字符集	指定外部脚本文件使用的字符编码，例如 UTF-8
crossorigin	元素如何处理跨域请求
nonce	用于内容安全策略（Content Security Policy）检查的加密随机数CSP3

第13.1节：处理禁用的 Javascript

客户端浏览器可能不支持 Javascript 或已禁用 Javascript 执行，可能是出于安全原因。为了能够告知用户页面中应执行脚本，可以使用<noscript>标签。只要当前页面禁用 Javascript，<noscript>的内容就会显示。

```
<script>
document.write("Hello, world!");

</script>

<noscript>此浏览器不支持Javascript。</noscript>
```

第13.2节：链接到外部JavaScript文件

```
<script src="example.js"></script>
```

src属性的作用类似于锚点的href属性：你可以指定绝对或相对URL。上面的示例链接到与HTML文档同一目录下的文件。通常这会被添加在<head>标签内，位于html文档的顶部

第13.3节：直接包含JavaScript代码

你也可以不链接外部文件，而是直接在HTML中包含JS代码：

```
<script>
// JavaScript代码
</script>
```

第13.4节：包含异步执行的JavaScript文件

```
<script type="text/javascript" src="URL" async></script>
```

Chapter 13: Include JavaScript Code in HTML

Attribute	Details
src	Specifies the path to a JavaScript file. Either a relative or absolute URL.
type	Specifies the MIME type. This attribute is required in HTML4, but optional in HTML5.
async	Specifies that the script shall be executed asynchronously (only for external scripts). This attribute does not require any value (except of XHTML).
defer	Specifies that the script shall be executed when the page has finished parsing (only for external scripts). This attribute does not require any value (except of XHTML).
charset	Specifies the character encoding used in an external script file, e.g. UTF-8
crossorigin	How the element handles crossorigin requests
nonce	Cryptographic nonce used in Content Security Policy checks CSP3

Section 13.1: Handling disabled Javascript

It is possible that the client browser does not support Javascript or have Javascript execution disabled, perhaps due to security reasons. To be able to tell users that a script is supposed to execute in the page, the <noscript> tag can be used. The content of <noscript> is displayed whenever Javascript is disabled for the current page.

```
<script>
  document.write("Hello, world!");
</script>
<noscript>This browser does not support Javascript.</noscript>
```

Section 13.2: Linking to an external JavaScript file

```
<script src="example.js"></script>
```

The src attribute works like the href attribute on anchors: you can either specify an absolute or relative URL. The example above links to a file inside the same directory of the HTML document. This is typically added inside the <head> tags at the top of the html document

Section 13.3: Directly including JavaScript code

Instead of linking to an external file, you can also include the JS code as-is in your HTML:

```
<script>
// JavaScript code
</script>
```

Section 13.4: Including a JavaScript file executing asynchronously

```
<script type="text/javascript" src="URL" async></script>
```

第14章：使用带有CSS的HTML

CSS为页面上的HTML元素提供样式。内联样式涉及在标签中使用style属性，但强烈不建议使用。内部样式表使用<style>标签，用于声明页面特定部分的规则。外部样式表可以通过<link>标签使用，该标签引用外部CSS文件并将规则应用于文档。本章节涵盖这三种附加方法的使用。

第14.1节：外部样式表的使用

在文档的<head>中使用link属性：

```
<head>
  <link rel="stylesheet" type="text/css" href="stylesheet.css">
</head>
```

你也可以使用通过内容分发网络（CDN）提供的网站样式表。（例如，Bootstrap）：

```
<head>
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiISiFeK1dGmJRAkycuHAHRg320mUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">
</head>
```

通常，你可以在框架的官方网站上找到CDN支持。

第14.2节：内部样式表

你也可以通过使用<style>标签在内部包含CSS元素：

```
<head>
  <style type="text/css">
    body {
      背景颜色：灰色；
    }
  </style>
</head>
```

程序中也可以包含多个内部样式表。

```
<head>
  <style type="text/css">
    body {
      背景颜色：灰色；
    }
  </style>

  <style type="text/css">
    p {
      背景颜色：蓝色；
    }
  </style>
</head>
```

Chapter 14: Using HTML with CSS

CSS provides styles to HTML elements on the page. Inline styling involves usage of the style attribute in tags, and is highly discouraged. Internal stylesheets use the <style> tag and are used to declare rules for directed portions of the page. External stylesheets may be used through a <link> tag which takes an external file of CSS and applies the rules to the document. This topic covers usage of all three methods of attachment.

Section 14.1: External Stylesheet Use

Use the link attribute in the document's head:

```
<head>
  <link rel="stylesheet" type="text/css" href="stylesheet.css">
</head>
```

You can also use stylesheets provided from websites via a content delivery network, or CDN for short. (for example, Bootstrap):

```
<head>
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiISiFeK1dGmJRAkycuHAHRg320mUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">
</head>
```

Generally, you can find CDN support for a framework on its website.

Section 14.2: Internal Stylesheet

You can also include CSS elements internally by using the <style> tag:

```
<head>
  <style type="text/css">
    body {
      background-color: gray;
    }
  </style>
</head>
```

Multiple internal stylesheets can be included in a program as well.

```
<head>
  <style type="text/css">
    body {
      background-color: gray;
    }
  </style>

  <style type="text/css">
    p {
      background-color: blue;
    }
  </style>
</head>
```

第14.3节：内联样式

你可以使用style属性为特定元素设置样式：

```
<span style="color: red">这段文字将显示为红色。</span>
```

注意：尽量避免这样做——CSS的目的是将内容与表现分离。

第14.4节：多个样式表

可以加载多个样式表：

```
<head>
  <link rel="stylesheet" type="text/css" href="general.css">
  <link rel="stylesheet" type="text/css" href="specific.css">
</head>
```

注意，后面的文件和声明会覆盖前面的内容。所以如果general.css包含：

```
body {
background-color: red;
}
```

而 specific.css包含：

```
body {
background-color: blue;
}
```

如果两者都被使用，文档的背景将是蓝色。

Section 14.3: Inline Style

You can style a specific element by using the style attribute:

```
<span style="color: red">This text will appear in red.</span>
```

Note: Try to avoid this -- the point of CSS is to separate content from presentation.

Section 14.4: Multiple Stylesheets

It's possible to load multiple stylesheets:

```
<head>
  <link rel="stylesheet" type="text/css" href="general.css">
  <link rel="stylesheet" type="text/css" href="specific.css">
</head>
```

Note that **later files and declarations will override earlier ones**. So if general.css contains:

```
body {
  background-color: red;
}
```

and specific.css contains:

```
body {
  background-color: blue;
}
```

if both are used, the background of the document will be blue.

第15章：图像

参数	详情
src	指定图像的URL
srcset	在不同情况下使用的图像（例如，高分辨率显示器、小型显示器等）
尺寸	断点之间的图像尺寸
crossorigin	元素如何处理跨域请求
usemap	要使用的图像映射名称
ismap	图像是否为服务器端图像映射
alt	如果由于某种原因图像无法显示，应显示的替代文本
宽度	指定图像的宽度（可选）
高度	指定图像的高度（可选）

第15.1节：创建图像

要向页面添加图像，请使用图像标签。

图像标签（img）没有闭合标签。你给img标签的两个主要属性是src，图像来源和alt，描述图像的替代文本。

```

```

你也可以从网页URL获取图像：

```

```

注意：图像并非技术上插入到HTML页面中，图像是链接到HTML页面的。标签为引用的图像创建一个占位空间。

也可以使用 base64 直接将图像嵌入页面中：

```

```

提示：要将图像链接到另一个文档，只需将标签嵌套在<a>标签内。

第15.2节：选择替代文本

替代文本用于视觉障碍用户的屏幕阅读器和搜索引擎。因此，为您的图像编写良好的替代文本非常重要。

即使用替代属性替换图像，文本也应看起来正确。例如：

```
<!-- 错误 -->
 一位匿名用户写道：
<blockquote>Lorem ipsum dolor sed.</blockquote>
<a href="https://google.com/"></a> /
<a href="https://google.com/"></a>
```

没有图像时，这将显示为：

匿名用户头像 一位匿名用户写道：

Chapter 15: Images

Parameters	Details
src	Specifies the URL of the image
srcset	Images to use in different situations (e.g., high-resolution displays, small monitors, etc)
sizes	Image sizes between breakpoints
crossorigin	How the element handles crossorigin requests
usemap	Name of image map to use
ismap	Whether the image is a server-side image map
alt	Alternative text that should be displayed if for some reason the image could not be displayed
width	Specifies the width of the image (optional)
height	Specifies the height of the image (optional)

Section 15.1: Creating an image

To add an image to a page, use the image tag.

Image tags (img) do not have closing tags. The two main attributes you give to the img tag are src, the image source and alt, which is alternative text describing the image.

```

```

You can also get images from a web URL:

```

```

Note: Images are not technically inserted into an HTML page, images are linked to HTML pages. The tag creates a holding space for the referenced image.

It is also possible to embed images directly inside the page using base64:

```

```

Tip: To link an image to another document, simply nest the tag inside <a> tags.

Section 15.2: Choosing alt text


Alt-text is used by screen readers for visually impaired users and by search engines. It's therefore important to write good alt-text for your images.

The text should look correct even if you replace the image with its alt attribute. For example:

```
<!-- Incorrect -->
 An anonymous user wrote:
<blockquote>Lorem ipsum dolor sed.</blockquote>
<a href="https://google.com/"></a> /
<a href="https://google.com/"></a>
```

Without the images, this would look like:

Anonymous user avatar An anonymous user wrote:

 虚构文本示例。


[编辑图标](#) / [删除图标](#)

为纠正此问题：

- 移除头像的替代文本。此图片为有视力的用户提供信息（一个易于识别的图标表示用户是匿名的），但这些信息已在文本中提供。1
- 移除图标替代文本中的“图标”字样。知道这本来是个图标并不能帮助传达其实际用途。

```
<!-- 正确 -->
 一位匿名用户写道：
<blockquote>虚构文本示例。</blockquote>
<a href="https://google.com/"></a> /
<a href="https://google.com/"></a>
```

一位匿名用户写道：

 虚构文本示例。

[编辑](#) / [删除](#)

脚注

1 包含空的 alt 属性与完全不包含 alt 属性在语义上是有区别的。空的 alt 属性表示该图像不是内容的关键部分（如本例所示——它只是一个附加图像，并非理解其余内容所必需），因此可以省略渲染。然而，缺少 alt 属性则表示该图像是内容的关键部分，且没有可用的文本等价物进行渲染。

第15.3节：使用 srcset 属性的响应式图像

使用 srcset 和 sizes


```

```

sizes 类似于媒体查询，描述图像占据视口的空间大小。

- 如果视口宽度大于1200px，图像宽度正好为580px（例如我们的内容居中在最大宽度为1200px的容器中，图像占据其一半减去边距）。
- 如果视口宽度在640px到1200px之间，图像占据视口宽度的48%（例如图像随页面缩放，占据视口宽度的一半减去边距）。
- 如果视口是其他尺寸，在我们的例子中小于640像素，图片占视口的98%（例如图片随着页面缩放，占据视口的全宽减去边距）。最后一项必须省略媒体条件。

srcset只是告诉浏览器我们有哪些图片可用，以及它们的尺寸。

 Lorem ipsum dolor sed.


[Edit icon](#) / [Delete icon](#)

To correct this:

- Remove the alt-text for the avatar. This image adds information for sighted users (an easily identifiable icon to show that the user is anonymous) but this information is already available in the text.1
- Remove the "icon" from the alt-text for the icons. Knowing that this would be an icon if it were there does not help to convey its actual purpose.

```
<!-- Correct -->
 An anonymous user wrote:
<blockquote>Lorem ipsum dolor sed.</blockquote>
<a href="https://google.com/"></a> /
<a href="https://google.com/"></a>
```

An anonymous user wrote:

 Lorem ipsum dolor sed.

[Edit](#) / [Delete](#)

Footnotes

1 There is a semantic difference between including an empty alt attribute and excluding it altogether. An empty alt attribute indicates that the image is *not* a key part of the content (as is true in this case - it's just an additive image that is not necessary to understand the rest) and thus may be omitted from rendering. However, the lack of an alt attribute indicates that the image *is* a key part of the content and that there simply is no textual equivalent available for rendering.

Section 15.3: Responsive image using the srcset attribute

Using srcset with sizes

```

```

sizes are like media queries, describing how much space the image takes of the viewport.

- if viewport is larger than 1200px, image is exactly 580px (for example our content is centered in container which is max 1200px wide. Image takes half of it minus margins).
- if viewport is between 640px and 1200px, image takes 48% of viewport (for example image scales with our page and takes half of viewport width minus margins).
- if viewport is any other size , in our case less than 640px, image takes 98% of viewport (for example image scales with our page and takes full width of viewport minus margins). **Media condition must be omitted for last item.**

srcset is just telling the browser what images we have available, and what are their sizes.

- `img/hello-300.jpg`宽度为300像素,
- `img/hello-600.jpg`宽度为600像素,
- `img/hello-900.jpg`宽度为900像素,
- `img/hello-1200.jpg`宽度为1200像素

`src`始终是必需的图片来源。如果与 `srcset`一起使用, `src`将在浏览器不支持 `srcset`时提供备用图片。

使用不带sizes的srcset

```

```

`srcset`提供可用图片列表, 带有设备像素比 `x`描述符。

- 如果设备像素比为1, 使用`img/hello-300.jpg`
- 如果设备像素比为2, 使用`img/hello-600.jpg`
- 如果设备像素比为3, 使用`img/hello-1200.jpg`

`src`始终是必需的图片来源。如果与 `srcset`一起使用, `src`将在浏览器不支持 `srcset`时提供备用图片。

第15.4节：使用picture元素的响应式图片

代码

```
<picture>
  <source media="(min-width: 600px)" srcset="large_image.jpg">
  <source media="(min-width: 450px)" srcset="small_image.jpg">
  
</picture>
```

用法

为了在不同屏幕宽度下显示不同的图片, 必须像上面示例中那样, 在`picture`标签中使用`source`标签包含所有图片。

结果

- 在屏幕宽度大于600像素时, 显示`large_image.jpg`
- 在屏幕宽度大于450像素时, 显示`small_image.jpg`
- 在其他屏幕宽度时, 显示`default_image.jpg`

- `img/hello-300.jpg` is 300px wide,
- `img/hello-600.jpg` is 600px wide,
- `img/hello-900.jpg` is 900px wide,
- `img/hello-1200.jpg` is 1200px wide

`src` is always mandatory image source. In case of using with `srcset`, `src` will serve fallback image in case browser is not supporting `srcset`.

Using srcset without sizes

```

```

`srcset` provides list of available images, with device-pixel ratio `x` descriptor.

- if device-pixel ratio is 1, use `img/hello-300.jpg`
- if device-pixel ratio is 2, use `img/hello-600.jpg`
- if device-pixel ratio is 3, use `img/hello-1200.jpg`

`src` is always mandatory image source. In case of using with `srcset`, `src` will serve fallback image in case browser is not supporting `srcset`.

Section 15.4: Responsive image using picture element

Code

```
<picture>
  <source media="(min-width: 600px)" srcset="large_image.jpg">
  <source media="(min-width: 450px)" srcset="small_image.jpg">
  
</picture>
```

Usage

To display different images under different screen width, you must include all images using the `source` tag in a `picture` tag as shown in the above example.

Result

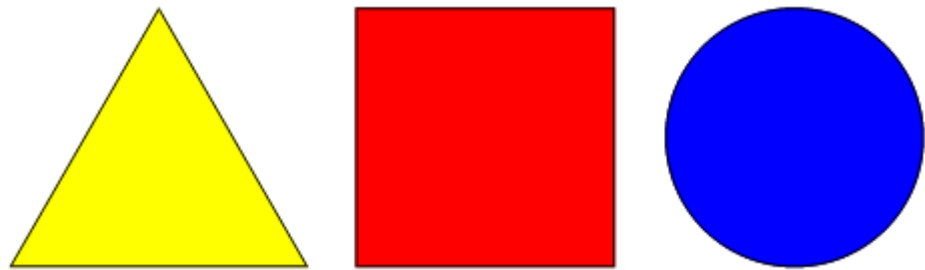
- On screens with screen width >600px, it shows `large_image.jpg`
- On screens with screen width >450px, it shows `small_image.jpg`
- On screens with other screen width, it shows `default_image.jpg`

第16章：图像映射

标签/属性	值
	以下是与一起使用的图像映射特定属性。常规的属性同样适用。
usemap	映射的name，前面加上井号。例如，对于一个name="map"的映射，图像应设置usemap="#map"。
<map>	
name	用于标识映射的名称。与图像的usemap属性配合使用。
<area>	以下是<area>特定属性。当指定了href，使<area>成为链接时，<area>还支持锚点标签（<a>）的所有属性，除了ping。详见MDN文档。
alt	如果不支持图像时显示的替代文本。仅当<area>上也设置了href时才需要。
coords	定义可选区域的坐标。当shape="polygon"时，应设置为由逗号分隔的“x, y”对的列表（即shape="polygon"coords="x1, y1, x2, y2, x3, y3, ..."）。当shape="rectangle"时，应设置为left, top, right, bottom。当shape="circle"时，应设置为centerX, centerY, radius。
href	超链接的URL，如果指定了。如果省略，则<area>不会表示超链接。
形状	<area>的形状。可以设置为default以选择整个图像（不需要coords属性），circle或circ表示圆形，rectangle或rect表示矩形，polygon或poly表示由角点指定的多边形区域。

第16.1节：图像映射简介

描述	图像映射是带有可点击区域的图像，这些区域通常作为超链接。
基本示例	要创建一个图像映射，使下图中的每个形状都可点击：



代码如下：

```

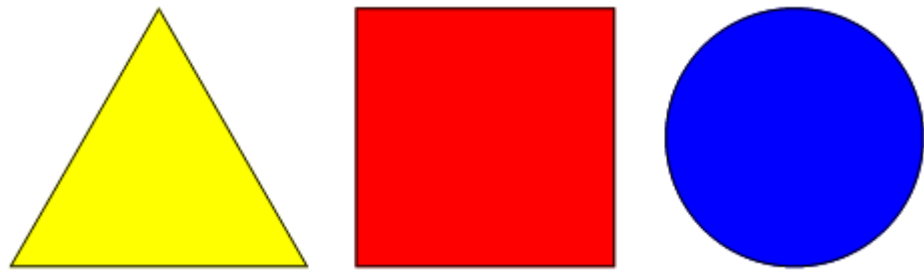
<map name="shapes">
  <area shape="polygon" coords="79,6,5,134,153,134">
  <area shape="rectangle" coords="177,6,306,134">
  <area shape="circle" coords="397,71,65">
```

Chapter 16: Image Maps

Tag/Attribute	Value
	Below are the image map-specific attributes to use with . Regular attributes apply.
usemap	The name of the map with a hash symbol prepended to it. For example, for a map with name="map", the image should have usemap="#map".
<map>	
name	The name of the map to identify it. To be used with the image's usemap attribute.
<area>	Below are <area>-specific attributes. When href is specified, making the <area> a link, <area> also supports all of the attributes of the anchor tag (<a>) except ping. See them at the MDN docs.
alt	The alternate text to display if images are not supported. This is only necessary if href is also set on the <area>.
coords	The coordinates outlining the selectable area. When shape="polygon", this should be set to a list of "x, y" pairs separated by commas (i.e., shape="polygon" coords="x1, y1, x2, y2, x3, y3, ..."). When shape="rectangle", this should be set to left, top, right, bottom. When shape="circle", this should be set to centerX, centerY, radius.
href	The URL of the hyperlink, if specified. If it is omitted, then the <area> will not represent a hyperlink.
shape	The shape of the <area>. Can be set to default to select the entire image (no coords attribute necessary), circle or circ for a circle, rectangle or rect for a rectangle, and polygon or poly for a polygonal area specified by corner points.

Section 16.1: Introduction to Image Maps

Description	An image maps is an image with clickable areas that usually act as hyperlinks.
Basic Example	To create an image map so that each of the shapes in the image below are clickable:



The code would be as follows:

```

<map name="shapes">
  <area shape="polygon" coords="79,6,5,134,153,134">
  <area shape="rectangle" coords="177,6,306,134">
  <area shape="circle" coords="397,71,65">
```

</map >

当光标变成指针时，您应该能看到浏览器识别这些区域。请参见

JSFiddle上的实时演示



</map>

You should see that the browser recognizes the areas when the cursor becomes a pointer. See a [live demo](#) on JSFiddle

第17章：输入控件元素

参数	详情
class	表示输入类
id	表示输入ID
类型	标识要显示的输入控件类型。可接受的值有hidden、text、tel、url、email、password、date、time、number、range、color、checkbox、radio、file、submit、image、reset 和 button。如果未指定，值无效，或浏览器不支持指定的类型，则默认为text。
name	表示输入的名称
禁用	布尔值，表示输入应被禁用。禁用的控件无法编辑，不会在表单提交时发送，也无法获得焦点。
选中	当type属性的值为radio或checkbox时，存在此布尔属性表示控件默认被选中；否则忽略。
多选	HTML5 表示可以传递多个文件或值（仅适用于file和email类型的输入）
占位符	HTML5 向用户提示控件中可以输入的内容。占位符文本不得包含回车或换行符
autocomplete	HTML5 指示控件的值是否可以由浏览器自动完成。
readonly	布尔值，表示输入不可编辑。只读控件仍会在表单提交时发送，但不会获得焦点。 HTML5: 当type属性的值设置为hidden、range、color、checkbox、radio、file或button时，此属性将被忽略。
required	HTML5 指示必须存在一个值或元素必须被选中，表单才能提交
alt	图像的替代文本，以防图像未显示。
autofocus	页面加载时，<input> 元素应获得焦点。
value	指定<input> 元素的值。
步骤	step 属性指定合法的数字间隔。它适用于以下输入类型：number、range、date、datetime-local、month、time 和 week。

交互式网页系统的关键组成部分，input 标签是设计用来接收用户特定形式输入的 HTML 元素。不同类型的输入元素可以规范输入的数据以符合指定格式，并为密码输入提供安全保障。

第17.1节：文本

最基本的输入类型，如果未指定 type，则默认为此类型。该输入类型定义了一个单行文本框，输入值中的换行符会被自动移除。所有其他字符均可输入。<input>元素用于 <form> 元素内，声明允许用户输入数据的输入控件。

语法

```
<input type="text">
```

或者（不指定 type，使用默认属性）：

```
<input>
```

文本字段输入的默认宽度是20个字符。可以通过指定 size 属性的值来更改，如下所示：

```
<input type="text" size="50">
```

Chapter 17: Input Control Elements

Parameter	Details
class	Indicates the Class of the input
id	Indicates the ID of the input
type	Identifies the type of input control to display. Acceptable values are hidden, text, tel, url, email, password, date, time, number, range, color, checkbox, radio, file, submit, image, reset, and button. Defaults to text if not specified, if the value is invalid, or if the browser does not support the type specified.
name	Indicates the name of the input
disabled	Boolean value that indicates the input should be disabled. Disabled controls cannot be edited, are not sent on form submission, and cannot receive focus.
checked	When the value of the type attribute is radio or checkbox, the presence of this Boolean attribute indicates that the control is selected by default; otherwise it is ignored.
multiple	HTML5 Indicates multiple files or values can be passed (Applies only to file and email type inputs)
placeholder	HTML5 A hint to the user of what can be entered in the control . The placeholder text must not contain carriage returns or line-feeds
autocomplete	HTML5 Indicates whether the value of the control can be automatically completed by the browser.
readonly	Boolean value that indicates the input is not editable. Readonly controls are still sent on form submission, but will not receive focus. HTML5: This attribute is ignored when the value of type attribute is either set to hidden, range, color, checkbox, radio, file or button.
required	HTML5 Indicates a value must be present or the element must be checked in order for the form to be submitted
alt	An alternative text for images, in case they are not displayed.
autofocus	The <input> element should get the focus when page loads.
value	Specifies the value of <input> element.
step	The step attribute specifies the legal number intervals. It works with the following input types: number, range, date, datetime-local, month, time and week.

A key component of interactive web systems, input tags are HTML elements designed to take a specific form of input from users. Different types of input elements can regulate the data entered to fit a specified format and provide security to password entry.

Section 17.1: Text

The most basic input type and the default input if no type is specified. This input type defines a single-line text field with line-breaks automatically removed from the input value. All other characters can be entered into this. <input> elements are used within a <form> element to declare input controls that allow users to input data.

Syntax

```
<input type="text">
```

or (without specifying a type, using the default attribute):

```
<input>
```

The default width of a text field input is 20 characters. This can be changed by specifying a value for the size attribute like this:

```
<input type="text" size="50">
```

size属性与使用CSS设置宽度有明显不同。使用宽度定义了一个具体的值（以像素数、父元素的百分比等为单位），输入框的宽度始终固定。使用 size则根据所用字体和字符的通常宽度计算分配的宽度。

注意：使用 size属性本身并不限制可以输入的字符数，只限制输入框显示的宽度。要限制长度，请参见输入验证。

输入字段只允许一行文本。如果需要多行文本输入以输入大量文本，请使用 <textarea>元素代替。

第17.2节：复选框和单选按钮

概述

复选框和单选按钮使用HTML标签<input>编写，其行为在[HTML规范](#)中定义。

最简单的复选框或单选按钮是带有type属性为checkbox或radio的<input>元素，分别如下：

```
<input type="checkbox">
<input type="radio">
```

单个独立的复选框元素用于单一的二元选项，例如是或否的问题。复选框是独立的，意味着用户可以在一组复选框中选择任意多个选项。换句话说，选中一个复选框不会取消选中同组中的其他复选框。

单选按钮通常成组出现（如果没有与其他单选按钮分组，您可能本应使用复选框），通过在该组内所有按钮上使用相同的name属性来标识。单选按钮的选择是互斥的，意味着用户只能从一组单选按钮中选择一个选项。

当选中一个单选按钮时，之前选中的同一name属性的其他单选按钮会被取消选中。

示例：

```
<input type="radio" name="color" id="red" value="#F00">
<input type="radio" name="color" id="green" value="#0F0">
<input type="radio" name="color" id="blue" value="#00F">
```

在显示时，单选按钮呈现为一个圆圈（未选中）或填充圆圈（选中）。复选框呈现为一个方框（未选中）或填充方框（选中）。根据浏览器和操作系统的不同，方框有时会带有圆角。

属性

复选框和单选按钮有许多属性用于控制其行为：

value

像其他输入元素一样，value属性指定在表单提交时与按钮关联的字符串值。然而，复选框和单选按钮有特殊之处，当省略该值时，提交时默认值为on，而不是发送空值。该value属性不会反映在按钮的外观上。

选中

The size attribute is distinctly different than setting a width with CSS. Using a width defines a specific value (in number of pixel, percentage of the parent element, etc.) that the input must always be wide. Using the size calculates the amount of width to allocate based on the font being used and how wide the characters normally are.

Note: Using the size attribute does not inherently limit the number of characters which can be entered into the box, only how wide the box is displayed. For limiting the length, see Input Validation.

An input field only allows one line of text. If you need a multi-line text input for substantial amount of text, use a <textarea> element instead.

Section 17.2: Checkbox and Radio Buttons

Overview

Checkboxes and radio buttons are written with the HTML tag <input>, and their behavior is defined in the [HTML specification](#).

The simplest checkbox or radio button is an <input> element with a type attribute of checkbox or radio, respectively:

```
<input type="checkbox">
<input type="radio">
```

A single stand-alone checkbox element is used for a single binary option such as a yes-or-no question. Checkboxes are independent, meaning the user may select as many choices as they would like in a group of checkboxes. In other words, checking one checkbox does *not* uncheck the other checkboxes in checkbox group.

Radio buttons usually come in groups (if it's not grouped with another radio button, you probably meant to use a checkbox instead) identified by using the same name attribute on all buttons within that group. The selection of radio buttons are *mutually exclusive*, meaning the user may only select one choice from a group of radio buttons. When a radio button is checked, any other radio button with the same name that was previously checked becomes unchecked.

Example:

```
<input type="radio" name="color" id="red" value="#F00">
<input type="radio" name="color" id="green" value="#0F0">
<input type="radio" name="color" id="blue" value="#00F">
```

When viewed, radio buttons appear as a circle (unchecked) or a filled circle (checked). Checkboxes appear as a square (unchecked) or a filled square (checked). Depending on the browser and operating system, the square sometimes has rounded corners.

Attributes

checkboxes and radio buttons have a number of attributes to control their behavior:

value

Like any other input element, the value attribute specifies the string value to associate with the button in the event of form submission. However, checkboxes and radio buttons are special in that when the value is omitted, it defaults to on when submitted, rather than sending a blank value. The value attribute is not reflected in the button's appearance.

checked

checked属性指定复选框或单选按钮的初始状态。它是一个布尔属性，可以省略。

以下都是定义选中单选按钮的有效且等效的方式：

```
<input checked>
<input checked="">
<input checked="checked">
<input checked="ChEcKeD">
```

未选中按钮的唯一有效语法是省略checked属性：

```
<input type="radio">
<input type="checkbox">
```

重置<form>时，复选框和单选按钮会恢复到其checked属性的状态。

辅助功能
标签

为了给按钮提供上下文并向用户展示每个按钮的用途，每个按钮都应有一个标签。可以使用<label>元素包裹按钮来实现。此外，这样标签是可点击的，点击标签即可选择对应的按钮。

示例：

```
<label>
  <input type="radio" name="color" value="#F00">
  红色
</label>
```

或者使用带有for属性且其值设置为按钮的id属性的<label>元素：

```
<input type="checkbox" name="color" value="#F00" id="red">
<label for="red">红色</label>
```

按钮组

由于每个单选按钮都会影响组内的其他按钮，通常会为整组单选按钮提供一个标签或上下文说明。

为了为整组提供标签，单选按钮应包含在一个<fieldset>元素中，且该元素内应有一个<legend>元素。

示例：

```
<fieldset>
  <legend>主题颜色：</legend>
  <p>
    <input type="radio" name="color" id="red" value="#F00">
    <label for="red">红色</label>
  </p>
  <p>
    <input type="radio" name="color" id="green" value="#0F0">
    <label for="green">绿色</label>
  </p>
```

The **checked** attribute specifies the initial state of a checkbox or radio button. This is a boolean attribute and may be omitted.

Each of these are valid, equivalent ways to define a checked radio button:

```
<input checked>
<input checked="">
<input checked="checked">
<input checked="ChEcKeD">
```

The absence of the checked attribute is the only valid syntax for an unchecked button:

```
<input type="radio">
<input type="checkbox">
```

When resetting a **<form>**, checkboxes and radio buttons revert to the state of their checked attribute.

Accessibility
Labels

To give context to the buttons and show users what each button is for, each of them should have a label. This can be done using a **<label>** element to wrap the button. Also, this makes the label clickable, so you select the corresponding button.

Example:

```
<label>
  <input type="radio" name="color" value="#F00">
  Red
</label>
```

or with a **<label>** element with a for attribute set to the id attribute of the button:

```
<input type="checkbox" name="color" value="#F00" id="red">
<label for="red">Red</label>
```

Button Groups

Since each radio button affects the others in the group, it is common to provide a label or context for the entire group of radio buttons.

To provide a label for the entire group, the radio buttons should be included in a **<fieldset>** element with a **<legend>** element within it.

Example:

```
<fieldset>
  <legend>Theme color:</legend>
  <p>
    <input type="radio" name="color" id="red" value="#F00">
    <label for="red">Red</label>
  </p>
  <p>
    <input type="radio" name="color" id="green" value="#0F0">
    <label for="green">Green</label>
  </p>
```

```
<p>
  <input type="radio" name="color" id="blue" value="#00F">
  <label for="blue">蓝色</label>
</p>
</fieldset>
```

复选框也可以用类似的方式分组，使用fieldset和legend来标识相关复选框的组。然而，请记住复选框不应该使用相同的名称，因为它们不是互斥的。这样做会导致表单为同一个键提交多个值，而不是所有服务器端语言都以相同方式处理这种情况（未定义行为）。每个复选框应有唯一的名称，或者使用一组方括号（[]）来表示表单应为该键提交一个值数组。

选择哪种方法应取决于你计划如何处理客户端或服务端的表单数据。你还应保持legend简短，因为某些浏览器和屏幕阅读器的组合会在fieldset中每个输入字段前读取legend。

第17.3节：输入验证

HTML输入验证由浏览器根据输入元素上的特殊属性自动完成。它可以部分或完全替代JavaScript输入验证。这种验证可以被用户通过特制的HTTP请求绕过，因此它不能替代服务器端输入验证。验证仅在尝试提交表单时发生，因此所有受限输入必须在表单内才能进行验证（除非你使用JavaScript）。请注意，禁用或只读的输入不会触发验证。

一些较新的输入类型（如email、url、tel、date等）会自动验证，无需你自己设置验证约束。

版本 ≥ 5
必填

使用required属性表示该字段必须填写才能通过验证。

```
<input required>
```

最小/最大长度

使用minlength和maxlength属性来指示长度要求。大多数浏览器会阻止用户输入超过max个字符，从而在用户尝试提交之前防止其输入无效内容。

```
<input minlength="3">
<input maxlength="15">
<input minlength="3" maxlength="15">
```

指定范围

使用min和max属性来限制用户在类型为number或range的输入框中输入的数字范围

```
分数：<input type="number" size="6" name="marks" min="0" max="100" />
科目反馈：<input type="range" size="2" name="feedback" min="1" max="5" />
```

版本 ≥ 5
匹配模式

为了更精确的控制，使用pattern属性指定必须匹配的正则表达式以通过验证。你还可以指定一个title，当字段未通过验证时，该标题会包含在验证消息中。

```
<p>
  <input type="radio" name="color" id="blue" value="#00F">
  <label for="blue">Blue</label>
</p>
</fieldset>
```

Checkboxes can also be grouped in a similar fashion, with a fieldset and legend identifying the group of related checkboxes. However, keep in mind that checkboxes should *not* share the same name because they are not mutually exclusive. Doing this will result in the form submitting multiple values for the same key and not all server-side languages handle this in the same way (undefined behavior). Each checkbox should either have a unique name, or use a set of square brackets ([]) to indicate that the form should submit an array of values for that key. Which method you choose should depend on how you plan to handle the form data client-side or server-side. You should also keep the legend short, since some combinations of browsers and screen readers read the legend before each input field in the fieldset.

Section 17.3: Input Validation

HTML input validation is done automatically by the browser based on special attributes on the input element. It could partially or completely replace JavaScript input validation. This kind of validation can be circumvented by the user via specially crafted HTTP requests, so it does not replace server-side input validation. The validation only occurs when attempting to submit the form, so all restricted inputs must be inside a form in order for validation to occur (unless you're using JavaScript). Keep in mind that inputs which are disabled or read-only will not trigger validation.

Some newer input types (like email, url, tel, date and many more) are automatically validated and do not require your own validation constraints.

Version ≥ 5
Required

Use the `required` attribute to indicate that a field must be completed in order to pass validation.

```
<input required>
```

Minimum / Maximum Length

Use the minlength and `maxlength` attributes to indicate length requirements. Most browsers will prevent the user from typing more than *max* characters into the box, preventing them from making their entry invalid even before they attempt submission.

```
<input minlength="3">
<input maxlength="15">
<input minlength="3" maxlength="15">
```

Specifying a range

Use min and `max` attributes to restrict the range of numbers a user can input into an input of type number or range

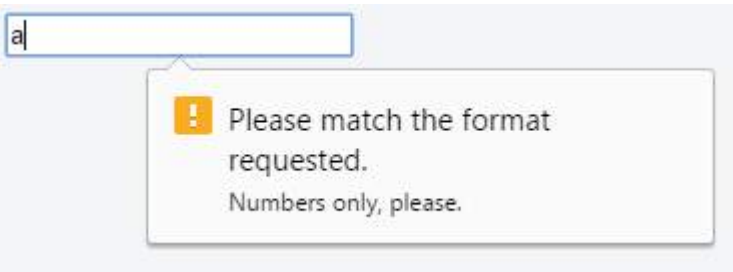
```
Marks: <input type="number" size="6" name="marks" min="0" max="100" />
Subject Feedback: <input type="range" size="2" name="feedback" min="1" max="5" />
```

Version ≥ 5
Match a Pattern

For more control, use the `pattern` attribute to specify any regular expression that must be matched in order to pass validation. You can also specify a `title`, which is included in the validation message if the field doesn't pass.


```
<input pattern="\d*" title="仅限数字，请输入。">
```

以下是Google Chrome 51版本在尝试提交该字段中无效值时显示的消息：



并非所有浏览器都会显示无效模式的提示，尽管大多数常用的现代浏览器都已完全支持。

请查看CanIUse上的最新支持情况并据此实现。

版本 ≥ 5
接受的文件类型

对于类型为file的输入字段，可以只接受某些类型的文件，例如视频、图片、音频、特定的文件扩展名或某些媒体类型。例如：

```
<input type="file" accept="image/*" title="只允许图片">
```

可以用逗号指定多个值，例如：

```
<input type="file" accept="image/*,.rar,application/zip">
```

注意：向form元素添加novalidate属性或向提交按钮添加formnovalidate属性，会阻止对表单元素的验证。例如：

```
<form>
  <input type="text" name="name" required>
  <input type="email" name="email" required>
  <input pattern="\d*" name="number" required>

  <input type="submit" value="发布"> <!-- 表单将被验证 -->
  <input type="submit" value="保存" formnovalidate> <!-- 表单将不被验证 -->
</form>
```

该表单包含一些字段，这些字段是“发布”草稿时必须填的，但“保存”草稿时并非必填。

第17.4节：颜色

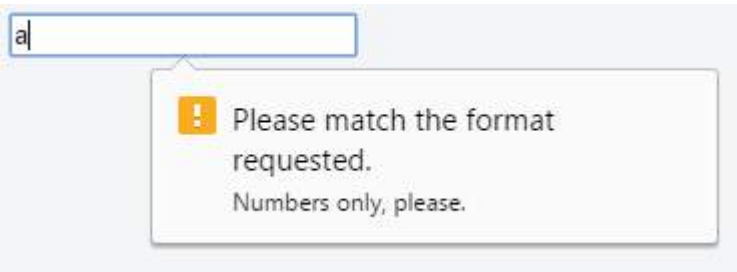
版本 ≥ 5
<input type="color" name="favcolor" value="#ff0000">

在支持的浏览器中，type属性值为color的输入元素会创建一个类似按钮的控件，其颜色等于color属性的值（如果未指定值或值为无效的十六进制格式，则默认为黑色）。



```
<input pattern="\d*" title="Numbers only, please.">
```

Here's the message shown in Google Chrome version 51 when attempting to submit the form with an invalid value inside this field:



Not all browsers display a message for invalid patterns, although there is full support among most used modern browsers.

Check the latest support on [CanIUse](#) and implement accordingly.

Version ≥ 5
Accept File Type

For input fields of type file, it is possible to accept only certain types of files, such as videos, images, audios, specific file extensions, or certain [media types](#). For example:

```
<input type="file" accept="image/*" title="Only images are allowed">
```

Multiple values can be specified with a comma, e.g.:

```
<input type="file" accept="image/*,.rar,application/zip">
```

Note: Adding novalidate attribute to the form element or formnovalidate attribute to the submit button, prevents validation on form elements. For example:

```
<form>
  <input type="text" name="name" required>
  <input type="email" name="email" required>
  <input pattern="\d*" name="number" required>

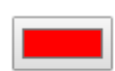
  <input type="submit" value="Publish"> <!-- form will be validated -->
  <input type="submit" value="Save" formnovalidate> <!-- form will NOT be validated -->
</form>
```

The form has fields that are required for "publishing" the draft but aren't required for "saving" the draft.

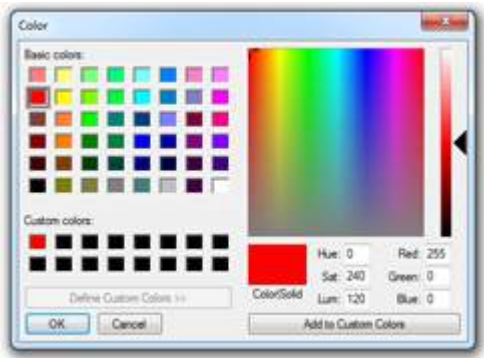
Section 17.4: Color

Version ≥ 5
<input type="color" name="favcolor" value="#ff0000">

In supporting browsers, the input element with a type attribute whose value is color creates a button-like control, with a color equal to the value of color attribute (defaults to black if value is not specified or is an invalid hexadecimal format).



点击此按钮会打开操作系统的颜色选择器，允许用户选择颜色。



对于不支持此输入类型的浏览器，回退为普通的输入框，type属性值为text。

第17.5节：密码

```
<input type="password" name="password">
```

type属性值为password的输入元素创建一个单行文本框，类似于type属性值为text的输入框，但用户输入时文本不会显示出来。

```
<input type="password" name="password" placeholder="Password">
```

占位符文本以纯文本形式显示，当用户开始输入时会自动被覆盖。

注意：一些浏览器和系统会修改密码字段的默认行为，短时间内显示最近输入的字符，如下所示：

第17.6节：文件

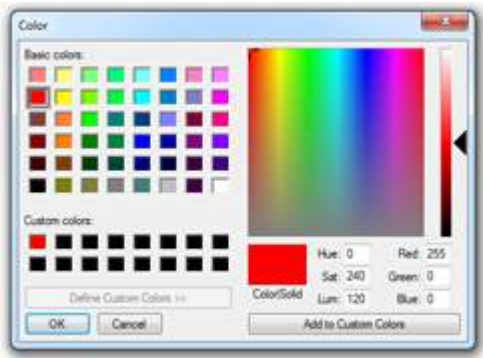
```
<input type="file" name="fileSubmission">
```

文件输入允许用户从本地文件系统选择文件以供当前页面使用。如果与form元素结合使用，可以允许用户将文件上传到服务器（更多信息请参见上传文件）。

下面的示例允许用户使用file输入从其文件系统选择文件，并将该文件上传到服务器上的名为upload_file.php的脚本。

```
<form action="upload_file.php" method="post" enctype="multipart/form-data">
  选择要上传的文件：
  <input type="file" name="fileSubmission" id="fileSubmission">
  <input type="submit" value="上传您的文件" name="submit">
</form>
```

Clicking this button opens the operating system's color widget, which allows user to select a color.



Fallback for browsers which do not support this input type is a regular input type=text.

Section 17.5: Password

```
<input type="password" name="password">
```

The input element with a type attribute whose value is password creates a single-line text field similar to the input type=text, except that text is not displayed as the user enters it.

```
<input type="password" name="password" placeholder="Password">
```

Placeholder text is shown in plain text and is overwritten automatically when a user starts typing.

Note: Some browsers and systems modify the default behavior of the password field to also display the most recently typed character for a short duration, like so:

Section 17.6: File

```
<input type="file" name="fileSubmission">
```

File inputs allow users to select a file from their local filesystem for use with the current page. If used in conjunction with a form element, they can be used to allow users to upload files to a server (for more info see Uploading Files).

The following example allows users to use the file input to select a file from their filesystem and upload that file to a script on the server named upload_file.php.

```
<form action="upload_file.php" method="post" enctype="multipart/form-data">
  Select file to upload:
  <input type="file" name="fileSubmission" id="fileSubmission">
  <input type="submit" value="Upload your file" name="submit">
</form>
```

多个文件

添加multiple属性后，用户将能够选择多个文件：

```
<input type="file" name="fileSubmission" id="fileSubmission" multiple>
```

接受文件

Accept 属性指定用户可以选择的文件类型。例如 .png、.gif、.jpeg。

```
<input type="file" name="fileSubmission" accept="image/x-png,image/gif,image/jpeg" />
```

第17.7节：按钮

```
<input type="button" value="按钮文本">
```

按钮可用于触发页面上的操作，而无需提交表单。你也可以使用 <button> 元素，如果你需要一个更容易样式化或包含其他元素的按钮：

```
<button type="button">按钮文本</button>
```

按钮通常与“onclick”事件一起使用：

```
<input type="button" onclick="alert('hello world!')" value="点击我">
```

或者

```
<button type="button" onclick="alert('hello world!')">点击我</button>
```

属性
[name]

按钮的name，随表单数据一起提交。

[type]

按钮的type。

可能的取值有：

submit ：按钮将表单数据提交到服务器。如果未指定该属性，或者该属性被动态更改为空或无效值，则默认使用此值。

reset ：按钮将所有控件重置为初始值。

button ：按钮没有默认行为。它可以关联客户端脚本，绑定到元素的事件，当事件发生时触发。

menu ：按钮打开通过其指定元素定义的弹出菜单。

[value]

按钮的初始值。

版本 ≥ 5

Multiple files

Adding the multiple attribute the user will be able to select **more than one** file:

```
<input type="file" name="fileSubmission" id="fileSubmission" multiple>
```

Accept Files

Accept attribute specifies the types of files that user can select. E.g. .png, .gif, .jpeg.

```
<input type="file" name="fileSubmission" accept="image/x-png,image/gif,image/jpeg" />
```

Section 17.7: Button

```
<input type="button" value="Button Text">
```

Buttons can be used for triggering actions to occur on the page, without submitting the form. You can also use the <button> element if you require a button that can be more easily styled or contain other elements:

```
<button type="button">Button Text</button>
```

Buttons are typically used with an "onclick" event:

```
<input type="button" onclick="alert('hello world!')" value="Click Me">
```

or

```
<button type="button" onclick="alert('hello world!')">Click Me</button>
```

Attributes
[name]

The name of the button, which is submitted with the form data.

[type]

The type of the button.

Possible values are:

submit : The button submits the form data to the server. This is the default if the attribute is not specified, or if the attribute is dynamically changed to an empty or invalid value.

reset : The button resets all the controls to their initial values.

button : The button has no default behavior. It can have client-side scripts associated with the element's events, which are triggered when the events occur.

menu : The button opens a popup menu defined via its designated element.

[value]

The initial value of the button.

Version ≥ 5

提交按钮的额外属性	
属性	描述
form	指定按钮所属表单的ID。 如果未指定，则按钮属于其祖先表单元素（如果存在）。
formaction	指定使用此按钮提交表单时，表单数据发送的位置。
formenctype	指定使用此按钮提交表单时，表单数据应如何编码。 只能与formmethod="post"一起使用。
formmethod	指定使用的HTTP方法（POST或GET） 当使用此按钮发送表单数据时。
formnovalidate	指定提交时不对表单数据进行验证。
formtarget	指定使用此按钮提交表单后，响应显示的位置。

第17.8节：提交

```
<input type="submit" value="提交">
```

提交输入创建一个按钮，点击时提交其所在的表单。

如果需要一个更易于样式化或包含其他元素的提交按钮，也可以使用<button>元素：

```
<button type="submit">
   提交
</button>
```

第17.9节：复位

```
<input type="reset" value="重置">
```

类型为reset的输入创建一个按钮，点击时会将其所在表单中的所有输入重置为默认状态。

- 输入字段中的文本将被重置为空白或其默认值（使用value属性指定）。
- 选择菜单中的任何选项将被取消选择，除非它们具有selected属性。
- 所有复选框和单选框将被取消选择，除非它们具有checked属性。

注意：重置按钮必须位于<form>元素内或通过form属性附加到该元素，才能生效。该按钮只会重置此表单内的元素。

第17.10节：隐藏

```
<input type="hidden" name="inputName" value="inputValue">
```

隐藏输入对用户不可见，但其值在表单提交时仍会发送到服务器。

第17.11节：电话

```
<input type="tel" value="+8400000000">
```

Extra Attributes for Submit Buttons	
Attribute	Description
form	Specifies the ID of the form the button belongs to. If none is specified, it will belong to its ancestor form element (if one exists).
formaction	Specifies where to send the form-data when the form is submitted using this button.
formenctype	Specifies how the form-data should be encoded when submitting it to the server using this button. Can only be used with formmethod="post".
formmethod	Specifies the HTTP method to use (POST or GET) when sending form-data using this button.
formnovalidate	Specifies that the form-data should not be validated on submission.
formtarget	Specifies where to display the response that is received after submitting the form using this button.

Section 17.8: Submit

```
<input type="submit" value="Submit">
```

A submit input creates a button which submits the form it is inside when clicked.

You can also use the <button> element if you require a submit button that can be more easily styled or contain other elements:

```
<button type="submit">
   Submit
</button>
```

Section 17.9: Reset

```
<input type="reset" value="Reset">
```

An input of type reset creates a button which, when clicked, resets all inputs in the form it is contained in to their default state.

- Text in an input field will be reset to blank or its default value (specified using the value attribute).
- Any option(s) in a selection menu will be deselected unless they have the selected attribute.
- All checkboxes and radio boxes will be deselected unless they have the checked attribute.

Note: A reset button must be inside or attached to (via the form attribute) a <form> element in order to have any effect. The button will only reset the elements within this form.

Section 17.10: Hidden

```
<input type="hidden" name="inputName" value="inputValue">
```

A hidden input won't be visible to the user, but its value will be sent to the server when the form is submitted nonetheless.

Section 17.11: Tel

```
<input type="tel" value="+8400000000">
```

带有 type 属性且值为 tel 的输入元素表示用于输入电话号码的单行纯文本编辑控件。

第17.12节：电子邮件

版本 ≥ 5

带有 type 属性且值为 email 的 <input> 用于应包含电子邮件地址的输入字段。

```
<form>
  <label>电子邮件：<label>
  <input type="email" name="email">
</form>
```

根据浏览器支持情况，提交时电子邮件地址可以自动验证。

第17.13节：数字

版本 ≥ 5

```
<input type="number" value="0" name="quantity">
```

带有 type 属性且值为 number 的输入元素表示用于将元素的值设置为表示数字的字符串的精确控件。

请注意，该字段不保证输入的是正确的数字。它仅允许所有可用于任何实数的符号，例如用户可以输入类似 e1e-0 的值。

第17.14节：范围

版本 ≥ 5

```
<input type="range" min="" max="" step="" />
```

用于输入数值的控件，其精确数值不重要。

属性	描述	默认值
最小值	范围的最小值	0
最大值	范围的最大值	100
步长	每次递增的数值。1	

第17.15节：搜索

版本 ≥ 5

输入类型search用于文本搜索。在大多数浏览器中，它会在文本输入框旁添加放大镜图标

```
<input type="search" name="googlesearch">
```

第17.16节：图像

```
<input type="image" src="img.png" alt="image_name" height="50px" width="50px"/>
```

一张图片。您必须使用src属性来定义图片的来源，使用alt属性来定义

The input element with a type attribute whose value is tel represents a one-line plain-text edit control for entering a telephone number.

Section 17.12: Email

Version ≥ 5

The <input type="email"> is used for input fields that should contain an e-mail address.

```
<form>
  <label>E-mail: <label>
  <input type="email" name="email">
</form>
```

E-mail address can be automatically validated when submitted depending on browser support.

Section 17.13: Number

Version ≥ 5

```
<input type="number" value="0" name="quantity">
```

The Input element with a type attribute whose value is number represents a precise control for setting the element’s value to a string representing a number.

Please note that this field does not guarantee to have a correct number. It just allows all the symbols which could be used in any real number, for example user will be able to enter value like e1e-, 0.

Section 17.14: Range

Version ≥ 5

```
<input type="range" min="" max="" step="" />
```

A control for entering a number whose exact value is not important.

Attribute	Description	Default value
min	Minimum value for range	0
max	Maximum value for range	100
step	Amount to increase by on each increment.	1

Section 17.15: Search

Version ≥ 5

Input type search is used for textual search. It will add magnifier symbol next to space for text on most browsers

```
<input type="search" name="googlesearch">
```

Section 17.16: Image

```
<input type="image" src="img.png" alt="image_name" height="50px" width="50px"/>
```

An Image. You must use the src attribute to define the source of the image and the alt attribute to define

替代文本。您可以使用height和width属性以像素为单位定义图片的大小。

第17.17节：周

版本 ≥ 5

```
<input type="week" />
```

根据浏览器支持情况，将显示一个控件，用于输入周-年份编号和周编号，且无时区。

第17.18节：网址

版本 ≥ 5

```
<input type="url" name="Homepage">
```

此字段用于应包含网址的输入框。

根据浏览器支持情况，url 字段在提交时可以自动验证。

一些智能手机识别url类型，并在键盘上添加“.com”以匹配url输入。

第17.19节：日期时间-本地

版本 ≥ 5

```
<input type="datetime-local" />
```

根据浏览器支持情况，屏幕上会弹出日期和时间选择器，供您选择日期和时间。

第17.20节：月份

版本 ≥ 5

```
<input type="month" />
```

根据浏览器支持情况，会显示一个控件以选择月份。

第17.21节：时间

版本 ≥ 5

```
<input type="time" />
```

时间输入标记该元素接受表示时间的字符串。格式定义在RFC 3339中，应为部分时间，例如

```
19:04:39
08:20:39.04
```

目前，所有版本的Edge、Chrome、Opera以及Android版Chrome均支持type="time"。较新版本的Android浏览器，特别是4.4及以上版本也支持。iOS版Safari提供部分支持，不支持min、max和step属性。

alternative text. You can use the height and width attributes to define the size of the image in pixels.

Section 17.17: Week

Version ≥ 5

```
<input type="week" />
```

Dependent on browser support, a control will show for entering a week-year number and a week number with no time zone.

Section 17.18: Url

Version ≥ 5

```
<input type="url" name="Homepage">
```

This is used for input fields that should contain a URL address.

Depending on browser support, the url field can be automatically validated when submitted.

Some smartphones recognize the url type, and adds ".com" to the keyboard to match url input.

Section 17.19: DateTime-Local

Version ≥ 5

```
<input type="datetime-local" />
```

Dependent on browser support, a date and time picker will pop up on screen for you to choose a date and time.

Section 17.20: Month

Version ≥ 5

```
<input type="month" />
```

Dependent on browser support, a control will show to pick the month.

Section 17.21: Time

Version ≥ 5

```
<input type="time" />
```

The time input marks this element as accepting a string representing a time. The format is defined in [RFC 3339](#) and should be a partial-time such as

```
19:04:39
08:20:39.04
```

Currently, all versions of Edge, Chrome, Opera, and Chrome for Android support type="time". The newer versions of Android Browser, specifically 4.4 and up support it. Safari for iOS offers partial support, not supporting min, max, and step attributes.

第17.22节：日期时间（全局）

type属性值为"datetime"的输入元素表示一个控件，用于设置元素'的值为表示**全球日期和时间（含时区信息）**的字符串。

```
<fieldset>
  <p><label>会议时间：<input type=datetime name="meeting.start"></label>
</fieldset>
```

允许的属性：

- 全局属性
- name
- disabled
- form
- 类型
- autocomplete
- autofocus
- list
- 最小值和最大值
- 步长（浮点数）
- 只读
- 必填值

第17.23节：日期

版本 ≥ 5

```
<input type="date" />
```

屏幕上会弹出一个日期选择器供您选择日期。Firefox或Internet Explorer不支持此功能。

Section 17.22: DateTime (Global)

The input element with a type attribute whose value is "**datetime**" represents a control for setting the element's value to a string representing a **global date and time (with timezone information)**.

```
<fieldset>
  <p><label>Meeting time: <input type=datetime name="meeting.start"></label>
</fieldset>
```

Permitted attributes:

- global attributes
- name
- disabled
- form
- type
- autocomplete
- autofocus
- list
- min & max
- step (float)
- readonly
- required value

Section 17.23: Date

Version ≥ 5

```
<input type="date" />
```

A date picker will pop up on screen for you to choose a date. This is not supported in Firefox or Internet Explorer.

第18章：表单

属性	描述
accept-charset	指定用于表单提交的字符编码。
action	指定提交表单时发送表单数据的位置。
autocomplete	指定表单是否应启用自动完成功能。
enctype	指定提交表单数据到服务器时应如何编码（仅适用于 method="post"）。
method	指定发送表单数据时使用的 HTTP 方法（POST 或 GET）。
name	指定表单的名称。
novalidate	指定提交表单时不进行验证。
target	指定提交表单后接收响应的显示位置。

为了分组输入元素并提交数据，HTML 使用 form 元素来封装输入和提交元素。这些表单负责以指定的方法将数据发送到由服务器或处理程序处理的页面。
本主题解释并演示了使用HTML表单收集和提交输入数据的方法。

第18.1节：提交

动作属性

动作属性定义了提交表单时要执行的操作，通常会调用一个脚本来收集提交的信息并进行处理。如果留空，则会将数据发送到同一个文件

```
<form action="action.php">
```

方法属性

方法属性用于定义表单的HTTP方法，通常是GET或POST。

```
<form action="action.php" method="get">
<form action="action.php" method="post">
```

GET方法主要用于获取数据，例如通过ID或名称接收帖子，或提交搜索查询。GET方法会将表单数据附加到动作属性指定的URL后面。

```
www.example.com/action.php?firstname=Mickey&lastname=Mouse
```

POST方法用于向脚本提交数据。POST方法不会将表单数据附加到动作URL，而是通过请求体发送。

要正确提交表单中的数据，必须指定一个名称属性 name。
例如，我们发送该字段的值，并将其名称设置为lastname：

```
<input type="text" name="lastname" value="Mouse">
```

更多属性

```
<form action="action.php" method="post" target="_blank" accept-charset="UTF-8"
enctype="application/x-www-form-urlencoded" autocomplete="off" novalidate>

<!-- 表单元素 -->
```

Chapter 18: Forms

Attribute	Description
accept-charset	Specifies the character encodings that are to be used for the form submission.
action	Specifies where to send the form-data when a form is submitted.
autocomplete	Specifies whether a form should have autocomplete on or off.
enctype	Specifies how the form-data should be encoded when submitting it to the server (only for method="post").
method	Specifies the HTTP method to use when sending form-data (POST or GET).
name	Specifies the name of a form.
novalidate	Specifies that the form should not be validated when submitted.
target	Specifies where to display the response that is received after submitting the form.

In order to group input elements and submit data, HTML uses a form element to encapsulate input and submission elements. These forms handle sending the data in the specified method to a page handled by a server or handler. This topic explains and demonstrates the usage of HTML forms in collecting and submitting input data.

Section 18.1: Submitting

The Action Attribute

The action attribute defines the action to be performed when the form is submitted, which usually leads to a script that collects the information submitted and works with it. if you leave it blank, it will send it to the same file

```
<form action="action.php">
```

The Method Attribute

The method attribute is used to define the HTTP method of the form which is either GET or POST.

```
<form action="action.php" method="get">
<form action="action.php" method="post">
```

The GET method is mostly used to get data, for example to receive a post by its ID or name, or to submit a search query. The GET method will append the form data to the URL specified in the action attribute.

```
www.example.com/action.php?firstname=Mickey&lastname=Mouse
```

The POST method is used when submitting data to a script. The POST method does not append the form data to the action URL but sends using the request body.

To submit the data from the form correctly, a name attribute name must be specified.
As an example let's send the value of the field and set its name to lastname:

```
<input type="text" name="lastname" value="Mouse">
```

More attributes

```
<form action="action.php" method="post" target="_blank" accept-charset="UTF-8"
enctype="application/x-www-form-urlencoded" autocomplete="off" novalidate>

<!-- form elements -->
```

</form>

第18.2节：form标签中的target属性

target属性指定一个名称或关键字，用于指示提交表单后接收的响应显示的位置。

target属性定义了浏览上下文（例如标签页、窗口或内联框架）的名称或关键字。

来自带有 target 属性的标签：

<form target="_blank">

属性值

值	描述
_blank	响应显示在新窗口或标签页中
_self	响应显示在相同框架中（默认）
_parent	响应显示在父框架中
_top	响应显示在整个窗口主体中
framename	响应显示在命名的iframe中

注意：target属性在HTML 4.01中被弃用。target属性在HTML5中被支持。

在HTML5中不支持框架和框架集，因此_parent、_top和framename值现在主要用于iframe。

第18.3节：上传文件

通过将form标签的enctype属性设置为multipart/form-data，可以上传/提交图像和文件到服务器。enctype指定提交到服务器时表单数据的编码方式。

示例

```
<form method="post" enctype="multipart/form-data" action="upload.php">
  <input type="file" name="pic" />
<input type="submit" value="上传" />
</form>
```

第18.4节：分组几个输入字段

在设计表单时，您可能希望将几个输入字段分组，以帮助组织表单布局。这可以通过使用标签来完成。以下是使用它的示例。

对于每个字段集，您可以使用标签 LEGEND TEXT 设置该集合的标题

示例

```
<form>
  <fieldset>
    <legend>第1个字段集：</legend>
    字段一：<br>
    <input type="text"><br>
```

</form>

Section 18.2: Target attribute in form tag

The target attribute specifies a name or a keyword that indicates where to display the response that is received after submitting the form.

The target attribute defines a name of, or keyword for, a browsing context (e.g. tab, window, or inline frame).

From Tag with a target attribute:

<form target="_blank">

Attribute Values

Value	Description
_blank	The response is displayed in a new window or tab
_self	The response is displayed in the same frame (this is default)
_parent	The response is displayed in the parent frame
_top	The response is displayed in the full body of the window
framename	The response is displayed in a named iframe

Note: The target attribute was **deprecated** in **HTML 4.01**. The target attribute is **supported** in **HTML5**.

Frames and framesets are not supported in **HTML5**, so the **_parent, _top and framename values are now mostly used with iframes**.

Section 18.3: Uploading Files

Images and files can be uploaded/submitted to server by setting enctype attribute of form tag to multipart/form-data. enctype specifies how form data would be encoded while submitting to the server.

Example

```
<form method="post" enctype="multipart/form-data" action="upload.php">
  <input type="file" name="pic" />
  <input type="submit" value="Upload" />
</form>
```

Section 18.4: Grouping a few input fields

While designing a form, you might like to group a few input fields into a group to help organise the form layout. This can be done by using the tag . Here is an example for using it.

For each fieldset, you can set a legend for the set using the tag LEGEND TEXT

Example

```
<form>
  <fieldset>
    <legend>1st field set:</legend>
    Field one:<br>
    <input type="text"><br>
```

```
字段二：<br>
    <input type="text"><br>
</fieldset><br>
<fieldset>
    <legend>第2个字段集：</legend>
    字段三：<br>
    <input type="text"><br>
    字段四：<br>
    <input type="text"><br>
</fieldset><br>
<input type="submit" value="提交">
</form>
```

结果

1st field set:

Field one:

Field two:

2nd field set:

Field three:

Field four:

Submit

浏览器支持

Chrome、IE、Edge、FireFox、Safari 和 Opera 的最新版也支持该标签

```
Field two:<br>
    <input type="text"><br>
</fieldset><br>
<fieldset>
    <legend>2nd field set:</legend>
    Field three:<br>
    <input type="text"><br>
    Field four:<br>
    <input type="text"><br>
</fieldset><br>
<input type="submit" value="Submit">
</form>
```

Result

1st field set:

Field one:

Field two:

2nd field set:

Field three:

Field four:

Submit

Browser Support

Chrome, IE, Edge, FireFox, Safari and Opera's latest versions also supports the tag

第19章：Div 元素

HTML 中的 div 元素是一个容器元素，用于封装其他元素，可以用来分组和分隔网页的部分内容。div 本身不具备固有的语义，但在网页设计中是一个强大的工具。本章节介绍了 div 元素的用途和应用。

第19.1节：基本用法

<div> 元素本身通常没有特定的语义意义，仅表示一个区块，通常用于在HTML文档中对其他元素进行分组和封装，并将其与其他内容组分开。因此，每个 <div> 最好通过其内容来描述。

```
<div>
  <p>你好！这是一个段落。</p>
</div>
```

div 元素通常是块级元素，意味着它分隔HTML文档中的一个块，并占据页面的最大宽度。浏览器通常具有以下默认CSS规则：

```
div {
  display: block;
}
```

万维网联盟（The World Wide Web Consortium，W3C）强烈建议将div元素视为最后的选择元素，仅在没有其他合适元素时使用。使用更合适的元素代替div元素，有助于提高读者的可访问性和作者的维护便利性。

例如，博客文章应使用<article>标记，章节使用<section>，页面的导航辅助使用<nav>，表单控件组使用<fieldset>。

div元素在样式目的上或用于包裹一个部分内多个段落（这些段落都需要以类似方式注释）时非常有用。

第19.2节：嵌套

将多个<div>放置在另一个<div>内是常见的做法。这通常被称为“嵌套”元素，允许进一步将元素划分为子部分，或帮助开发者进行CSS样式设计。

<div class="outer-div">用于将两个<div class="inner-div">元素组合在一起；每个元素中都包含一个<p>元素。

```
<div class="outer-div">
  <div class="inner-div">
    <p>这是一个段落</p>
  </div>
  <div class="inner-div">
    <p>这是另一个段落</p>
  </div>
</div>
```

这将产生以下结果（为清晰起见应用了CSS样式）：

Chapter 19: Div Element

The div element in HTML is a container element that encapsulates other elements and can be used to group and separate parts of a webpage. A div by itself does not inherently represent anything but is a powerful tool in web design. This topic covers the purpose and applications of the div element.

Section 19.1: Basic usage

The <div> element usually has no specific semantic meaning by itself, simply representing a division, and is typically used for grouping and encapsulating other elements within an HTML document and separating those from other groups of content. As such, each <div> is best described by its contents.

```
<div>
  <p>Hello! This is a paragraph.</p>
</div>
```

The div element is typically a block-level element, meaning that it separates a block of an HTML document and occupying the maximum width of the page. Browsers typically have the following default CSS rule:

```
div {
  display: block;
}
```

It's strongly encouraged by the **The World Wide Web Consortium (W3C)** to view the div element as an element of last resort, for when no other element is suitable. The use of more appropriate elements instead of the div element leads to better accessibility for readers and easier maintainability for authors.

For example, a blog post would be marked up using <article>, a chapter using <section>, a page's navigation aids using <nav>, and a group of form controls using <fieldset>.

div elements can be useful for stylistic purposes or to wrap multiple paragraphs within a section that are all to be annotated in a similar way.

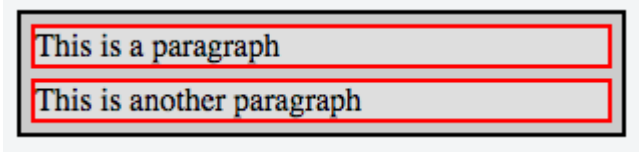
Section 19.2: Nesting

It is a common practice to place multiple <div> inside another <div>. This is usually referred to as "nesting" elements and allows for further dividing elements into subsections or aid developers with CSS styling.

The <div class="outer-div"> is used to group together two <div class="inner-div"> elements; each containing a <p> element.

```
<div class="outer-div">
  <div class="inner-div">
    <p>This is a paragraph</p>
  </div>
  <div class="inner-div">
    <p>This is another paragraph</p>
  </div>
</div>
```

This will yield the following result (CSS styles applied for clarity):



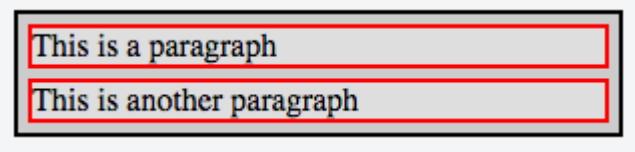
嵌套内联和块级元素在嵌套元素时应注意，存在内联元素和块级元素。块级元素“在背景中添加换行”，这意味着其他嵌套元素会自动显示在下一行，内联元素默认可以并排放置。

避免深层<div>嵌套

深层且频繁使用的嵌套容器布局显示出不良的编码风格。

圆角或类似功能经常会生成这样的HTML代码。对于大多数最新一代浏览器，已有CSS3对应方案。尽量减少HTML元素的使用，以提高内容与标签的比例，减少页面加载时间，从而提升搜索引擎排名。

div section 元素不应嵌套超过6层。



Nesting inline and block elements While nesting elements you should keep in mind, that there are inline and block elements. while block elements "add a line break in the background", what means, other nested elements are shown in the next line automatically, inline elements can be positioned next to each other by default

Avoid deep <div> nesting

A deep and oftenly used nested container layouts shows a bad coding style.

Rounded corners or some similar functions often create such an HTML code. For most of the last generation browsers there are CSS3 counterparts. Try to use as little as possible HTML elements to increase the content to tag ratio and reduce page load, resulting in a better ranking in search engines.

div section Element should be not nested deeper than 6 layers.

第20章：分区元素

第20.1节：导航元素

<nav> 元素主要用于包含网站导航块的部分，这可以包括指向网页其他部分的链接（例如目录锚点）或完全指向其他页面的链接。

内联项目

下面将显示一组内联超链接。

```
<nav>
  <a href="https://google.com">Google</a>
  <a href="https://www.yahoo.com">Yahoo!</a>
  <a href="https://www.bing.com">Bing</a>
</nav>
```

需要时使用列表项

如果内容表示一组项目，请使用列表项来展示，以提升用户体验。

注意role="navigation"，下面会详细说明。

```
<nav role="navigation">
  <ul>
    <li><a href="https://google.com">Google</a></li>
    <li><a href="https://www.yahoo.com">Yahoo!</a></li>
    <li><a href="https://www.bing.com">Bing</a></li>
  </ul>
</nav>
```

避免不必要的使用

<footer> 元素可能包含指向网站其他部分（常见问题、条款等）的链接列表。在这种情况下，单独使用 footer 元素就足够了，您不需要在<footer>中再用<nav>元素包裹链接。

```
<!-- 在<footer>中不需要<nav> -->
<footer>
  <nav>
    <a href="#">...</a>
  </nav>
</footer>

<!-- 单独使用 footer 就足够了 -->
<footer>
  <a href="#">...</a>
</footer>
```

注释：

- <main> 元素的后代不允许出现在 <nav>

建议为 <nav> 元素添加 role="navigation" ARIA 角色，以帮助不支持 HTML5 的用户代理，同时也为支持 HTML5 的用户代理提供更多上下文。

Chapter 20: Sectioning Elements

Section 20.1: Nav Element

The <nav> element is primarily intended to be used for sections that contain **main navigation blocks** for the website, this can include links to other parts of the web page (*e.g. anchors for a table of contents*) or other pages entirely.

Inline items

The following will display an inline set of hyperlinks.

```
<nav>
  <a href="https://google.com">Google</a>
  <a href="https://www.yahoo.com">Yahoo!</a>
  <a href="https://www.bing.com">Bing</a>
</nav>
```

Use list items when needed

If the content represents a list of items, use a list item to show this and enhance the user experience.

Note the role="navigation", *more on this below*.

```
<nav role="navigation">
  <ul>
    <li><a href="https://google.com">Google</a></li>
    <li><a href="https://www.yahoo.com">Yahoo!</a></li>
    <li><a href="https://www.bing.com">Bing</a></li>
  </ul>
</nav>
```

Avoid unnecessary usage

<footer> elements may have a list of links to other parts of the site (FAQ, T&C, etc.). The footer element alone is sufficient in this case, you don't *need* to further wrap your links with a <nav> element in the <footer>.

```
<!-- the <nav> is not required in the <footer> -->
<footer>
  <nav>
    <a href="#">...</a>
  </nav>
</footer>

<!-- The footer alone is sufficient -->
<footer>
  <a href="#">...</a>
</footer>
```

Notes:

- <main> [element](#) descendants are not allowed within a <nav>

Adding a role="navigation" [ARIA role](#) to the <nav> element is advised to aid user agents that don't support HTML5 and also to provide more context for those that do.

```
<nav role="navigation"><!-- ... --></nav>
```

屏幕阅读器：（允许盲人或视力障碍用户浏览网站的软件）像屏幕阅读器这样的用户代理会

根据其需求不同而对 <nav> 元素进行不同的解释。

- 它可能会在渲染页面时赋予 <nav> 元素更高的优先级
 - 它可能会延迟该元素的渲染
 - 它可能会以特定方式调整页面以满足用户需求
- 例如：为视力障碍者将 <nav> 元素内的文本链接放大。

[点击此处阅读 <nav> 元素的官方 HTML5 规范](#)

第20.2节：文章元素

<article> 元素包含 独立内容，如文章、博客帖子、用户评论或交互式小部件，这些内容可以在页面上下文之外分发，例如通过 RSS。

- 当文章元素嵌套时，内层文章节点的内容应与外层文章元素相关。

一个包含多个帖子（section）和评论（article）的博客可能看起来像这样。

```
<section>
  <!-- 每个独立的博客帖子是一个 <article> -->
  <article>
    <header>
      <h1>博客帖子</h1>
      <time datetime="2016-03-13">2016年3月13日</time>
    </header>

    <p>article 元素表示一个独立的文章或文档。</p>
    <p>section 元素表示内容的分组。</p>

    <section>
      <h2>评论<small>关于“博客文章”</small></h2>

      <!-- 相关评论也是一篇独立的文章 -->
      <article id="user-comment-1">
        <p>非常棒！</p>
        <footer><p>...</p><time>...</time></footer>
      </article>
    </section>
  </article>

  <!-- ./重复：<article> -->

</section>

<!-- 与博客或文章无关的内容应放在 section 外部。 -->
<footer>
  <p>此内容应与博客无关。</p>
</footer>
```

避免不必要的使用

```
<nav role="navigation"><!-- ... --></nav>
```

Screen Readers: (software that allows blind or visually impaired users to navigate the site)

User agents like screen readers will interpret the <nav> element differently depending on their requirements.

- It could give the <nav> element a higher priority when rendering the page
 - It could delay the rendering of the element
 - It could adapt the page in a specific way to tailor for the user's needs
- example: make the text links within the <nav> elements larger for someone who's visually impaired.

[Click here to read the official HTML5 Specification for the <nav> element](#)

Section 20.2: Article Element

The <article> element contains **self-contained content** like articles, blog posts, user comments or an interactive widget that could be distributed outside the context of the page, for example by RSS.

- When article elements are nested, the contents of the inner article node should be related to the outer article element.

A blog (section) with multiple posts (article), and comments (article) might look something like this.

```
<section>
  <!-- Each individual blog post is an <article> -->
  <article>
    <header>
      <h1>Blog Post</h1>
      <time datetime="2016-03-13">13th March 2016</time>
    </header>

    <p>The article element represents a self contained article or document.</p>
    <p>The section element represents a grouping of content.</p>

    <section>
      <h2>Comments <small>relating to "Blog Post"</small></h2>

      <!-- Related comment is also a self-contained article -->
      <article id="user-comment-1">
        <p>Excellent!</p>
        <footer><p>...</p><time>...</time></footer>
      </article>
    </section>
  </article>

  <!-- ./repeat: <article> -->

</section>

<!-- Content unrelated to the blog or posts should be outside the section. -->
<footer>
  <p>This content should be unrelated to the blog.</p>
</footer>
```

Avoid unnecessary usage

当页面的主要内容（不包括页眉、页脚、导航栏等）仅是一组元素时，可以省略<article>，改用<main>元素。

```
<article>
  <p>这没有道理，这篇文章没有真正的“上下文”。</p>
</article>
```

相反，应使用<main>元素替换article元素，以表明这是该页面的main内容。

```
<main>
  <p>我是主要内容，不需要属于article元素。</p>
</main>
```

如果使用其他元素，确保指定<main>的ARIA角色，以便在多种设备和非HTML5浏览器中正确解释和渲染。

```
<section role="main">
  <p>该部分是本页面的主要内容。</p>
</section>
```

注释：

- <main>元素的子元素不允许出现在<article>内

[点击此处阅读<article>元素的官方HTML5规范](#)

第20.3节：Main元素

<main>元素包含网页的**主要内容**。该内容是页面独有的，不应在网站的其他地方出现。重复内容如页眉、页脚、导航、标志等应放置在该元素之外。

- 页面上最多只能使用一次<main>元素。
- 不得将<main>元素作为article、aside、footer、header或<nav>元素的后代包含。

在以下示例中，我们展示了一个**单个博客帖子**（以及相关信息，如引用和评论）。

```
<body>
  <header>
    <nav>...</nav>
  </header>

  <main>
    <h1>个人博客帖子</h1>
    <p>帖子的介绍。</p>

    <article>
      <h2>引用</h2>
      <p>...</p>
    </article>

    <article>
```

When the main content of the page (excluding headers, footers, navigation bars, etc.) is simply one group of elements. You can omit the <article> in favour of the <main> element.

```
<article>
  <p>This doesn't make sense, this article has no real `context`.</p>
</article>
```

Instead, replace the article with a <main> element to indicate this is the main content for this page.

```
<main>
  <p>I'm the main content, I don't need to belong to an article.</p>
</main>
```

If you use another element, ensure you specify the <main> ARIA role for correct interpretation and rendering across multiple devices and non HTML5 browsers.

```
<section role="main">
  <p>This section is the main content of this page.</p>
</section>
```

Notes:

- <main> element descendants are not allowed within a <article>

[Click here to read the official HTML5 Specification for the <article> element](#)

Section 20.3: Main Element

The <main> element contains the **main content** for your web page. This content is unique to the individual page, and should not appear elsewhere on the site. Repeating content like headers, footers, navigation, logos, etc., is placed outside the element.

- The <main> element should only ever be used at most **once** on a single page.
- The <main> element must not be included as a descendant of an article, aside, footer, header or nav element.

In the following example, we're displaying a **single blog post** (and related information like references and comments).

```
<body>
  <header>
    <nav>...</nav>
  </header>

  <main>
    <h1>Individual Blog Post</h1>
    <p>An introduction for the post.</p>

    <article>
      <h2>References</h2>
      <p>...</p>
    </article>

    <article>
```

```

        <h2>评论</h2> ...
    </article>
</main>

    <footer>...</footer>
</body>

```

- 博客文章包含在<main>元素中，以表明这是该页面的主要内容（因此，在整个网站中是唯一的）。

- 标签<header>和<footer>是<main>元素的兄弟元素。

注释：

HTML5规范将<main>元素视为一个**分组**元素，而非分区元素。

- [ARIA角色属性：main（默认），presentation](#)

建议向其他元素添加role="main"ARIA角色属性，以便支持HTML5的用户代理更好地识别主内容，同时为支持HTML5的用户代理提供更多上下文。

<main>元素默认具有main角色，因此无需额外提供该属性。

[点击此处阅读<main>元素的官方HTML5规范](#)

第20.4节：头部元素

<header>元素表示其最近的祖先分区内容或分区根元素的引导内容。一个<header>通常包含一组引导或导航辅助内容。

注意：<header>元素不是分区内容；它不会引入新的分区。

示例：

```

<header>
  <p>欢迎来到...</p>
  <h1>虚空战争！</h1>
</header>

```

在这个例子中，<article> 有一个 <header>。

```

<article>
  <header>
    <h1>Flexbox：权威指南</h1>
  </header>
  <p>关于Flexbox的指南本应在这里，但结果是韦斯也不是Flexbox专家。</p>
</article>

```

[W3C建议的推荐标准](#)

```

    <h2>Comments</h2> ...
  </article>
</main>

  <footer>...</footer>
</body>

```

- The blog post is contained within the <main> element to indicate this is the main content for this page (and therefore, unique across the website).

- The <header> and <footer> tags are *siblings* to the <main> element.

Notes:

The HTML5 specification recognizes the <main> element as a **grouping** element, and not a *sectioning* element.

- [ARIA role attributes: main \(default\), presentation](#)

Adding a role="main" ARIA role attribute to **other elements** intended to be used as main content is advised to aid user agents that don't support HTML5 and also to provide more context for those that do.

The <main> element by default has the main role, and so does not need to be provided.

[Click here to read the official HTML5 Specification for the <main> element](#)

Section 20.4: Header Element

The <header> element represents introductory content for its nearest ancestor sectioning content or sectioning root element. A <header> typically contains a group of introductory or navigational aids.

Note: The header element is not sectioning content; it doesn't introduce a new section.

Examples:

```

<header>
  <p>Welcome to...</p>
  <h1>Voidwars!</h1>
</header>

```

In this example, the <article> has a <header>.

```

<article>
  <header>
    <h1>Flexbox：The definitive guide</h1>
  </header>
  <p>The guide about Flexbox was supposed to be here, but it turned out Wes wasn't a Flexbox expert either.</p>
</article>

```

[W3C Proposed Recommendation](#)

第20.5节：页脚元素

<footer> 元素包含页面的页脚部分。

这是一个包含 p 段落标签的 <footer> 元素示例。

```
<footer>
  <p>版权所有</p>
</footer>
```

第20.6节：区块元素

<section> 元素表示一个通用的部分，用于按主题对内容进行分组。每个部分通常都应该能够通过作为 section 子元素的标题元素来识别。

- 你可以在<article>元素内使用<section>元素，反之亦然。
- 每个部分都应该有一个主题（一个标识该区域的标题元素）
- 不要将<section>元素用作通用的样式“容器”。如果你需要一个容器来应用样式，请改用<div>。

在下面的示例中，我们展示了一篇单篇博客文章，包含多个章节，每个章节都是一个部分（一组主题相关的内容，可以通过每个部分中的标题元素来识别）。

```
<article>
  <header>
    <h2>博客文章</h2>
  </header>
  <p>文章简介。</p>
  <section>
    <h3>第一章</h3>
    <p>...</p>
  </section>
  <section>
    <h3>第二章</h3>
    <p>...</p>
  </section>
  <section>
    <h3>评论</h3> ...
  </section>
</article>
```

注释：

开发者应在适合对元素内容进行联合发布时使用article元素。

[点击此处阅读<main>元素的官方HTML5规范](#)

Section 20.5: Footer Element

The <footer> element contains the footer part of the page.

Here is an example for <footer> element that contain p paragraph tag.

```
<footer>
  <p>All rights reserved</p>
</footer>
```

Section 20.6: Section Element

The <section> element represents a generic section to thematically group content. Every section, typically, should be able to be identified with a heading element as a child of the section.

- You can use the <section> element within an <article> and vice-versa.
- Every section should have a theme (a heading element identifying this region)
- Don't use the <section> element as a general styling 'container'. If you need a container to apply styling, use a <div> instead.

In the following example, we're displaying a **single blog post** with multiple chapters each chapter is a section (a set of thematically grouped content, which can be identified by the heading elements in each section).

```
<article>
  <header>
    <h2>Blog Post</h2>
  </header>
  <p>An introduction for the post.</p>
  <section>
    <h3>Chapter 1</h3>
    <p>...</p>
  </section>
  <section>
    <h3>Chapter 2</h3>
    <p>...</p>
  </section>
  <section>
    <h3>Comments</h3> ...
  </section>
</article>
```

Notes:

Developers should use the **article** element when it makes sense to syndicate the contents of the element.

[Click here to read the official HTML5 Specification for the <main> element](#)

第21章：导航栏

第21.1节：基本导航栏

导航栏本质上是一组链接列表，因此使用ul和li元素来包裹导航链接。

```
<ul>
  <li><a href="#">首页</a></li>
  <li><a href="#">关于</a></li>
  <li><a href="#">联系</a></li>
</ul>
```

第21.2节：HTML5导航栏

要使用HTML5的nav元素制作导航栏，请将链接包裹在nav标签内。

```
<nav>
  <a href="#">首页</a>
  <a href="#">关于</a>
  <a href="#">联系</a>
</nav>
```

Chapter 21: Navigation Bars

Section 21.1: Basic Navigation Bar

Navigation bars are essentially a list of links, so the ul and li elements are used to encase navigation links.

```
<ul>
  <li><a href="#">Home</a></li>
  <li><a href="#">About</a></li>
  <li><a href="#">Contact</a></li>
</ul>
```

Section 21.2: HTML5 Navigation Bar

To make a navigation bar using the HTML5 nav element, encase the links within the nav tag.

```
<nav>
  <a href="#">Home</a>
  <a href="#">About</a>
  <a href="#">Contact</a>
</nav>
```

第22章：标签元素

属性	描述
为	引用目标ID元素。例如：for="surname"
form	HTML5, [已废弃] 引用包含目标元素的表单。标签元素预期位于<form>元素内。如果提供了form="someFormId", 则允许你将标签放置在文档中的任意位置。

第22.1节：关于标签

<label>元素用于引用表单操作元素。
在用户界面范围内，它用于简化对类型为radio或checkbox等元素的目标/选择。

<label>作为包装器

它可以包含所需的操作元素

```
<label>
  <input type="checkbox" name="Cats">
  我喜欢猫！
</label>
```

(点击文本时，目标input将切换其状态/值)

<label>作为引用

使用for属性时，不必将控件元素作为label的子元素，但for的值必须与其ID匹配

```
<input id="cats" type="checkbox" name="Cats">
<label for="cats" >我喜欢猫！</label>
```

注意

不要在<label>元素内使用多个控件元素

第22.2节：基本用法

带标签的简单表单...

```
<form action="/login" method="POST">

  <label for="username">用户名：</label>
  <input id="username" type="text" name="username" />

  <label for="pass">密码：</label>
  <input id="pass" type="password" name="pass" />

  <input type="submit" name="submit" />

</form>
```

版本 ≥ 5

```
<form id="my-form" action="/login" method="POST">

  <input id="username" type="text" name="username" />

</form>
```

Chapter 22: Label Element

Attributes	Description
for	Reference to the target ID Element. I.e: for="surname"
form	HTML5, [Obsolete] Reference to the form containing the Target Element. Label elements are expected within a <form> Element. If the form="someFormId" is provided this allows you to place the Label anywhere in the document.

Section 22.1: About Label

The <label> element is used to reference a form action element.
In the scope of **User Interface** it's used to ease the target / selection of elements like Type radio or checkbox.

<label> as wrapper

It can enclose the desired action element

```
<label>
  <input type="checkbox" name="Cats">
  I like Cats!
</label>
```

(Clicking on the text the target input will toggle it's state / value)

<label> as reference

Using the for attribute you don't have to place the control element as descendant of label - but the for value must match it's ID

```
<input id="cats" type="checkbox" name="Cats">
<label for="cats" >I like Cats!</label>
```

Note

Don't use more than one Control Element within a <label> element

Section 22.2: Basic Use

Simple form with labels...

```
<form action="/login" method="POST">

  <label for="username">Username:</label>
  <input id="username" type="text" name="username" />

  <label for="pass">Password:</label>
  <input id="pass" type="password" name="pass" />

  <input type="submit" name="submit" />

</form>
```

Version ≥ 5

```
<form id="my-form" action="/login" method="POST">

  <input id="username" type="text" name="username" />

</form>
```

```
<label for="pass">密码:</label>
<input id="pass" type="password" name="pass" />

<input type="submit" name="submit" />

</form>

<label for="username" form="my-form">用户名:</label>
```

```
<label for="pass">Password:</label>
<input id="pass" type="password" name="pass" />

<input type="submit" name="submit" />

</form>

<label for="username" form="my-form">Username:</label>
```

第23章：输出元素

属性	描述
全局	适用于任何HTML5元素的属性。有关这些属性的全面文档， 请参见：MDN全局属性
name	表示输出名称的字符串。作为表单元素，output可以通过其名称使用document.forms属性进行引用。此属性也用于表单提交时收集值。
为	一个以空格分隔的表单元素id列表（例如 <inputs >对应的值是"inp1"），该输出用于显示这些元素的计算结果。<="" id="inp1" td=""></inputs>
form	表示与输出关联的<form>的字符串。如果输出实际上位于<form>之外，此属性将确保输出仍属于该<form>，并受该<form>的收集和提交控制。

第23.1节：使用 For 和 Form 属性的输出元素

以下演示展示了<output>元素如何使用[for]和[form]属性。请记住，<output>需要JavaScript才能正常工作。内联JavaScript通常用于表单中，正如本例所示。虽然<input>元素的类型是type="number"，但它们的value不是数字，而是文本。因此，如果需要对value进行计算，必须使用诸如parseInt()、parseFloat()、Number()等方法将每个value转换为数字。

实时演示

```
<!--form1将在“input”事件时收集in1和in2的值。-->
<!--out1的值将是in1和in2值的和。-->

<form id="form1" name="form1" oninput="out1.value = parseInt(in1.value, 10) + parseInt(in2.value, 10)">

  <fieldset>

    <legend>输出示例</legend>

    <input type="number" id="in1" name="in1" value="0">
    <br/>
    +
    <input type="number" id="in2" name="in2" value="0">

  </fieldset>

</form>

<!--[for] 属性使 out1 显示 in1 和 in2 的计算结果。-->
<!--[form] 属性指定 form1 作为 out1 的表单所有者，即使它不是后代元素。-->

<output name="out1" for="in1 in2" form="form1">0</output>
```

第23.2节：带属性的输出元素

```
<output name="out1" form="form1" for="inp1 inp2"></output>
```

Chapter 23: Output Element

Attribute	Description
Global	Attributes that are available to any HTML5 element. For comprehensive documentation of these attributes see: MDN Global Attributes
name	A string representing the name of an output. As a form element, output can be referenced by it's name using the document.forms property. This attribute is also used for collecting values on a form submit.
for	A space separated list of form element ids (e.g. <inputs id="inp1"> for value is "inp1") that the output is meant to display calculations for.
form	A string representing the <form> that is associated to the output. If the output is actually outside the <form>, this attribute will ensure that the output still belongs to the <form> and subject to collections and submits of said <form>.

Section 23.1: Output Element Using For and Form Attributes

The following demo features an <output> element's use of the [for] and [form] attributes. Keep in mind, <output> needs JavaScript in order to function. Inline JavaScript is commonly used in forms as this example demonstrates. Although the <input> elements are type="number", their values are not numbers, they are text. So if you require the values to be calculated, you must convert each value into a number using methods such as: parseInt(), parseFloat(), Number(), etc.

Live Demo

```
<!--form1 will collect the values of in1 and in2 on 'input' event.-->
<!--out1 value will be the sum of in1 and in2 values.-->

<form id="form1" name="form1" oninput="out1.value = parseInt(in1.value, 10) + parseInt(in2.value, 10)">

  <fieldset>

    <legend>Output Example</legend>

    <input type="number" id="in1" name="in1" value="0">
    <br/>
    +
    <input type="number" id="in2" name="in2" value="0">

  </fieldset>

</form>

<!--[for] attribute enables out1 to display calculations for in1 and in2.-->
<!--[form] attribute designates form1 as the form owner of out1 even if it isn't a descendant.-->

<output name="out1" for="in1 in2" form="form1">0</output>
```

Section 23.2: Output Element with Attributes

```
<output name="out1" form="form1" for="inp1 inp2"></output>
```


第24章：空元素

并非所有HTML标签结构相同。虽然大多数元素需要开始标签、结束标签和内容，但有些元素——称为空元素——只需要开始标签，因为它们本身不包含任何元素。本节介绍并演示HTML中空元素的正确用法。

第24.1节：空元素

HTML 4.01/XHTML 1.0 Strict 包含以下空元素：

- area - 图像中可点击的定义区域
- base - 指定所有链接的基础URL
- br - 换行符
- col - 表格中的列 [已废弃]
- hr - 水平线（分隔线）
- img - 图片
- input - 用户输入数据的字段
- link - 将外部资源链接到文档
- meta - 提供关于文档的信息
- param - 定义插件的参数

HTML 5 标准包含了之前列表中所有未废弃的标签，并且

- command - 表示用户可以调用的命令 [过时]
- keygen - 便于生成用于网页证书的公钥 [已废弃]
- source - 指定 picture、audio 和 video 元素的媒体来源

下面的示例不包含空元素（void elements）：

```
<div>
  <a href="http://stackoverflow.com/">
    <h3>点击这里访问 <i>Stack Overflow!</i></h3>
  </a>
  <button onclick="alert('Hello!');">打个招呼！</button>
  <p>我最喜欢的语言是 <b>HTML</b>。以下是我喜欢的其他语言：</p>
  <ol>
    <li>CSS</li>
    <li>JavaScript</li>
    <li>PHP</li>
  </ol>
</div>
```

注意每个元素都有一个开始标签、一个结束标签，以及位于开始标签和结束标签之间的文本或其他元素。然而，空元素标签在下面的示例中展示：

```

<br>
<hr>
<input type="number" placeholder="Enter your favorite number">
```

除了 img 标签之外，所有这些空元素只有开始标签。img 标签与其他标签不同，在开始标签的右尖括号前有一个自闭合的斜杠 /。最佳实践是在斜杠前留一个空格。

Chapter 24: Void Elements

Not all HTML tags are of the same structure. While most elements require an opening tag, a closing tag, and contents, some elements - known as void elements - only require an opening tag as they themselves do not contain any elements. This topic explains and demonstrates the proper usage of void elements in HTML

Section 24.1: Void elements

HTML 4.01/XHTML 1.0 Strict includes the following void elements:

- area - clickable, defined area in an image
- base - specifies a base URL from which all links base
- br - line break
- col - column in a table [deprecated]
- hr - horizontal rule (line)
- img - image
- input - field where users enter data
- link - links an external resource to the document
- meta - provides information about the document
- param - defines parameters for plugins

HTML 5 standards include all non-deprecated tags from the previous list and

- command - represents a command users can invoke [obsolete]
- keygen - facilitates public key generation for web certificates [deprecated]
- source - specifies media sources for picture, audio, and video elements

The example below does **not** include void elements:

```
<div>
  <a href="http://stackoverflow.com/">
    <h3>Click here to visit <i>Stack Overflow!</i></h3>
  </a>
  <button onclick="alert('Hello!');">Say Hello!</button>
  <p>My favorite language is <b>HTML</b>. Here are my others:</p>
  <ol>
    <li>CSS</li>
    <li>JavaScript</li>
    <li>PHP</li>
  </ol>
</div>
```

Notice how every element has an opening tag, a closing tag, and text or other elements inside the opening and closing tags. Void tags however, are shown in the example below:

```

<br>
<hr>
<input type="number" placeholder="Enter your favorite number">
```

With the exception of the img tag, all of these void elements have only an opening tag. The img tag, unlike any other tag, has a self closing / before the greater than sign of the opening tag. It is best practice to have a space before the slash.

第25章：媒体元素

属性	详情
宽度	设置元素的宽度（像素）。
高度	设置元素的高度（像素）。
<source>	定义音频或视频文件的资源
track	定义媒体元素的文本轨道
controls	显示控件
autoplay	自动开始播放媒体
loop	循环播放媒体
静音	无声播放媒体
poster	指定在视频加载前显示的图片

第25.1节：音频

HTML5为在网页上嵌入音频文件提供了新的标准。

你可以使用<audio>元素将音频文件嵌入页面：

```
<audio controls>
  <source src="file.mp3" type="audio/mpeg">
  您的浏览器不支持音频元素。
</audio>
```

第25.2节：视频

您也可以使用<video>元素将视频嵌入网页：

```
<video width="500" height="700" controls>
  <source src="video.mp4" type="video/mp4">
  您的浏览器不支持video标签。
</video>
```

第25.3节：使用`<video>`和`<audio>`元素显示音频/视频内容

使用HTML的<audio>元素在文档中嵌入视频/音频内容。视频/音频元素包含一个或多个视频/音频源。要指定源，可以使用src属性或<source>元素；浏览器将选择最合适的一个。

音频标签示例：

```
<!-- 简单视频示例 -->
<video src="videofile.webm" autoplay poster="posterimage.jpg">
  抱歉，您的浏览器不支持嵌入视频，
  但别担心，你可以<a href="videofile.webm">下载它</a>
  并用你喜欢的视频播放器观看！
</video>

<!-- 带字幕的视频 -->
<video src="foo.webm">
  <track kind="subtitles" src="foo.en.vtt" srclang="en" label="英语">
  <track kind="subtitles" src="foo.sv.vtt" srclang="sv" label="瑞典语">
```

Chapter 25: Media Elements

Attribute	Details
width	Sets the element's width in pixels.
height	Sets the element's height in pixels.
<source>	Defines resources of the audio or video files
track	Defines the text track for media elements
controls	Displays controls
autoplay	Automatically start playing the media
loop	Plays the media in a repeated cycle
muted	Plays the media without sound
poster	Assigns an image to display until a video is loaded

Section 25.1: Audio

HTML5 provides a new standard for embedding an audio file on a web page.

You can embed an audio file to a page using the <audio> element:

```
<audio controls>
  <source src="file.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio>
```

Section 25.2: Video

You can embed also a video to a webpage using the <video> element:

```
<video width="500" height="700" controls>
  <source src="video.mp4" type="video/mp4">
  Your browser does not support the video tag.
</video>
```

Section 25.3: Using `<video>` and `<audio>` element to display audio/video content

Use the HTML or <audio> element to embed video/audio content in a document. The video/audio element contains one or more video/audio sources. To specify a source, use either the src attribute or the <source> element; the browser will choose the most suitable one.

Audio tag example:

```
<!-- Simple video example -->
<video src="videofile.webm" autoplay poster="posterimage.jpg">
  Sorry, your browser doesn't support embedded videos,
  but don't worry, you can <a href="videofile.webm">download it</a>
  and watch it with your favorite video player!
</video>

<!-- Video with subtitles -->
<video src="foo.webm">
  <track kind="subtitles" src="foo.en.vtt" srclang="en" label="English">
  <track kind="subtitles" src="foo.sv.vtt" srclang="sv" label="Svenska">
```

```
</video>
<!-- 简单视频示例 -->
<video width="480" controls poster="https://archive.org/download/WebmVp8Vorbis/webmvp8.gif" >
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8.webm" type="video/webm">
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8_512kb.mp4" type="video/mp4">
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8.ogv" type="video/ogg">
  你的浏览器不支持HTML5视频标签。
</video>
```

音频标签示例：

```
<!-- 简单音频播放 -->
<audio src="http://developer.mozilla.org/@api/deki/files/2926/=AudioTest_(1).ogg" autoplay>
  你的浏览器不支持<code>audio</code>元素。
</audio>

<!-- 带字幕的音频播放 -->
<audio src="foo.ogg">
  <track kind="captions" src="foo.en.vtt" srclang="en" label="English">
  <track kind="captions" src="foo.sv.vtt" srclang="sv" label="Svenska">
</audio>
```

第25.4节：视频标题或背景

添加一个自动循环播放且无控制和声音的视频。非常适合用作视频标题或背景。

```
<video width="1280" height="720" autoplay muted loop poster="video.jpg" id="videobg">
  <source src="video.mp4" type="video/mp4">
  <source src="video.webm" type="video/webm">
  <source src="video.ogg" type="video/ogg">
</video>
```

此CSS提供了视频无法加载时的备用方案。请注意，建议使用视频的第一帧作为poster视频video.jpg。

```
#videobg {
background: url(video.jpg) no-repeat;
background-size: cover;
}
```

```
</video>
<!-- Simple video example -->
<video width="480" controls poster="https://archive.org/download/WebmVp8Vorbis/webmvp8.gif" >
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8.webm" type="video/webm">
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8_512kb.mp4" type="video/mp4">
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8.ogv" type="video/ogg">
  Your browser doesn't support HTML5 video tag.
</video>
```

Audio tag example:

```
<!-- Simple audio playback -->
<audio src="http://developer.mozilla.org/@api/deki/files/2926/=AudioTest_(1).ogg" autoplay>
  Your browser does not support the <code>audio</code> element.
</audio>

<!-- Audio playback with captions -->
<audio src="foo.ogg">
  <track kind="captions" src="foo.en.vtt" srclang="en" label="English">
  <track kind="captions" src="foo.sv.vtt" srclang="sv" label="Svenska">
</audio>
```

Section 25.4: Video header or background

Adding a video that will autoplay on a loop and has no controls or sound. Perfect for a video header or background.

```
<video width="1280" height="720" autoplay muted loop poster="video.jpg" id="videobg">
  <source src="video.mp4" type="video/mp4">
  <source src="video.webm" type="video/webm">
  <source src="video.ogg" type="video/ogg">
</video>
```

This CSS provides a fallback if the video cannot be loaded. Note that is it recommended to use the first frame of the video as the poster video.jpg.

```
#videobg {
background: url(video.jpg) no-repeat;
background-size: cover;
}
```

第26章：进度元素

参数	值
最大值	任务总共需要多少工作量
value	已经完成了多少工作量
位置	此属性返回<progress>元素的当前位置
标签	此属性返回<progress>元素的标签列表（如果有的话）

第26.1节：进度

<progress>元素是HTML5中新引入的，用于表示任务的进度

```
<progress value="22" max="100"></progress>
```

这会创建一个填充了22%的进度条

第26.2节：更改进度条的颜色

进度条可以使用progress[value]选择器进行样式设置。

此示例为进度条设置了宽度为250px，高度为20px

```
progress[value] {
  width: 250px;
  height: 20px;
}
```

进度条的样式设置尤其困难。

Chrome / Safari / Opera

这些浏览器使用-webkit-appearance选择器来设置progress标签的样式。要覆盖此样式，我们可以重置appearance。

```
progress[value] {
  -webkit-appearance: none;
  appearance: none;
}
```

现在，我们可以为容器本身设置样式

```
progress[value]::-webkit-progress-bar {
  background-color: "green";
}
```

Firefox

Firefox 对进度条的样式处理略有不同。我们必须使用这些样式

```
progress[value] {
  -moz-appearance: none;
  appearance: none;
  border: none;
}

/* Firefox 也会渲染一个 border */
```

Chapter 26: Progress Element

Parameter	Value
max	How much work the task requires in total
value	How much of the work has been accomplished already
position	This attribute returns the current position of the <progress> element
labels	This attribute returns a list of <progress> element labels (if any)

Section 26.1: Progress

The <progress> element is new in HTML5 and is used to represent the progress of a task

```
<progress value="22" max="100"></progress>
```

This creates a bar filled 22%

Section 26.2: Changing the color of a progress bar

Progress bars can be styled with the progress[value] selector.

This example gives a progress bar a width of 250px and a height of 20px

```
progress[value] {
  width: 250px;
  height: 20px;
}
```

Progress bars can be especially difficult to style.

Chrome / Safari / Opera

These browsers use the -webkit-appearance selector to style the progress tag. To override this, we can reset the appearance.

```
progress[value] {
  -webkit-appearance: none;
  appearance: none;
}
```

Now, we can style the container itself

```
progress[value]::-webkit-progress-bar {
  background-color: "green";
}
```

Firefox

Firefox styles the progress bar a little differently. We have to use these styles

```
progress[value] {
  -moz-appearance: none;
  appearance: none;
  border: none;
}

/* Firefox also renders a border */
```

```
}
```

Internet Explorer

Internet Explorer 10 及以上版本支持 progress 元素。但它不支持 background-color 属性。你需要改用 color 属性。

```
progress[value] {
  -webkit-appearance: none;
  -moz-appearance: none;
  appearance: none;

  border: none; /* 移除 Firefox 的 border */

  width: 250px;
  height: 20px;

  color: blue;
}
```

第26.3节：HTML回退

对于不支持progress元素的浏览器，您可以使用此方法作为解决方案。

```
<progress max="100" value="20">
  <div class="progress-bar">
    <span style="width: 20%;">进度：20%</span>
  </div>
</progress>
```

支持progress标签的浏览器会忽略嵌套在其中的div。无法识别progress标签的旧版浏览器则会渲染div。

```
}
```

Internet Explorer

Internet Explorer 10+ supports the progress element. However, it does not support the background-color property. You'll need to use the color property instead.

```
progress[value] {
  -webkit-appearance: none;
  -moz-appearance: none;
  appearance: none;

  border: none; /* Remove border from Firefox */

  width: 250px;
  height: 20px;

  color: blue;
}
```

Section 26.3: HTML Fallback

For browsers that do not support the progress element, you can use this as a workaround.

```
<progress max="100" value="20">
  <div class="progress-bar">
    <span style="width: 20%;">Progress: 20%</span>
  </div>
</progress>
```

Browsers that support the progress tag will ignore the div nested inside. Legacy browsers which cannot identify the progress tag will render the div instead.

第27章：选择菜单控件

第27.1节：选择菜单

<select>元素生成一个下拉菜单，用户可以从选择一个选项。

```
<select name="">
  <option value="1">一</option>
  <option value="2">二</option>
  <option value="3">三</option>
  <option value="4">四</option>
</select>
```

更改大小

您可以使用大小属性更改选择菜单的大小。大小为0或1时，显示标准的下拉样式菜单。大于1的大小会将下拉菜单转换为一个显示多行的框，每行显示一个选项，并带有滚动条以便浏览可用选项。

```
<select name="" size="4"></select>
```

多选选择菜单

默认情况下，用户只能选择一个选项。添加multiple属性允许用户一次选择多个选项，并通过表单提交所有选中的选项。使用multiple属性会自动将下拉菜单转换为一个框，就像定义了大小一样。此时的默认大小由您使用的具体浏览器决定，且无法在允许多选的情况下将其改回下拉样式菜单。

```
<select name="" multiple></select>
```

使用multiple属性时，大小为0和1存在区别，而不使用该属性时则无区别。使用0时，浏览器将按其默认方式行为。使用1时，会明确将结果框的大小设置为仅一行高。

第27.2节：选项

选择菜单中的选项是用户将要选择的内容。选项的正常语法如下：

```
<option>某个选项</option>
```

但是，需要注意的是，<option> 元素内部的文本并不总是被使用，实际上它成为未指定属性的默认值。

控制选项实际外观和功能的属性是value和label。label表示将在下拉菜单中显示的文本（你看到并点击以选择的内容）。value表示随表单提交一起发送的文本。如果省略了这两个值中的任意一个，则会使用元素内部的文本作为值。因此，我们上面给出的示例可以“扩展”为如下形式：

```
<option label="Some Option" value="Some Option">
```

注意省略了内部文本和结束标签，这些实际上并不是构建菜单中选项所必需的。

Chapter 27: Selection Menu Controls

Section 27.1: Select Menu

The <select> element generates a drop-down menu from which the user can choose an option.

```
<select name=" ">
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
  <option value="4">Four</option>
</select>
```

Changing the Size

You can change the size of the selection menu with the size attribute. A size of 0 or 1 displays the standard drop-down style menu. A size greater than 1 will convert the drop-down into a box displaying that many lines, with one option per line and a scrollbar in order to scroll through the available options.

```
<select name=" " size="4"></select>
```

Multi-option Selection Menus

By default, users can only select a single option. Adding the multiple attribute allows users to select multiple options at once and submit all selected options with the form. Using the multiple attribute automatically converts the drop-down menu into a box as if it had a size defined. The default size when this occurs is determined by the specific browser you are using, and it is not possible to change it back to a drop-down style menu while allowing multiple selections.

```
<select name=" " multiple></select>
```

When using the multiple attribute, there is a difference between using 0 and 1 for the size, whereas no difference exists when not using the attribute. Using 0 will cause the browser to behave in whatever default manner it is programmed to do. Using 1 will explicitly set the size of the resulting box to only one row high.

Section 27.2: Options

The options inside a selection menu are what the user will be selection. The normal syntax for an option is as follows:

```
<option>Some Option</option>
```

However, it's important to note that the text inside the <option> element itself is not always used, and essentially becomes the default value for attributes which are not specified.

The attributes which control the actual appearance and function of the option are value and label. The label represents the text which will be displayed in the drop-down menu (what you're looking at and will click on to select it). The value represents the text which will be sent along with form submission. If either of these values is omitted, it uses the text inside the element as the value instead. So the example we gave above could be "expanded" to this:

```
<option label="Some Option" value="Some Option">
```

Note the omission of the inside text and end tag, which are not required to actually construct an option inside the

如果包含了它们，内部文本会被忽略，因为两个属性都已指定，文本不再需要。然而，你可能不会看到很多人这样写。最常见的写法是，value属性用于发送到服务器，内部文本最终成为label属性，如下所示：

```
<option value="option1">Some Option</option>
```

默认选择一个选项

你也可以通过为某个选项添加selected属性，指定该选项在菜单中默认被选中。默认情况下，如果菜单中没有指定选中项，渲染时会默认选中菜单中的第一个选项。如果有多个选项带有selected属性，则菜单中带有该属性的最后一个选项将被默认选中。

```
<option value="option1" selected>Some option</option>
```

如果你在多选菜单中使用该属性，则所有带有该属性的选项都会被默认选中，如果没有选项带有该属性，则没有选项被选中。

```
<select multiple>
  <option value="option1" selected>某个选项</option>
  <option value="option2" selected>某个选项</option>
</select>
```

第27.3节：选项组

您可以使用<optgroup>元素，将选项整齐地分组在选择菜单中，以便在长选项列表中提供更有结构的布局。

语法非常简单，只需使用带有label属性的元素来标识组的标题，并包含零个或多个应属于该组的选项。

```
<select name="">
  <option value="milk">牛奶</option>
  <optgroup label="水果">
    <option value="banana">香蕉</option>
    <option value="strawberry">草莓</option>
  </optgroup>
  <optgroup label="蔬菜" disabled>
    <option value="carrot">胡萝卜</option>
    <option value="zucchini">西葫芦</option>
  </optgroup>
</select>
```

使用选项组时，并非所有选项都必须包含在组内。此外，禁用一个选项组将禁用组内的所有选项，且无法手动重新启用禁用组内的单个选项。

第27.4节：数据列表

<datalist> 标签为 <input> 元素指定了一组预定义选项。它为 <input> 元素提供了“自动完成”功能。用户在输入时会看到一个下拉选项列表。

```
<input list="Languages">
```

menu. If they were included, the inside text would be ignored because both attributes are already specified and the text is not needed. However, you probably won't see a lot of people writing them this way. The most common way it's written is with a value that will be sent to the server, along with the inside text which eventually becomes the label attribute, like so:

```
<option value="option1">Some Option</option>
```

Selecting an option by default

You can also specify a certain option to be selected in the menu by default by attaching the selected attribute to it. By default, if no option is specified as selected in the menu, the first option in the menu will be selected when rendered. If more than one option has the selected attribute attached, then the last option present in the menu with the attribute will be the one selected by default.

```
<option value="option1" selected>Some option</option>
```

If you're using the attribute in a multi-option selection menu, then all the options with the attribute will be selected by default, and none will be selected if no options have the attribute.

```
<select multiple>
  <option value="option1" selected>Some option</option>
  <option value="option2" selected>Some option</option>
</select>
```

Section 27.3: Option Groups

You can neatly group your options within a selection menu in order to provide a more structured layout in a long list of options by using the <optgroup> element.

The syntax is very basic, by simply using the element with a label attribute to identify the title for the group, and containing zero or more options that should be within that group.

```
<select name="">
  <option value="milk">Milk</option>
  <optgroup label="Fruits">
    <option value="banana">Bananas</option>
    <option value="strawberry">Strawberries</option>
  </optgroup>
  <optgroup label="Vegetables" disabled>
    <option value="carrot">Carrots</option>
    <option value="zucchini">Zucchini</option>
  </optgroup>
</select>
```

When using option groups, not all options need to be contained within a group. As well, disabling an option group will disable all options within the group, and it is not possible to manually re-enable a single option within a disabled group.

Section 27.4: Datalist

The <datalist> tag specifies a list of pre-defined options for an <input> element. It provide an "autocomplete" feature on <input> elements. Users will see a drop-down list of options as they write.

```
<input list="Languages">
```

```
<datalist id="Languages">
  <option value="PHP">
  <option value="Perl">
  <option value="Python">
  <option value="Ruby">
  <option value="C+">
</datalist>
```

浏览器支持

Chrome	Edge	Mozilla	Safari	Opera
20.0	10.0	4.0	不支持	9.0

```
<datalist id="Languages">
  <option value="PHP">
  <option value="Perl">
  <option value="Python">
  <option value="Ruby">
  <option value="C+">
</datalist>
```

Browser Support

Chrome	Edge	Mozilla	Safari	Opera
20.0	10.0	4.0	Not Supported	9.0

第28章：嵌入

参数	详情
src	资源地址
类型	嵌入资源类型
宽度	水平尺寸
高度	垂直尺寸

第28.1节：基本用法

embed标签是HTML5中新引入的元素。该元素为外部（通常是非HTML）应用程序或交互内容提供了一个集成点。

```
<embed src="myflash.swf">
```

第28.2节：定义MIME类型

必须使用type属性定义MIME类型。

```
<embed type="video/mp4" src="video.mp4" width="640" height="480">
```

Chapter 28: Embed

Parameters	Details
src	Address of the resource
type	Type of embedded resource
width	Horizontal dimension
height	Vertical dimension

Section 28.1: Basic usage

The embed tag is new in HTML5. This element provides an integration point for an external (typically non-HTML) application or interactive content.

```
<embed src="myflash.swf">
```

Section 28.2: Defining the MIME type

The [MIME](#) type must be defined using the type attribute.

```
<embed type="video/mp4" src="video.mp4" width="640" height="480">
```

第29章：内联框架（IFrames）

属性	详情
name	设置元素的名称，用于配合 a 标签更改iframe的 src。
宽度	设置元素的宽度（像素）。
高度	设置元素的高度（像素）。
src	指定将在框架中显示的页面。
srcdoc	指定将在框架中显示的内容，前提是浏览器支持。内容必须是有效的HTML。
sandbox	设置后，iframe的内容将被视为来自唯一源，脚本、插件、表单和弹出窗口等功能将被禁用。可以通过添加以空格分隔的值列表选择性地放宽限制。有关可能的值，请参见备注中的表格。
allowfullscreen	是否允许iframe的内容使用requestFullscreen()

第29.1节：内联框架基础

“IFrame”一词指内联框架。它可以用来在你的页面中包含另一个页面。这将生成一个小框架，显示 base.html的确切内容。

```
<iframe src="base.html"></iframe>
```

第29.2节：沙箱

下面嵌入了一个启用所有限制的不受信任网页

```
<iframe sandbox src="http://example.com/"></iframe>
```

要允许页面运行脚本和提交表单，请在 sandbox属性中添加allow-scripts和allow-forms。

```
<iframe sandbox="allow-scripts allow-forms" src="http://example.com/"></iframe>
```

如果在与父网页相同的域中存在不受信任的内容（例如用户评论），可以使用iframe禁用脚本，同时仍允许父文档通过JavaScript与其内容交互。

```
<iframe sandbox="allow-same-origin allow-top-navigation" src="http://example.com/untrusted/comments/page2">
```

父文档可以添加事件监听器并调整IFrame大小以适应其内容。结合allow-top-navigation，可以使沙箱iframe看起来像是父文档的一部分。

此沙箱不能替代输入的清理，但可以作为纵深防御策略的一部分使用。

还要注意，攻击者可能通过说服用户直接访问 iframe 的源来绕过此沙箱。可以使用内容安全策略（Content Security Policy）HTTP 头来缓解此类攻击。

第29.3节：设置框架大小

可以使用width和height属性调整IFrame的大小，数值以像素为单位（HTML 4.01允许百分比值，但HTML 5仅允许使用CSS像素值）。

```
<iframe src="base.html" width="800" height="600"></iframe>
```

Chapter 29: IFrames

Attribute	Details
name	Sets the element's name, to be used with an a tag to change the iframe's src.
width	Sets the element's width in pixels.
height	Sets the element's height in pixels.
src	Specifies the page that will be displayed in the frame.
srcdoc	Specifies the content that will be displayed in the frame, assuming the browser supports it. The content must be valid HTML.
sandbox	When set, the contents of the iframe is treated as being from a unique origin and features including scripts, plugins, forms and popups will be disabled. Restrictions can be selectively relaxed by adding a space separated list of values. See the table in Remarks for possible values.
allowfullscreen	Whether to allow the iframe's contents to use requestFullscreen()

Section 29.1: Basics of an Inline Frame

The term "IFrame" means Inline Frame. It can be used to include another page in your page. This will yield a small frame which shows the exact contents of the base.html.

```
<iframe src="base.html"></iframe>
```

Section 29.2: Sandboxing

The following embeds an untrusted web page with all restrictions enabled

```
<iframe sandbox src="http://example.com/"></iframe>
```

To allow the page to run scripts and submit forms, add allow-scripts and allow-forms to the sandbox attribute.

```
<iframe sandbox="allow-scripts allow-forms" src="http://example.com/"></iframe>
```

If there is untrusted content (such as user comments) on the same domain as the parent web page, an iframe can be used to disable scripts while still allowing the parent document to interact with it's content using JavaScript.

```
<iframe sandbox="allow-same-origin allow-top-navigation" src="http://example.com/untrusted/comments/page2">
```

The parent document can add event listeners and resize the IFrame to fit its contents. This, along with allow-top-navigation, can make the sandboxed iframe appear to be part of parent document.

This sandbox is not a replacement for sanitizing input but can be used as part of a defense in depth strategy.

Also be aware that this sandbox can be subverted by an attacker convincing a user to visit the iframe's source directly. The Content Security Policy HTTP header can be used to mitigate this attack.

Section 29.3: Setting the Frame Size

The IFrame can be resized using the width and height attributes, where the values are represented in pixels (HTML 4.01 allowed percentage values, but HTML 5 only allows values in CSS pixels).

```
<iframe src="base.html" width="800" height="600"></iframe>
```


第29.4节：使用“srcdoc”属性

srcdoc属性可以用来（代替src属性）指定iframe的完整HTML文档内容。这将生成一个显示文本“IFrames are cool!”的IFrame。

```
<iframe srcdoc="<p>IFrames are cool!</p>"></iframe>
```

如果浏览器不支持srcdoc属性，IFrame将退回使用src属性，但如果浏览器同时支持且存在src和srcdoc属性，则以srcdoc为优先。

```
<iframe srcdoc="<p>IFrames are cool!</p>" src="base.html"></iframe>
```

在上述示例中，如果浏览器不支持srcdoc属性，则会显示base.html页面的内容。

第29.5节：在IFrame中使用锚点

通常，Iframe 内网页的更改是由 Iframe 内部发起的，例如点击 Iframe 内的链接。然而，也可以从 Iframe 外部更改 Iframe 的内容。你可以使用一个锚点标签，其 href 属性设置为所需的 URL，target 属性设置为 iframe 的 name 属性。

```
<iframe src="webpage.html" name="myIframe"></iframe>
<a href="different_webpage.html" target="myIframe">将 Iframe 内容更改为
different_webpage.html</a>
```

Section 29.4: Using the "srcdoc" Attribute

The srcdoc attribute can be used (instead of the src attribute) to specify the exact contents of the iframe as a whole HTML document. This will yield an IFrame with the text "IFrames are cool!"

```
<iframe srcdoc="<p>IFrames are cool!</p>"></iframe>
```

If the srcdoc attribute isn't supported by the browser, the IFrame will instead fall back to using the src attribute, but if both the src and srcdoc attributes are present and supported by the browser, srcdoc takes precedence.

```
<iframe srcdoc="<p>IFrames are cool!</p>" src="base.html"></iframe>
```

In the above example, if the browser does not support the srcdoc attribute, it will instead display the contents of the base.html page.

Section 29.5: Using Anchors with IFrames

Normally a change of webpage within an Iframe is initiated from within the Iframe, for example, clicking a link inside the Iframe. However, it is possible to change an IFrame's content from outside the IFrame. You can use an anchor tag whose href attribute is set to the desired URL and whose target attribute is set to the iframe's name attribute.

```
<iframe src="webpage.html" name="myIframe"></iframe>
<a href="different_webpage.html" target="myIframe">Change the Iframe content to
different_webpage.html</a>
```

第30章：内容语言

第30.1节：基础文档语言

在 `html` 元素中声明文档的主要语言是一种良好做法：

```
<html lang="en">
```

如果文档中没有指定其他 `lang` 属性，则表示 `everything`（即元素内容和属性文本值）均为该语言。

如果文档包含其他语言的部分，这些部分应获得自己的 `lang` 属性以“覆盖”语言声明。

第30.2节：元素语言

`lang` 属性用于指定元素内容和属性文本值的语言：

```
<p lang="en">该元素的内容为英文。</p><p lang="en" title="该属性的值也是英文。">该元素的内容为英文。</p>
```

语言声明会被继承：

```
<div lang="en">
  <p>该元素包含英文内容。</p>
  <p title="这个属性也是。">该元素同样如此。</p>
</div>
```

第30.3节：包含多语言的元素

您可以“覆盖”语言声明：

```
<p lang="en">这句英文中包含德语单词<span lang="de">Hallo</span>。</p>
```

第30.4节：区域性网址

可以向创建超链接的`<a>`和`<area>`元素添加属性 `hreflang`。该属性指定了链接资源的语言。定义的语言必须是有效的BC P 47[1]语言标签。

```
<p>
  <a href="example.org" hreflang="en">example.org</a>是IANA的示例域名之一。
</p>
```

1. ↑ IETF网络工作组：RFC 5646 语言标识标签，IETF，2009年9月

第30.5节：处理不同语言的属性

您可以通过引入除`applet`、`base`、

`basefont`、`br`、`frame`、`frameset`、`hr`、`iframe`、`meta`、`param`、`script`（HTML 4.0）之外的任何元素，并为其设置自己的`lang`属性，来“覆盖”父元素的语言声明：

Chapter 30: Content Languages

Section 30.1: Base Document Language

It's a good practice to declare the primary language of the document in the `html` element:

```
<html lang="en">
```

If no other `lang` attribute is specified in the document, it means that *everything* (i.e., element content and attribute text values) is in that language.

If the document contains parts in other languages, these parts should get their own `lang` attributes to "overwrite" the language declaration.

Section 30.2: Element Language

The `lang` attribute is used to specify the language of element content and attribute text values:

```
<p lang="en">The content of this element is in English.</p>
<p lang="en" title="The value of this attribute is also in English.">The content of this element is in English.</p>
```

The language declaration gets inherited:

```
<div lang="en">
  <p>This element contains English content.</p>
  <p title="This attribute, too.">Same with this element.</p>
</div>
```

Section 30.3: Elements with Multiple Languages

You can "overwrite" a language declaration:

```
<p lang="en">This English sentence contains the German word <span lang="de">Hallo</span>.</p>
```

Section 30.4: Regional URLs

It is possible to add the attribute `hreflang` to the elements `<a>` and `<area>` that create hyperlinks. Such it specifies the language of the linked resource. The language defined must be a valid BCP 47[1] language tag.

```
<p>
  <a href="example.org" hreflang="en">example.org</a> is one of IANA's example domains.
</p>
```

1. ↑ IETF Network Working Group: RFC 5646 *Tags for Identifying Languages*, IETF, September 2009

Section 30.5: Handling Attributes with Different Languages

You can "overwrite" a parent element's language declaration by introducing any element apart from `applet`, `base`, `basefont`, `br`, `frame`, `frameset`, `hr`, `iframe`, `meta`, `param`, `script` (of HTML 4.0) with an own `lang` attribute:

```
<p lang="en" title="An English paragraph">
  <span lang="de" title="A German sentence">Hallo Welt!</span>
</p>
```

```
<p lang="en" title="An English paragraph">
  <span lang="de" title="A German sentence">Hallo Welt!</span>
</p>
```

第31章：SVG

SVG代表可缩放矢量图形。SVG用于定义网页图形

HTML中的<svg>元素是SVG图形的容器。

SVG有多种方法用于绘制路径、盒子、圆形、文本和图像。

第31.1节：内联SVG

SVG可以直接写入HTML文档。内联SVG可以使用CSS和

JavaScript进行样式设置和操作。

```
<body>
  <svg class="attention" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 0 1000 1000" >
    <path id="attention"
d="m571,767l0,-106q0,-8,-5,-13t-12,-5l-108,0q-7,0,-12,5t-5,13l0,106q0,8,5,13t12,6l108,0q7,0,12,-6t5,-13Zm-1,-208l10,-257q0,-6,-5,-10q-7,-6,-14,-6l-122,0q-7,0,-14,6q-5,4,-5,12l9,255q0,5,6,9t13,3l103,0q8,0,13,-3t6,-9Zm-7,-522l428,786q20,35,-1,70q-10,17,-26,26t-35,10l-858,0q-18,0,-35,-10t-26,-26q-21,-35,-1,-70l429,-786q9,-17,26,-27t36,-10t36,10t27,27Z" />
  </svg>
</body>
```

上述内联SVG随后可以使用相应的CSS类进行样式设置：

```
.attention {
  fill: red;
  width: 50px;
  height: 50px;
}
```

效果如下所示：



第31.2节：在HTML中嵌入外部SVG文件

您可以使用或<object>元素来嵌入外部SVG元素。设置高度和宽度是可选的，但强烈推荐设置。

使用图像元素

```

```

使用不允许您使用CSS对SVG进行样式设置或使用JavaScript进行操作。

使用 object 元素

```
<object type="image/svg+xml" data="attention.svg" width="50" height="50">
```

与不同，<object>直接将 SVG 导入文档，因此可以使用

Javascript 和 CSS 进行操作。

Chapter 31: SVG

SVG stands for Scalable Vector Graphics. SVG is used to define graphics for the Web

The HTML <svg> element is a container for SVG graphics.

SVG has several methods for drawing paths, boxes, circles, text, and graphic images.

Section 31.1: Inline SVG

SVG can be written directly into a HTML document. Inline SVG can be styled and manipulated using CSS and JavaScript.

```
<body>
  <svg class="attention" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 0 1000 1000" >
    <path id="attention"
d="m571,767l0,-106q0,-8,-5,-13t-12,-5l-108,0q-7,0,-12,5t-5,13l0,106q0,8,5,13t12,6l108,0q7,0,12,-6t5,-13Zm-1,-208l10,-257q0,-6,-5,-10q-7,-6,-14,-6l-122,0q-7,0,-14,6q-5,4,-5,12l9,255q0,5,6,9t13,3l103,0q8,0,13,-3t6,-9Zm-7,-522l428,786q20,35,-1,70q-10,17,-26,26t-35,10l-858,0q-18,0,-35,-10t-26,-26q-21,-35,-1,-70l429,-786q9,-17,26,-27t36,-10t36,10t27,27Z" />
  </svg>
</body>
```

The above inline SVG can then be styled using the corresponding CSS class:

```
.attention {
  fill: red;
  width: 50px;
  height: 50px;
}
```

The result looks like this:



Section 31.2: Embedding external SVG files in HTML

You can use the or <object> elements to embed external SVG elements. Setting the height and width is optional but is highly recommended.

Using the image element

```

```

Using does not allow you to style the SVG using CSS or manipulate it using JavaScript.

Using the object element

```
<object type="image/svg+xml" data="attention.svg" width="50" height="50">
```

Unlike , <object> directly imports the SVG into the document and therefore it can be manipulated using Javascript and CSS.

第 31.3 节：使用 CSS 嵌入 SVG

您可以使用background-image属性添加外部 SVG 文件，就像添加其他图片一样。

HTML：

```
<div class="attention"></div>
```

CSS：

```
.attention {
background-image: url(attention.svg);
background-size: 100% 100%;
宽度: 50像素;
高度: 50像素;
}
```

你也可以使用数据 URL 直接将图像嵌入到 CSS 文件中：

```
background-image:
url(data:image/svg+xml,%3Csvg%20xmlns%3D%22http%3A%2F%2Fwww.w3.org%2F2000%2Fsvg%22%20xmlns%3Axlink%3D%22http%3A%2F%2Fwww.w3.org%2F1999%2Fxlink%22%20viewBox%3D%220%200%201000%201000%22%20%3E%0D%0A%3Cpath%20id%3D%22attention%22%20d%3D%22m571%2C767l0%2C-106q0%2C-8%2C-5%2C-13t-12%2C-5l-108%2C0q-7%2C0%2C-12%2C5t-5%2C13l0%2C106q0%2C8%2C5%2C13t12%2C6l108%2C0q7%2C0%2C12%2C-6t5%2C-13Zm-1%2C-208l10%2C-257q0%2C-6%2C-5%2C-10q-7%2C-6%2C-14%2C-6l-122%2C0q-7%2C0%2C-14%2C6q-5%2C4%2C-5%2C12l9%2C255q0%2C5%2C6%2C9t13%2C3l103%2C0q8%2C0%2C13%2C-3t6%2C-9Zm-7%2C-522l428%2C786q20%2C35%2C-1%2C70q-10%2C17%2C-26%2C26t-35%2C10l-858%2C0q-18%2C0%2C-35%2C-10t-26%2C-26q-21%2C-35%2C-1%2C-70l429%2C-786q9%2C-17%2C26%2C-27t36%2C-10t36%2C10t27%2C27Z%22%20%2F%3E%0D%0A%3C%2Fsvg%3E);
```

Section 31.3: Embedding SVG using CSS

You can add external SVG files using the background-image property, just as you would do with any other image.

HTML:

```
<div class="attention"></div>
```

CSS:

```
.attention {
background-image: url(attention.svg);
background-size: 100% 100%;
width: 50px;
height: 50px;
}
```

You can also embed the image directly into a css file using a data url:

```
background-image:
url(data:image/svg+xml,%3Csvg%20xmlns%3D%22http%3A%2F%2Fwww.w3.org%2F2000%2Fsvg%22%20xmlns%3Axlink%3D%22http%3A%2F%2Fwww.w3.org%2F1999%2Fxlink%22%20viewBox%3D%220%200%201000%201000%22%20%3E%0D%0A%3Cpath%20id%3D%22attention%22%20d%3D%22m571%2C767l0%2C-106q0%2C-8%2C-5%2C-13t-12%2C-5l-108%2C0q-7%2C0%2C-12%2C5t-5%2C13l0%2C106q0%2C8%2C5%2C13t12%2C6l108%2C0q7%2C0%2C12%2C-6t5%2C-13Zm-1%2C-208l10%2C-257q0%2C-6%2C-5%2C-10q-7%2C-6%2C-14%2C-6l-122%2C0q-7%2C0%2C-14%2C6q-5%2C4%2C-5%2C12l9%2C255q0%2C5%2C6%2C9t13%2C3l103%2C0q8%2C0%2C13%2C-3t6%2C-9Zm-7%2C-522l428%2C786q20%2C35%2C-1%2C70q-10%2C17%2C-26%2C26t-35%2C10l-858%2C0q-18%2C0%2C-35%2C-10t-26%2C-26q-21%2C-35%2C-1%2C-70l429%2C-786q9%2C-17%2C26%2C-27t36%2C-10t36%2C10t27%2C27Z%22%20%2F%3E%0D%0A%3C%2Fsvg%3E);
```


第32章：画布

属性	描述
高度	指定画布高度
宽度	指定画布宽度

第32.1节：基本示例

HTML5 引入了canvas元素用于绘制图形。

```
<canvas id="myCanvas">
无法显示图形。您的浏览器不支持画布（IE<9）
</canvas>
```

上述代码将创建一个透明的 HTML<canvas> 元素，大小为 300×150 像素。

你可以使用canvas元素通过JavaScript绘制各种精彩内容，如形状、图表、处理图像、制作有趣的游戏等。

canvas 的二维可绘制层表面对象称为CanvasRenderingContext2D；也可以通过 HTMLCanvasElement的.getContext("2d")方法获取：

```
var ctx = document.getElementById("myCanvas").getContext("2d");
// 现在我们可以使用 `ctx` 来引用 canvas 的二维层上下文

ctx.fillStyle = "#f00";
ctx.fillRect(0, 0, ctx.canvas.width, ctx.canvas.height); // x, y, 宽度, 高度

ctx.fillStyle = "#000";
ctx.fillText("我的红色画布，上面有一些黑色文字", 24, 32); // 文字, x, y
```

[jsFiddle 示例](#)

第32.2节：在<canvas>上绘制两个矩形

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <title>在画布上绘制两个矩形</title>
  <style>
    canvas{
border:1px solid gray;
    }
  </style>
  <script async>
window.onload = init; // 窗口完全加载后调用 init()
function init(){
// #1 - 获取 <canvas> 元素的引用
var canvas = document.querySelector('canvas');

// #2 - 获取绘图上下文和绘图API的引用
var ctx = canvas.getContext('2d');

// #3 - 所有填充操作现在使用红色
ctx.fillStyle = 'red';
```

Chapter 32: Canvas

Attribute	Description
height	Specifies the canvas height
width	Specifies the canvas width

Section 32.1: Basic Example

The canvas element was introduced in HTML5 for drawing graphics.

```
<canvas id="myCanvas">
  Cannot display graphic. Canvas is not supported by your browser (IE<9)
</canvas>
```

The above will create a transparent HTML<canvas> element of 300×150 px in size.

You can use the **canvas** element to draw amazing stuff like shapes, graphs, manipulate images, create engaging games etc. with **JavaScript**. The canvas's 2D *drawable layer* surface Object is referred to as CanvasRenderingContext2D; or from a HTMLCanvasElement using the .getContext("2d") method:

```
var ctx = document.getElementById("myCanvas").getContext("2d");
// now we can refer to the canvas's 2D layer context using `ctx`

ctx.fillStyle = "#f00";
ctx.fillRect(0, 0, ctx.canvas.width, ctx.canvas.height); // x, y, width, height

ctx.fillStyle = "#000";
ctx.fillText("My red canvas with some black text", 24, 32); // text, x, y
```

[jsFiddle example](#)

Section 32.2: Drawing two rectangles on a <canvas>

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <title>Draw two rectangles on the canvas</title>
  <style>
    canvas{
      border:1px solid gray;
    }
  </style>
  <script async>
    window.onload = init; // call init() once the window is completely loaded
    function init(){
      // #1 - get reference to <canvas> element
      var canvas = document.querySelector('canvas');

      // #2 - get reference to the drawing context and drawing API
      var ctx = canvas.getContext('2d');

      // #3 - all fill operations are now in red
      ctx.fillStyle = 'red';
```

```
// #4 - 在 x=0,y=0 处填充一个 100x100 的矩形
    ctx.fillRect(0,0,100,100);

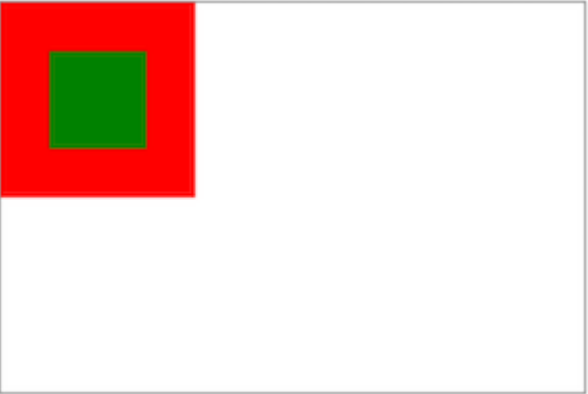
// #5 - 所有填充操作现在都是绿色的
    ctx.fillStyle = 'green';

// #6 - 在 x=25, y=25 处填充一个 50x50 的矩形
    ctx.fillRect(25,25,50,50);

    }

</head>
<body>
    <canvas width=300 height=200>您的浏览器不支持 canvas。</canvas>
</body>
</html>
```

此示例如下所示：



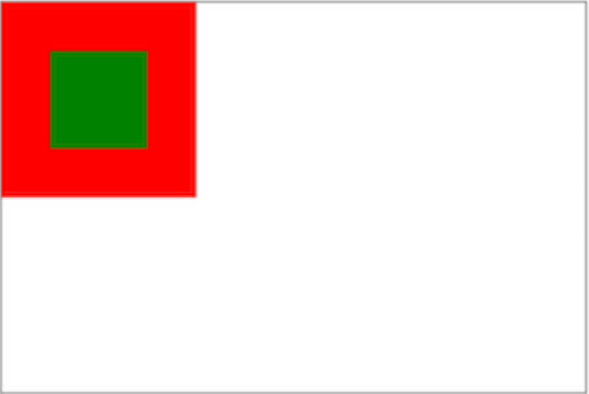
```
// #4 - fill a 100x100 rectangle at x=0,y=0
ctx.fillRect(0,0,100,100);

// #5 - all fill operations are now in green
ctx.fillStyle = 'green';

// #6 - fill a 50x50 rectangle at x=25,y=25
ctx.fillRect(25,25,50,50);

    }
</script>
</head>
<body>
    <canvas width=300 height=200>Your browser does not support canvas.</canvas>
</body>
</html>
```

This example looks like this:



第33章：元信息

HTML文档中的元标签提供有关文档的有用信息，包括描述、关键词、作者、修改日期以及大约90个其他字段。本主题涵盖这些标签的用法和目的。

第33.1节：页面信息

application-name

指定网页所代表的Web应用程序的名称。

```
<meta name="application-name" content="OpenStreetMap">
```

如果这不是一个Web应用程序，则不得使用application-name元标签。

作者

设置页面的作者：

```
<meta name="author" content="您的姓名">
```

只能填写一个姓名。

描述

设置页面的描述：

```
<meta name="description" content="页面描述">
```

description元标签可被各种搜索引擎用于索引您的网页以供搜索目的。通常，元标签中的描述是显示在搜索引擎结果中网页/网站主标题下方的简短摘要。谷歌通常只使用描述的前20-25个单词。

生成器

```
<meta name="generator" content="HTML Generator 1.42">
```

标识用于生成文档的软件包之一。仅用于标记自动生成的页面。

关键词

为搜索引擎设置关键词（用逗号分隔）：

```
<meta name="keywords" content="Keyword1, Keyword2">
```

关键词（keywords）元标签有时被搜索引擎用来了解与您的网页相关的搜索查询。

一般来说，最好不要添加太多词汇，因为大多数使用此元标签进行索引的搜索引擎只会索引前约20个词。确保将最重要的关键词放在最前面。

第33.2节：字符编码

charset属性指定HTML文档的字符编码，必须是有效的字符编码

Chapter 33: Meta Information

Meta tags in HTML documents provide useful information about the document including a description, keywords, author, dates of modifications and around 90 other fields. This topic covers the usage and purpose of these tags.

Section 33.1: Page Information

application-name

Giving the name of the Web application that the page represents.

```
<meta name="application-name" content="OpenStreetMap">
```

If it's not a Web application, the application-name meta tag must not be used.

author

Set the author of the page:

```
<meta name="author" content="Your Name">
```

Only one name can be given.

description

Set the description of the page:

```
<meta name="description" content="Page Description">
```

The description meta tag can be used by various search engines while indexing your web page for searching purpose. Usually, the description contained within the meta tag is the short summary that shows up under the page/website's main title in the search engine results. Google usually uses only the first 20-25 words of your description.

generator

```
<meta name="generator" content="HTML Generator 1.42">
```

Identifies one of the software packages used to generate the document. Only to be used for pages where the markup is automatically generated.

keywords

Set keywords for search engines (comma-separated):

```
<meta name="keywords" content="Keyword1, Keyword2">
```

The keywords meta tag is sometimes used by search engines to know the search query which is relevant to your web page. As a rule of thumb, it is probably a good idea to not add too many words, as most search engines that use this meta tag for indexing will only index the first ~20 words. Make sure that you put the most important keywords first.

Section 33.2: Character Encoding

The charset attribute specifies the character encoding for the HTML document and needs to be a valid character

（例如windows-1252、ISO-8859-2、Shift_JIS和UTF-8）。UTF-8（Unicode）是最广泛使用的，应当用于任何新项目。

```
版本 = 5

<meta charset="UTF-8">

<meta charset="ISO-8859-1">
```

所有浏览器一直都支持<meta charset>形式，但如果你因为某些原因需要你的页面符合有效的HTML 4.01标准，可以改用以下方式：

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

<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
```

另请参见[编码标准](#)，以查看浏览器识别的所有可用字符编码标签。

第33.3节：机器人

robots属性，由多个主要搜索引擎支持，用于控制搜索引擎蜘蛛是否允许索引页面，以及是否应跟踪页面上的链接。

```
<meta name="robots" content="noindex">
```

此示例指示所有搜索引擎不在搜索结果中显示该页面。其他允许的值包括：

值/指令	含义
全部	默认。 等同于 index，遵循。见下方说明。
noindex	完全不对该页面建立索引。
nofollow	不跟踪该页面上的链接
follow	页面上的链接可以被跟踪。见下方说明。
none	等同于 noindex, nofollow。
noarchive	不要在搜索结果中提供此页面的缓存版本。
nocache	某些机器人（如必应）使用的noarchive的同义词。
nosnippet	不要在搜索结果中显示此页面的摘要。
noodp	不要使用 开放目录项目 中的此页面元数据作为搜索结果中的标题或摘要。
notranslate	不要在搜索结果中提供此页面的翻译。
noimageindex	请勿对本页中的图片建立索引。
unavailable_after [RFC-850 date/time]	指定的日期/时间之后，请勿在搜索结果中显示此页面。日期/时间必须以RFC 850格式指定。

注意： 明确定义index和/或follow虽然是有效值，但并非必要，因为几乎所有搜索引擎都会假设允许这样做，除非明确禁止。类似于robots.txt文件的工作方式，搜索引擎通常只关注它们不允许做的事情。仅声明搜索引擎不允许做的事情，也避免了意外声明相反内容（例如index, ..., noindex），而不是所有搜索引擎都会以相同方式处理这些内容。

第33.4节：社交媒体

Open Graph是一种元数据标准，用于扩展网站head标记中包含的常规信息。这使得像Facebook这样的网站能够以结构化格式显示有关网站的更深入、更丰富的信息。当用户分享包含

encoding (examples include windows-1252, ISO-8859-2, Shift_JIS, and UTF-8). UTF-8 (Unicode) is the most widely used and should be used for any new project.

```
Version = 5

<meta charset="UTF-8">

<meta charset="ISO-8859-1">
```

All browsers have always recognized the <meta charset> form, but if you for some reason need your page to be valid HTML 4.01, you can use the following instead:

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

<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
```

See also the [Encoding Standard](#), to view all available character encoding labels that browsers recognize.

Section 33.3: Robots

The robots attribute, supported by several major search engines, controls whether search engine spiders are allowed to index a page or not and whether they should follow links from a page or not.

```
<meta name="robots" content="noindex">
```

This example instructs all search engines to not show the page in search results. Other allowed values are:

Value/Directive	Meaning
all	Default. Equivalent to index, follow. See note below.
noindex	Do not index the page at all.
nofollow	Do not follow the links on this page
follow	The links on the page can be followed. See note below.
none	Equivalent to noindex, nofollow.
noarchive	Do not make a cached version of this page available in search results.
nocache	Synonym of noarchive used by some bots such as Bing.
nosnippet	Do not show a snippet of this page in search results.
noodp	Do not use metadata of this page from the Open Directory project for titles or snippets in search results.
notranslate	Do not offer translations of this page in search results.
noimageindex	Do not index images on this page.
unavailable_after [RFC-850 date/time]	Do not show this page in search results after the specified date/time. The date/time must be specified in the RFC 850 format .

Note: Explicitly defining index and/or follow, while valid values, is not necessary as pretty much all search engines will assume they are allowed to do so if not explicitly prevented from doing so. Similar to how the robots.txt file operates, search engines generally only look for things they are *not allowed* to do. Only stating things a search engine isn't allowed to do also prevents accidentally stating opposites (such as index, ..., noindex) which not all search engines will treat in the same way.

Section 33.4: Social Media

Open Graph is a standard for metadata that extends the normal information contained within a site's head markup. This enables websites such as Facebook to display deeper and richer information about a website in a structured format. This information is then automatically displayed when users share links to websites containing

Facebook / Open Graph

```
<meta property="fb:app_id" content="123456789">
<meta property="og:url" content="https://example.com/page.html">
<meta property="og:type" content="website">
<meta property="og:title" content="内容标题">
<meta property="og:image" content="https://example.com/image.jpg">
<meta property="og:description" content="此处为描述">
<meta property="og:site_name" content="Site Name">
<meta property="og:locale" content="en-US">
<meta property="article:author" content="">
<!-- Facebook: https://developers.facebook.com/docs/sharing/webmasters#markup -->
<!-- Open Graph: http://ogp.me/ -->
```

- [Facebook Open Graph 标记](#)
- [Open Graph 协议](#)

Facebook / 即时文章

```
<meta charset="utf-8">
<meta property="op:markup_version" content="v1.0">

<!-- 您文章网页版的 URL -->
<link rel="canonical" href="http://example.com/article.html">

<!-- 本文使用的样式 -->
<meta property="fb:article_style" content="myarticlestyle">
```

- [Facebook 即时文章：创建文章](#)
- [即时文章：格式参考](#)

Twitter 使用其自己的元数据标记。该元数据用于控制当推文包含指向该网站链接时，推文的显示方式。

Twitter

```
<meta name="twitter:card" content="summary">
<meta name="twitter:site" content="@site_account">
<meta name="twitter:creator" content="@individual_account">
<meta name="twitter:url" content="https://example.com/page.html">
<meta name="twitter:title" content="内容标题">
<meta name="twitter:description" content="内容描述，不超过200个字符">
<meta name="twitter:image" content="https://example.com/image.jpg">
```

- [Twitter 卡片：入门指南](#)
- [Twitter 卡片验证工具](#)

Google+ / Schema.org

```
<link href="https://plus.google.com/+YourPage" rel="publisher">
<meta itemprop="name" content="内容标题">
<meta itemprop="description" content="内容描述，不超过200个字符">
<meta itemprop="image" content="https://example.com/image.jpg">
```

第33.5节：移动布局控制

常见的移动优化网站使用如下的<meta name="viewport">标签：

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

OG metadata on Facebook.

Facebook / Open Graph

```
<meta property="fb:app_id" content="123456789">
<meta property="og:url" content="https://example.com/page.html">
<meta property="og:type" content="website">
<meta property="og:title" content="Content Title">
<meta property="og:image" content="https://example.com/image.jpg">
<meta property="og:description" content="Description Here">
<meta property="og:site_name" content="Site Name">
<meta property="og:locale" content="en-US">
<meta property="article:author" content="">
<!-- Facebook: https://developers.facebook.com/docs/sharing/webmasters#markup -->
<!-- Open Graph: http://ogp.me/ -->
```

- [Facebook Open Graph Markup](#)
- [Open Graph protocol](#)

Facebook / Instant Articles

```
<meta charset="utf-8">
<meta property="op:markup_version" content="v1.0">

<!-- The URL of the web version of your article -->
<link rel="canonical" href="http://example.com/article.html">

<!-- The style to be used for this article -->
<meta property="fb:article_style" content="myarticlestyle">
```

- [Facebook Instant Articles: Creating Articles](#)
- [Instant Articles: Format Reference](#)

Twitter uses its own markup for metadata. This metadata is used as information to control how tweets are displayed when they contain a link to the site.

Twitter

```
<meta name="twitter:card" content="summary">
<meta name="twitter:site" content="@site_account">
<meta name="twitter:creator" content="@individual_account">
<meta name="twitter:url" content="https://example.com/page.html">
<meta name="twitter:title" content="Content Title">
<meta name="twitter:description" content="Content description less than 200 characters">
<meta name="twitter:image" content="https://example.com/image.jpg">
```

- [Twitter Cards: Getting Started Guide](#)
- [Twitter Card Validator](#)

Google+ / Schema.org

```
<link href="https://plus.google.com/+YourPage" rel="publisher">
<meta itemprop="name" content="Content Title">
<meta itemprop="description" content="Content description less than 200 characters">
<meta itemprop="image" content="https://example.com/image.jpg">
```

Section 33.5: Mobile Layout Control

Common mobile-optimized sites use the <meta name="viewport"> tag like this:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```


viewport元素为浏览器提供了如何根据您使用的设备控制页面尺寸和缩放的指令。

在上述示例中，content="width=device-width"表示浏览器将以其自身屏幕的宽度来渲染页面宽度。所以如果该屏幕宽度为480px，浏览器窗口宽度也将是480px。initial-scale=1表示初始缩放比例（此处为1，意味着不缩放）。

下面是该标签支持的属性：

属性	描述
width	设备虚拟视口的宽度。 取值1：device-width或实际像素宽度，如480
height	设备虚拟视口的高度。 取值2：device-height或实际像素高度，如600
initial-scale	页面加载时的初始缩放比例。1.0表示不缩放。
minimum-scale	访客在页面上可以缩放的最小比例。1.0表示不缩放。
maximum-scale	访客在页面上可以缩放的最大比例。1.0表示不缩放。
user-scalable	允许设备进行缩放。取值为 yes 或 no。如果设置为 no，用户将无法在网页中缩放网页。默认值为 yes。浏览器设置可能会忽略此规则。

注释：

1 width 属性可以用 pixels 指定（width=600），也可以用 device-width 指定（width=device-width），表示设备屏幕的物理宽度。

2 同样，height 属性可以用 pixels 指定（height=600），也可以用 device-height（height=device-height）指定，表示设备屏幕的物理高度。

第33.6节：自动刷新

要每五秒刷新一次页面，请在 head 元素中添加此 meta 元素：

```
<meta http-equiv="refresh" content="5">
```

注意！虽然这是一个有效的命令，但建议不要使用它，因为它会对用户体验产生负面影响。页面刷新过于频繁可能导致页面无响应，且经常滚动到页面顶部。如果页面上的某些信息需要持续更新，有更好的方法只刷新页面的一部分。

第33.7节：电话号码识别

像 iOS 这样的移动平台会自动识别电话号码并将其转换为 tel: 链接。虽然该功能非常实用，但系统有时会将 ISBN 码和其他数字误识别为电话号码。

对于移动版Safari和其他一些基于WebKit的移动浏览器，要关闭自动电话号码识别和格式化，您需要使用此meta标签：

```
<meta name="format-detection" content="telephone=no">
```

第33.8节：自动重定向

有时您的网页需要自动重定向。

The viewport element gives the browser instructions on how to control the page's dimensions and scaling based on the device you are using.

In the above example, content="width=device-width means that the browser will render the width of the page at the width of its own screen. So if that screen is 480px wide, the browser window will be 480px wide. initial-scale=1 depicts that the initial zoom (which is 1 in this case, means it does not zoom).

Below are the attributes this tag supports:

Attribute	Description
width	The width of the virtual viewport of the device. Values1: device-width or the actual width in pixels, like 480
height	The height of the virtual viewport of the device. Values2: device-height or the actual width in pixels, like 600
initial-scale	The initial zoom when the page is loaded. 1.0 does not zoom.
minimum-scale	The minimum amount the visitor can zoom on the page. 1.0 does not zoom.
maximum-scale	The maximum amount the visitor can zoom on the page. 1.0 does not zoom.
user-scalable	Allows the device to zoom in and out. Values are yes or no. If set to no, the user is not able to zoom in the webpage. The default is yes. Browser settings can ignore this rule.

Notes:

1 The width property can be either specified in pixels (width=600) or by device-width (width=device-width) which represents the physical width of the device's screen.

2 Similarly, the height property can be either specified in pixels (height=600) or by device-height (height=device-height) which represents the physical height of the device's screen.

Section 33.6: Automatic Refresh

To refresh the page every five seconds, add this meta element in the head element:

```
<meta http-equiv="refresh" content="5">
```

CAUTION! While this is a valid command, it is recommended that you do not use it because of its negative effects on user experience. Refreshing the page too often can cause it to become unresponsive, and often scrolls to the top of the page. If some information on the page needs to be updated continuously, there are much better ways to do that by only refreshing a portion of a page.

Section 33.7: Phone Number Recognition

Mobile platforms like iOS automatically recognize phone numbers and turn them into tel: links. While the feature is very practical, the system sometimes detects ISBN codes and other numbers as telephone numbers.

For mobile Safari and some other WebKit-based mobile browsers to turn off automatic phone number recognition and formatting, you need this meta tag:

```
<meta name="format-detection" content="telephone=no">
```

Section 33.8: Automatic redirect

Sometimes your webpage needs a automatic redirect.

例如，5秒后重定向到example.com：

```
<meta http-equiv="refresh" content="5;url=https://www.example.com/" />
```

这一行将在5秒后将您发送到指定的网站（本例中为example.com）。

如果您需要更改重定向前的延迟时间，只需更改;url=前面的数字即可改变延迟时间。

第33.9节：网页应用

您可以设置您的网页应用或网站，在设备主屏幕上添加应用快捷图标，并使用Android版Chrome的“添加到主屏幕”菜单项以全屏“应用模式”启动该应用。

以下meta标签将以全屏模式（无地址栏）打开网页应用。

安卓 Chrome

```
<meta name="mobile-web-app-capable" content="yes">
```

苹果 iOS

```
<meta name="apple-mobile-web-app-capable" content="yes">
```

你也可以在 meta 标签中设置状态栏和地址栏的颜色。

安卓 Chrome

```
<meta name="theme-color" content="black">
```

苹果 iOS

```
<meta name="apple-mobile-web-app-status-bar-style" content="black">
```

For example, to redirect to example.com after 5 seconds:

```
<meta http-equiv="refresh" content="5;url=https://www.example.com/" />
```

This is line will send you to the designated website (in this case example.com after 5 seconds.

If you need to change the time delay before a redirect, simply changing the number right before your ;url= will alter the time delay.

Section 33.9: Web App

You can set up your web app or website to have an application shortcut icon added to a device's homescreen, and have the app launch in full-screen "app mode" using Chrome for Android's "[Add to homescreen](#)" menu item.

Below meta tag(s) will open web app in full-screen mode (without address bar).

Android Chrome

```
<meta name="mobile-web-app-capable" content="yes">
```

IOS

```
<meta name="apple-mobile-web-app-capable" content="yes">
```

You can also set color for status bar and address bar in meta tag.

Android Chrome

```
<meta name="theme-color" content="black">
```

IOS

```
<meta name="apple-mobile-web-app-status-bar-style" content="black">
```

第34章：计算机代码的标记

Chapter 34: Marking up computer code

第34.1节：使用 <pre> 和 <code> 的代码块

Section 34.1: Block with <pre> and <code>

如果代码的格式（空白、换行、缩进）很重要，请将 pre 元素与 code 元素结合使用：

If the formatting (white space, new lines, indentation) of the code matters, use the pre element in combination with the code element:

```
<pre>
  <code>
x = 42
  if x == 42:
print "x is ...      ... 42"
  </code>
</pre>
```

```
<pre>
  <code>
x = 42
  if x == 42:
    print "x is ...      ... 42"
  </code>
</pre>
```

你仍然需要对HTML中具有特殊含义的字符进行转义（例如用<代替<），因此对于显示一段HTML代码（<p>这是一个段落。</p>），它可能看起来像这样：

You still have to escape characters with special meaning in HTML (like < with < ;), so for displaying a block of HTML code (<p>This is a paragraph.</p>), it could look like this:

```
<pre>
  <code>
&lt;p>这是一个段落。&lt;/p>
  </code>
</pre>
```

```
<pre>
  <code>
&lt;p>This is a paragraph.&lt;/p>
  </code>
</pre>
```

第34.2节：与<code>内联

Section 34.2: Inline with <code>

如果句子中包含计算机代码（例如HTML元素的名称），请使用code元素进行标记：

If a sentence contains computer code (for example, the name of an HTML element), use the code element to mark it up:

```
<p><code>a</code>元素创建一个超链接。</p>
```

```
<p>The <code>a</code> element creates a hyperlink.</p>
```

第35章：引用的标记

第35.1节：与<code>q</code>标签一致

q元素可用于句子中的引用：

```
<p>她写道<q>答案是42。</q>大家都同意。</p>
```

引号

版本 ≤ 4.01

不应添加引号。用户代理应（在HTML 4.01中）或必须（在HTML 4.0中）自动渲染引号。

版本 = 5

不得添加引号。用户代理将自动渲染引号。

来源URL（cite属性）

cite属性可用于引用来源的URL：

```
<p>她写道<q cite="http://example.com/blog/hello-world">答案是42。</q>大家都同意。</p>
```

请注意，浏览器通常不会显示此URL，因此如果来源相关，您应另外添加超链接（a元素）。

第35.2节：带有<code>blockquote</code>的块

blockquote元素可用于（块级）引用：

```
<blockquote>
  <p>答案是42。</p>
</blockquote>
```

来源URL（cite属性）

cite属性可用于引用来源的URL：

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>答案是42。</p>
</blockquote>
```

注意浏览器通常不会显示此URL，因此如果来源相关，您应另外添加超链接（a元素）（参见“引用/归属”部分关于链接放置的位置）。

引用/归属

版本 ≤ 4.01

引用/归属不应成为blockquote元素的一部分：

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>答案是42。</p>
</blockquote>
<p>来源：<cite><a href="http://example.com/blog/hello-world" rel="external">Hello
World</a></cite></p>
```

Chapter 35: Marking-up Quotes

Section 35.1: Inline with <code>q</code>

The **q element** can be used for a quote that is part of a sentence:

```
<p>She wrote <q>The answer is 42.</q> and everyone agreed.</p>
```

Quotation marks

Version ≤ 4.01

Quotation marks should not be added. User agents should (in HTML 4.01) resp. must (in HTML 4.0) render them automatically.

Version = 5

Quotation marks must not be added. User agents will render them automatically.

Source URL (cite attribute)

The **cite attribute** can be used to reference the URL of the quoted source:

```
<p>She wrote <q cite="http://example.com/blog/hello-world">The answer is 42.</q> and everyone agreed.</p>
```

Note that browsers typically don't show this URL, so if the source is relevant, you should add a hyperlink (a element) in addition.

Section 35.2: Block with <code>blockquote</code>

The **blockquote element** can be used for a (block-level) quote:

```
<blockquote>
  <p>The answer is 42.</p>
</blockquote>
```

Source URL (cite attribute)

The **cite attribute** can be used to reference the URL of the quoted source:

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>The answer is 42.</p>
</blockquote>
```

Note that browsers typically don't show this URL, so if the source is relevant, you should add a hyperlink (a element) in addition (see the section *Citation/Attribution* about where to place this link).

Citation/Attribution

Version ≤ 4.01

The citation/attribution should not be part of the `blockquote` element:

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>The answer is 42.</p>
</blockquote>
<p>Source: <cite><a href="http://example.com/blog/hello-world" rel="external">Hello
World</a></cite></p>
```

您可以添加一个div元素来组合引用和引文，但没有方法将它们在语义上关联起来。

cite元素可以用于引用来源的参考（但不能用于作者姓名）。

版本 = 5

引用/归属（例如，提供来源URL的超链接）可以放在blockquote内，但在这种情况下必须放在cite元素内（用于文内归属）或footer元素内：

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>答案是42。 </p>
  <footer>
    <p>来源：<cite><a href="http://example.com/blog/hello-world" rel="external">Hello
World</a></cite></p>
  </footer>
</blockquote>
```

cite元素可以用于引用来源的参考，或用于引用作者的姓名。

You can add a div element to group the quote and the citation, but it exists no way to associate them semantically.

The **cite element** can be used for the reference of the quoted source (but not for the author name).

Version = 5

The citation/attribution (e.g., the hyperlink giving the source URL) can be inside the `blockquote`, but in that case it must be within a `cite` element (for in-text attributions) or a footer element:

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>The answer is 42.</p>
  <footer>
    <p>Source: <cite><a href="http://example.com/blog/hello-world" rel="external">Hello
World</a></cite></p>
  </footer>
</blockquote>
```

The **cite element** can be used for the reference of the quoted source, or for the name of the quote's author.

第36章：Tabindex

值	含义
负值	元素将可聚焦，但不应通过顺序键盘导航访问
0	元素将可聚焦并可通过键盘顺序导航访问，但其相对顺序由平台约定定义
正值	元素必须可聚焦并可通过顺序键盘导航访问；其相对顺序将由属性值定义：顺序遵循 tabindex 的递增数字

第36.1节：向制表顺序添加元素

```
<div tabindex="0">某个按钮</div>
```

注意: 尽量在合适的地方使用原生HTML的button或 a 标签。

第36.2节：从制表顺序中移除元素

```
<button tabindex="-1">此按钮无法通过Tab键访问</button>
```

该元素将从制表顺序中移除，但仍然可以获得焦点。

第36.3节：定义自定义制表顺序（不推荐）

```
<div tabindex="2">第二</div>
<div tabindex="1">第一</div>
```

正值将把元素插入到其对应值的制表顺序位置。没有偏好（即 tabindex="0"或原生元素如button和 a）的元素将被追加到有偏好的元素之后。

不推荐使用正值，因为它们会破坏预期的制表行为，可能会让依赖屏幕阅读器的人感到困惑。请尝试通过重新排列DOM结构来创建自然顺序。

Chapter 36: Tabindex

Value	Meaning
negative	element will be focusable, but it should not be reachable via sequential keyboard navigation
0	element will be focusable and reachable through keyboard sequential navigation, but it's relative order is defined by the platform convention
positive	element must be focusable and accessible via sequential keyboard navigation; it's relative order will be defined by the attribute value: the sequential follow the increasing number of the <code>tabindex</code>

Section 36.1: Add an element to the tabbing order

```
<div tabindex="0">Some button</div>
```

Note: Try to use a native HTML button or an a tag where appropriate.

Section 36.2: Remove an element from the tabbing order

```
<button tabindex="-1">This button will not be reachable by tab</button>
```

The element will be removed from the tabbing order but will still be focusable.

Section 36.3: Define a custom tabbing order (not recommended)

```
<div tabindex="2">Second</div>
<div tabindex="1">First</div>
```

Positive values will insert the element at the tabbing order position of its respective value. Elements without preference (i.e. `tabindex="0"` or native elements such as button and a) will be appended after those with preference.

Positive values are **not recommended** as they disrupt the expected behavior of tabbing and might confuse people who rely on screenreaders. Try to create a natural order by rearranging your DOM structure.

第37章：全局属性

属性	描述
<code>class</code>	为元素定义一个或多个类名。参见类和ID。
<code>contenteditable</code>	设置元素内容是否可编辑。
<code>contextmenu</code>	定义用户右键点击元素时显示的上下文菜单。
<code>dir</code>	设置元素内文本的书写方向。
<code>draggable</code>	设置元素是否可被拖动。
<code>hidden</code>	隐藏当前页面上未使用的元素。
<code>id</code>	定义元素的唯一标识符。参见类和ID。
<code>语言</code>	定义元素内容及其文本属性值的语言。参见内容语言。
<code>拼写检查</code>	设置是否对元素内容进行拼写/语法检查。
<code>样式</code>	定义元素的一组内联CSS样式。
<code>标签顺序</code>	设置页面中元素通过Tab键快捷键导航的顺序。
<code>title</code>	定义有关元素的附加信息，通常以鼠标悬停时的工具提示文本形式显示。
<code>翻译</code>	定义是否翻译元素的内容。

第37.1节：contenteditable属性

```
<p contenteditable>这是一个可编辑的段落。</p>
```

点击该段落，其内容可以像输入文本框一样被编辑。

当元素上未设置contenteditable属性时，该元素将继承其父元素的属性。因此，内容可编辑元素的所有子文本也将是可编辑的，但你可以针对特定文本关闭编辑功能，如下所示：

```
<p contenteditable>
  这是一个可编辑的段落。
  <span contenteditable="false">但这部分不可编辑。</span>
</p>
```

注意，位于可编辑元素内的不可编辑文本元素仍会继承其父元素的文本光标。

Chapter 37: Global Attributes

Attribute	Description
<code>class</code>	Defines one or more class names for an element. See Classes and IDs.
<code>contenteditable</code>	Sets whether the content of an element can be edited.
<code>contextmenu</code>	Defines a context menu shown when a user right-clicks an element.
<code>dir</code>	Sets the text direction for text within an element.
<code>draggable</code>	Sets whether an element can be dragged.
<code>hidden</code>	Hides an element not currently in use on the page.
<code>id</code>	Defines a unique identifier for an element. See Classes and IDs.
<code>lang</code>	Defines the language of an element's content and its text attribute values. See Content Languages.
<code>spellcheck</code>	Sets whether to spell/grammar check the content of an element.
<code>style</code>	Defines a set of inline CSS styles for an element.
<code>tabindex</code>	Sets the order in which elements on a page are navigated by the tab keyboard shortcut.
<code>title</code>	Defines additional information about an element, generally in the form of tooltip text on mouseover.
<code>translate</code>	Defines whether to translate the content of an element.

Section 37.1: Contenteditable Attribute

```
<p contenteditable>This is an editable paragraph.</p>
```

Upon clicking on the paragraph, the content of it can be edited similar to an input text field.

When the contenteditable attribute is not set on an element, the element will inherit it from its parent. So all child text of a content editable element will also be editable, but you *can* turn it off for specific text, like so:

```
<p contenteditable>
  This is an editable paragraph.
  <span contenteditable="false">But not this.</span>
</p>
```

Note that an uneditable text element inside an editable element will still have a text cursor as inherited from its parent as well.

第38章：HTML5缓存

第38.1节：HTML5缓存的基本示例

这是我们的 index.html 文件

```
<!DOCTYPE html>
<html manifest="index.appcache">
<body>
  <p>内容</p>
</body>
</html>
```

然后我们将创建包含以下代码的 index.appcache 文件

```
CACHE MANIFEST
index.html
```

写入你想缓存的那些文件，加载 index.html，然后切换到离线模式并重新加载标签页

注意： 在此示例中，这两个文件必须位于同一文件夹内

Chapter 38: HTML 5 Cache

Section 38.1: Basic Example of HTML5 cache

this is our index.html file

```
<!DOCTYPE html>
<html manifest="index.appcache">
<body>
  <p>Content</p>
</body>
</html>
```

then we will create index.appcache file with below codes

```
CACHE MANIFEST
index.html
```

write those files that you want to be cached load index.html then go for offline mode and reload the tab

Note: The two files must be in the same folder in this example

第39章：HTML事件属性

第39.1节：HTML表单事件

由HTML表单内的操作触发的事件（适用于几乎所有HTML元素，但主要用于表单elements）：

属性	描述
<small>失去焦点事件 (onblur)</small>	元素失去焦点时触发
onchange	元素的值发生变化时触发
当触发上下文菜单时运行的脚本	
onfocus	元素获得焦点时触发
oninput	元素接收用户输入时运行的脚本
oninvalid	元素无效时运行的脚本
onreset	当表单中的重置按钮被点击时触发
onsearch	当用户在搜索字段（针对 <input="search">）中输入内容时触发
onselect	当元素中的部分文本被选中后触发
onsubmit	当表单提交时触发

第39.2节：键盘事件

属性	描述
onkeydown	当用户按下按键时触发
onkeypress	当用户按下按键时触发
onkeyup	当用户释放按键时触发

Chapter 39: HTML Event Attributes

Section 39.1: HTML Form Events

Events triggered by actions inside a HTML form (applies to almost all HTML elements, but is most used in form elements):

Attribute	Description
onblur	Fires the moment that the element loses focus
onchange	Fires the moment when the value of the element is changed
oncontextmenu	Script to be run when a context menu is triggered
onfocus	Fires the moment when the element gets focus
oninput	Script to be run when an element gets user input
oninvalid	Script to be run when an element is invalid
onreset	Fires when the Reset button in a form is clicked
onsearch	Fires when the user writes something in a search field (for <input="search">)
onselect	Fires after some text has been selected in an element
onsubmit	Fires when a form is submitted

Section 39.2: Keyboard Events

Attribute	Description
onkeydown	Fires when a user is pressing a key
onkeypress	Fires when a user presses a key
onkeyup	Fires when a user releases a key

第40章：字符实体

第40.1节：HTML中的字符实体

在使用HTML开发网页时，需要使用许多符号和特殊字符，但正如我们所知，有时直接使用字符可能会干扰实际的HTML代码，因为某些字符是保留的，且某些字符在键盘上不可用。因此，为了避免冲突，同时能够在代码中使用不同的符号，w3 org为我们提供了“字符实体”。

字符实体预定义了“实体名称” - &entity_name; 和“实体编号” - &entity_number;，因此我们需要使用这两者中的任意一个来在页面上呈现所需的符号。

部分字符实体列表可见于 <https://dev.w3.org/html5/html-author/charref>

使用字符实体表示“放大镜”的简单示例：

```
<input type="text" placeholder="  &#128269; 搜索"/>
```

渲染效果为



第40.2节：常用特殊字符

某些字符可能被HTML保留，不能直接使用，因为它们可能会阻碍实际的HTML代码。例如，尝试在源代码中显示左右尖括号（<>）可能会导致输出结果异常。同样，源代码中写入的空格在输出的HTML中可能不会按预期显示。有些字符，如☎，不包含在ASCII字符集中。

为此，创建了字符实体。它们的形式为&entity_name;或&entity_number;。以下是一些可用的HTML实体。

字符	描述	实体名称	实体编号
· ·	不间断空格 		
"<"	小于	<	<
">"	大于	>	>
"&"	和号	&	&
"—"	破折号	—	—
"—"	en dash	—	–
"©"	版权	©	©
"®"	注册商标 ®		®
"™"	商标	™	™
"☎"	电话	☎	☎

因此，写作

© 2016 Stack Exchange Inc.

使用了以下HTML代码：

```
<b>&copy; 2016 Stack Exchange Inc.</b>
```

Chapter 40: Character Entities

Section 40.1: Character Entities in HTML

Many symbols and special characters are required while developing a web page in html, but as we know that sometimes the use of characters directly may interfere with the actual html code which have certain characters reserved and also certain characters being not available on keyboard. Thus, to avoid the conflict and at same time to be able to use different symbols in our code w3 org provides us with 'Character Entities'.

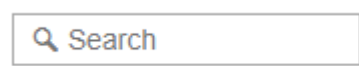
Character Entities are predefined with 'Entity Name' - &entity_name; and 'Entity Number' - &entity_number; so we need to use either of the two for the required symbol to be rendered on our page.

The list of few Character Entities can be found at <https://dev.w3.org/html5/html-author/charref>

A simple example with the use of character entity for 'magnifying glass' :

```
<input type="text" placeholder="  &#128269; Search"/>
```

which renders as



Section 40.2: Common Special Characters

Some character may be reserved for HTML and cannot be used directly as it may obstruct the actual HTML codes. For example, trying to display the left and right angle brackets (<>) in the source code may cause unexpected results in the output. Similarly, white spaces as written in the source code may not display as expected in the output HTML. Some, like ☎, are not available in the ASCII character set.

For this purpose, character entities are created. These are of the form &entity_name; or &entity_number;. The following are some of the available HTML entities.

Character	Description	Entity Name	Entity Number
" "	non-breaking space	 	
"<"	less than	<	<
">"	greater than	>	>
"&"	ampersand	&	&
"—"	em dash	—	—
"—"	en dash	–	–
"©"	copyright	©	©
"®"	registered trademark	®	®
"™"	trademark	™	™
"☎"	phone	☎	☎

Thus, to write

© 2016 Stack Exchange Inc.

the following HTML code is used:

```
<b>&copy; 2016 Stack Exchange Inc.</b>
```


第41章：ARIA

第41.1节：role="presentation"

一个其隐含的本地角色语义不会映射到辅助功能API的元素。

```
<div style="float:left;">左侧的一些内容。</div>
<div style="float:right;">右侧的一些内容</div>
<div role="presentation" style="clear:both;"></div> <!-- 仅用于清除浮动 -->
```

第41.2节：role="alert"

包含重要且通常是时间敏感信息的消息。

```
<div role="alert" aria-live="assertive">您的会话将在60秒后过期。</div>
```

请注意，我同时包含了role="alert"和aria-live="assertive"。这些是同义属性，但有些屏幕阅读器只支持其中一个。通过同时使用两者，我们最大化了实时区域按预期工作的可能性。

来源 - Heydon Pickering [‘一些实用的ARIA示例’](#)

第41.3节：role="alertdialog"

一种包含警告信息的对话框类型，初始焦点会移至对话框内的某个元素。

```
<div role="alertdialog">
  <h1>警告</h1>
  <div role="alert">您的会话将在60秒后过期。</div>
</div>
```

第41.4节：role="application"

声明为网络应用程序的区域，与网络文档相对。在此示例中，该应用程序是一个简单的计算器，可能用于将两个数字相加。

```
<div role="application">
  <h1>计算器</h1>
  <input id="num1" type="text"> + <input id="num2" type="text"> =
  <span id="result"></span>
</div>
```

第41.5节：role="article"

页面的一部分，由构成文档、页面或网站独立部分的内容组成。

设置与默认隐式ARIA语义匹配的ARIA角色和/或aria-*属性是不必要且不推荐，因为这些属性已经由浏览器设置。

Chapter 41: ARIA

Section 41.1: role="presentation"

An element whose implicit native role semantics will not be mapped to the accessibility API.

```
<div style="float:left;">Some content on the left.</div>
<div style="float:right;">Some content on the right</div>
<div role="presentation" style="clear:both;"></div> <!-- Only used to clear floats -->
```

Section 41.2: role="alert"

A message with important, and usually time-sensitive, information.

```
<div role="alert" aria-live="assertive">Your session will expire in 60 seconds.</div>
```

Note that I've included both role="alert" and aria-live="assertive" at the same time. These are synonymous attributes, but some screen readers only support one or the other. By using both simultaneously we therefore maximize the chances that the live region will function as expected.

Source - Heydon Pickering ['Some practical ARIA examples'](#)

Section 41.3: role="alertdialog"

A type of dialog that contains an alert message, where initial focus goes to an element within the dialog.

```
<div role="alertdialog">
  <h1>Warning</h1>
  <div role="alert">Your session will expire in 60 seconds.</div>
</div>
```

Section 41.4: role="application"

A region declared as a web application, as opposed to a web document. In this example, the application is a simple calculator that might add two numbers together.

```
<div role="application">
  <h1>Calculator</h1>
  <input id="num1" type="text"> + <input id="num2" type="text"> =
  <span id="result"></span>
</div>
```

Section 41.5: role="article"

A section of a page that consists of a composition that forms an independent part of a document, page, or site.

Setting an ARIA role and/or aria-* attribute that matches the default implicit ARIA semantics is unnecessary and is not recommended as these properties are already set by the browser.

```
<article>
  <h1>我的第一篇文章</h1>
  <p>Lorem ipsum...</p>
</article>
```

您会在非语义元素上使用role=article（不推荐，无效）

```
<div role="article">
  <h1>我的第一篇文章</h1>
  <p>Lorem ipsum...</p>
</div>
```

W3C关于role=article的条目

第41.6节：role="banner"

包含主要面向站点内容的区域，而非特定页面内容。

```
<div role="banner">
  <h1>我的网站</h1>

  <ul>
    <li><a href="/">首页</a></li>
    <li><a href="/about">关于</a></li>
    <li><a href="/contact">联系</a></li>
  </ul>
</div>
```

第41.7节：role="button"

允许用户点击或按下时触发操作的输入控件。

```
<button role="button">添加</button>
```

第41.8节：role="cell"

表格容器中的单元格。

```
<table>
  <thead>
<!-- 等等 -->
  </thead>
  <tbody>
    <td role="cell">95</td>
    <td role="cell">14</td>
    <td role="cell">25</td>
  </tbody>
</table>
```

第41.9节：role="checkbox"

一个可勾选的输入控件，具有三种可能的值：true、false 或 mixed。

```
<p>
  <input type="checkbox" role="checkbox" aria-checked="false">
  我同意条款
```

```
<article>
  <h1>My first article</h1>
  <p>Lorem ipsum...</p>
</article>
```

You would use role=article on non-semantic elements (not recommended, invalid)

```
<div role="article">
  <h1>My first article</h1>
  <p>Lorem ipsum...</p>
</div>
```

W3C Entry for role=article

Section 41.6: role="banner"

A region that contains mostly site-oriented content, rather than page-specific content.

```
<div role="banner">
  <h1>My Site</h1>

  <ul>
    <li><a href="/">Home</a></li>
    <li><a href="/about">About</a></li>
    <li><a href="/contact">Contact</a></li>
  </ul>
</div>
```

Section 41.7: role="button"

An input that allows for user-triggered actions when clicked or pressed.

```
<button role="button">Add</button>
```

Section 41.8: role="cell"

A cell in a tabular container.

```
<table>
  <thead>
<!-- etc -->
  </thead>
  <tbody>
    <td role="cell">95</td>
    <td role="cell">14</td>
    <td role="cell">25</td>
  </tbody>
</table>
```

Section 41.9: role="checkbox"

A checkable input that has three possible values: true, false, or mixed.

```
<p>
  <input type="checkbox" role="checkbox" aria-checked="false">
  I agree to the terms
```

</p>

第41.10节：role="columnheader"

包含列头信息的单元格。

```
<table role="grid">
  <thead>
    <tr>
      <th role="columnheader">第1天</th>
      <th role="columnheader">第2天</th>
      <th role="columnheader">第3天</th>
    </tr>
  </thead>
  <tbody>
<!-- 等等 -->
  </tbody>
</table>
```

第41.11节：role="combobox"

选择框的呈现；通常类似于文本框，用户可以输入提示以选择一个选项，或者输入文本作为列表中的新项。

```
<input type="text" role="combobox" aria-expanded="false">
```

通常，您会使用JavaScript来构建其余的输入提示或列表选择功能。

第41.12节：role="complementary"

文档的辅助部分，设计为与DOM层级中主内容处于相似级别的补充内容，但即使与主内容分开也具有意义。

```
<div role="complementary">
  <h2>更多文章</h2>

  <ul>
<!-- 等等 -->
  </ul>
</div>
```

第41.13节：role="contentinfo"

包含有关父文档信息的大型可感知区域。

```
<p role="contentinfo">
  作者：阿尔伯特·爱因斯坦<br>
  发表时间：1940年8月15日
</p>
```

第41.14节：role="definition"

术语或概念的定义。

```
<span role="term" aria-labelledby="def1">爱</span>
```

</p>

Section 41.10: role="columnheader"

A cell containing header information for a column.

```
<table role="grid">
  <thead>
    <tr>
      <th role="columnheader">Day 1</th>
      <th role="columnheader">Day 2</th>
      <th role="columnheader">Day 3</th>
    </tr>
  </thead>
  <tbody>
  <!-- etc -->
  </tbody>
</table>
```

Section 41.11: role="combobox"

A presentation of a select; usually similar to a textbox where users can type ahead to select an option, or type to enter arbitrary text as a new item in the list.

```
<input type="text" role="combobox" aria-expanded="false">
```

Typically, you would use JavaScript to build the rest of the typeahead or list select functionality.

Section 41.12: role="complementary"

A supporting section of the document, designed to be complementary to the main content at a similar level in the DOM hierarchy, but remains meaningful when separated from the main content.

```
<div role="complementary">
  <h2>More Articles</h2>

  <ul>
  <!-- etc -->
  </ul>
</div>
```

Section 41.13: role="contentinfo"

A large perceivable region that contains information about the parent document.

```
<p role="contentinfo">
  Author: Albert Einstein<br>
  Published: August 15, 1940
</p>
```

Section 41.14: role="definition"

A definition of a term or concept.

```
<span role="term" aria-labelledby="def1">Love</span>
```

一种强烈的深厚感情。

第41.15节：role="dialog"

对话框是一个应用窗口，设计用于中断当前应用的处理，以提示用户输入信息或要求响应。

```
<div role="dialog">
  <p>你确定吗？</p>
  <button role="button">是</button>
  <button role="button">否</button>
</div>
```

第41.16节：role="directory"

指向某个组成员的引用列表，例如静态目录表。

```
<ul role="directory">
  <li><a href="/chapter-1">第1章</a></li>
  <li><a href="/chapter-2">第2章</a></li>
  <li><a href="/chapter-3">第3章</a></li>
</ul>
```

第41.17节：role="document"

包含相关信息的区域，被声明为文档内容，而非网页应用。

```
<div role="document">
  <h1>阿尔伯特·爱因斯坦的一生</h1>
  <p>Lorem ipsum...</p>
</div>
```

第41.18节：role="form"

一个地标区域，包含一组项目和对象，整体组合形成一个表单。

使用语义正确的HTML元素<form>意味着默认的ARIA语义，这表示不需要使用role=form，因为不应对已经具有语义的元素应用相反的角色，添加角色会覆盖元素的原生语义。

设置与默认隐式ARIA语义匹配的ARIA角色和/或aria-*属性是不必要且不推荐，因为这些属性已经由浏览器设置。

```
<form action="">
  <fieldset>
    <legend>登录表单</legend>
    <div>
      <label for="username">您的用户名</label>
      <input type="text" id="username" aria-describedby="username-tip" required />
      <div role="tooltip" id="username-tip">您的用户名是您的电子邮件地址</div>
    </div>
    <div>
      <label for="password">您的密码</label>
      <input type="text" id="password" aria-describedby="password-tip" required />
    </div>
  </fieldset>
</form>
```

```
<span id="def1" role="definition">an intense feeling of deep affection.</span>
```

Section 41.15: role="dialog"

A dialog is an application window that is designed to interrupt the current processing of an application in order to prompt the user to enter information or require a response.

```
<div role="dialog">
  <p>Are you sure?</p>
  <button role="button">Yes</button>
  <button role="button">No</button>
</div>
```

Section 41.16: role="directory"

A list of references to members of a group, such as a static table of contents.

```
<ul role="directory">
  <li><a href="/chapter-1">Chapter 1</a></li>
  <li><a href="/chapter-2">Chapter 2</a></li>
  <li><a href="/chapter-3">Chapter 3</a></li>
</ul>
```

Section 41.17: role="document"

A region containing related information that is declared as document content, as opposed to a web application.

```
<div role="document">
  <h1>The Life of Albert Einstein</h1>
  <p>Lorem ipsum...</p>
</div>
```

Section 41.18: role="form"

A landmark region that contains a collection of items and objects that, as a whole, combine to create a form.

Using the semantically correct HTML element <form> implies default ARIA semantics, meaning role=form is not required as you should not apply a contrasting role to an element that is already semantic, as adding a role overrides the native semantics of an element.

Setting an ARIA role and/or aria-* attribute that matches the default implicit ARIA semantics is unnecessary and is not recommended as these properties are already set by the browser.

```
<form action="">
  <fieldset>
    <legend>Login form</legend>
    <div>
      <label for="username">Your username</label>
      <input type="text" id="username" aria-describedby="username-tip" required />
      <div role="tooltip" id="username-tip">Your username is your email address</div>
    </div>
    <div>
      <label for="password">Your password</label>
      <input type="text" id="password" aria-describedby="password-tip" required />
    </div>
  </fieldset>
</form>
```

```

    <div role="tooltip" id="password-tip">注册时通过电子邮件发送给您的</div>
  </div>
</fieldset>
</form>
```

您会在非语义元素上使用role=form（不推荐，且无效）

```
<div role=表单>
  <input type="email" placeholder="您的电子邮件地址">
  <button>注册</button>
</div>
```

第41.19节：role="grid"

网格是一种交互控件，包含按行和列排列的表格数据单元格，类似于表格。

```
<table role="grid">
  <thead>
<!-- 等等 -->
  </thead>
  <tbody>
<!-- 等等 -->
  </tbody>
</table>
```

第41.20节：role="gridcell"

网格或树状网格中的单元格。

```
<table role="grid">
  <thead>
<!-- 等等 -->
  </thead>
  <tbody>
    <tr>
      <td role="gridcell">17</td>
      <td role="gridcell">64</td>
      <td role="gridcell">18</td>
    </tr>
  </tbody>
</table>
```

第41.21节：role="group"

一组用户界面对象，辅助技术不打算将其包含在页面摘要或目录中。

```
<div role="group">
  <button role"button">上一页</button>
  <button role"button">下一页</button>
</div>
```

第41.22节：role="heading"

页面某部分的标题。

```

    <div role="tooltip" id="password-tip">Was emailed to you when you signed up</div>
  </div>
</fieldset>
</form>
```

You would use role=form on non-semantic elements (not recommended, invalid)

```
<div role=form>
  <input type="email" placeholder="Your email address">
  <button>Sign up</button>
</div>
```

Section 41.19: role="grid"

A grid is an interactive control which contains cells of tabular data arranged in rows and columns, like a table.

```
<table role="grid">
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <!-- etc -->
  </tbody>
</table>
```

Section 41.20: role="gridcell"

A cell in a grid or treegrid.

```
<table role="grid">
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <tr>
      <td role="gridcell">17</td>
      <td role="gridcell">64</td>
      <td role="gridcell">18</td>
    </tr>
  </tbody>
</table>
```

Section 41.21: role="group"

A set of user interface objects which are not intended to be included in a page summary or table of contents by assistive technologies.

```
<div role="group">
  <button role"button">Previous</button>
  <button role"button">Next</button>
</div>
```

Section 41.22: role="heading"

A heading for a section of the page.


```
<h1 role="heading">介绍</h1>
<p>Lorem ipsum...</p>
```

第41.23节：role="img"

包含构成图像的一组元素的容器。

```
<figure role="img">
  
  <figcaption>这是我的猫，阿尔伯特。</figcaption>
</figure>
```

第41.24节：role="link"

指向内部或外部资源的交互式引用，激活时会使用户代理导航至该资源。

在大多数情况下，设置与默认隐式ARIA语义匹配的ARIA角色和/或aria-*属性是不必要且不推荐的，因为这些属性已经由浏览器设置。

来源 - <https://www.w3.org/TR/html5/dom.html#aria-usage-note>

第41.25节：role="list"

一组非交互式列表项。

```
<ul role="list">
  <li role="listitem">一</li>
  <li role="listitem">二</li>
  <li role="listitem">三</li>
</ul>
```

第41.26节：role="listbox"

允许用户从选项列表选择一个或多个项目的小部件。

```
<ul role="listbox">
  <li>一</li>
  <li>二</li>
  <li>三</li>
</ul>
```

通常，您会使用JavaScript来构建多选功能。

第41.27节：role="listitem"

列表或目录中的单个项目。

```
<ul role="list">
  <li role="listitem">一</li>
  <li role="listitem">二</li>
  <li role="listitem">三</li>
</ul>
```

```
<h1 role="heading">Introduction</h1>
<p>Lorem ipsum...</p>
```

Section 41.23: role="img"

A container for a collection of elements that form an image.

```
<figure role="img">
  
  <figcaption>This is my cat, Albert.</figcaption>
</figure>
```

Section 41.24: role="link"

An interactive reference to an internal or external resource that, when activated, causes the user agent to navigate to that resource.

In the majority of cases setting an ARIA role and/or aria-* attribute that matches the [default implicit ARIA semantics](#) is unnecessary and not recommended as these properties are already set by the browser.

Source - <https://www.w3.org/TR/html5/dom.html#aria-usage-note>

Section 41.25: role="list"

A group of non-interactive list items.

```
<ul role="list">
  <li role="listitem">One</li>
  <li role="listitem">Two</li>
  <li role="listitem">Three</li>
</ul>
```

Section 41.26: role="listbox"

A widget that allows the user to select one or more items from a list of choices.

```
<ul role="listbox">
  <li>One</li>
  <li>Two</li>
  <li>Three</li>
</ul>
```

Typically, you would use JavaScript to build the multiple-selection functionality.

Section 41.27: role="listitem"

A single item in a list or directory.

```
<ul role="list">
  <li role="listitem">One</li>
  <li role="listitem">Two</li>
  <li role="listitem">Three</li>
</ul>
```

第41.28节：role="log"

一种实时区域，其中新信息按有意义的顺序添加，旧信息可能会消失。

```
<ul role="log">
  <li>用户1已登录。</li>
  <li>用户2已登录。</li>
  <li>用户1已登出。</li>
</ul>
```

第41.29节：role="main"

文档的主要内容。

```
<!-- header & nav here -->
<div role="main">
  <p>Lorem ipsum...</p>
</div>
<!-- footer here -->
```

第41.30节：role="marquee"

一种实时区域，其中非必要信息频繁变化。

```
<ul role="marquee">
  <li>道琼斯指数 +0.26%</li>
  <li>纳斯达克指数 +0.54%</li>
  <li>标准普尔指数 +0.44%</li>
</ul>
```

第41.31节：role="math"

表示数学表达式的内容。

```

```

第41.32节：role="menu"

一种为用户提供选择列表的小部件类型。

```
<ul role="menu">
  <li role="menuitem">新建</li>
  <li role="menuitem">打开</li>
  <li role="menuitem">保存</li>
  <li role="menuitem">关闭</li>
</ul>
```

第41.33节：role="menubar"

菜单的呈现方式，通常保持可见，且通常水平展示。

```
<ul role="menubar">
  <li role="menuitem">文件</li>
  <li role="menuitem">编辑</li>
  <li role="menuitem">视图</li>
```

Section 41.28: role="log"

A type of live region where new information is added in meaningful order and old information may disappear.

```
<ul role="log">
  <li>User 1 logged in.</li>
  <li>User 2 logged in.</li>
  <li>User 1 logged out.</li>
</ul>
```

Section 41.29: role="main"

The main content of a document.

```
<!-- header & nav here -->
<div role="main">
  <p>Lorem ipsum...</p>
</div>
<!-- footer here -->
```

Section 41.30: role="marquee"

A type of live region where non-essential information changes frequently.

```
<ul role="marquee">
  <li>Dow +0.26%</li>
  <li>Nasdaq +0.54%</li>
  <li>S&P +0.44%</li>
</ul>
```

Section 41.31: role="math"

Content that represents a mathematical expression.

```

```

Section 41.32: role="menu"

A type of widget that offers a list of choices to the user.

```
<ul role="menu">
  <li role="menuitem">New</li>
  <li role="menuitem">Open</li>
  <li role="menuitem">Save</li>
  <li role="menuitem">Close</li>
</ul>
```

Section 41.33: role="menubar"

A presentation of menu that usually remains visible and is usually presented horizontally.

```
<ul role="menubar">
  <li role="menuitem">File</li>
  <li role="menuitem">Edit</li>
  <li role="menuitem">View</li>
```

```
<li role="menuitem">帮助</li>
</ul>
```

第41.34节：role="menuitem"

菜单或菜单栏中包含的一组选项中的一个选项。

```
<ul role="menubar">
  <li role="menuitem">文件</li>
  <li role="menuitem">编辑</li>
  <li role="menuitem">视图</li>
  <li role="menuitem">帮助</li>
</ul>
```

第41.35节：role="menuitemcheckbox"

一个可勾选的菜单项，具有三种可能的值：true、false 或 mixed。

```
<ul role="menu">
  <li role="menuitem">控制台</li>
  <li role="menuitem">布局</li>
  <li role="menuitemcheckbox" aria-checked="true">自动换行</li>
</ul>
```

第41.36节：role="menuitemradio"

一组 menuitemradio 角色中的可选中菜单项，其中一次只能选中一个。

```
<ul role="menu">
  <li role="menuitemradio" aria-checked="true">左对齐</li>
  <li role="menuitemradio" aria-checked="false">居中</li>
  <li role="menuitemradio" aria-checked="false">右对齐</li>
</ul>
```

第41.37节：role="navigation"

一组导航元素（通常是链接），用于导航文档或相关文档。

```
<ul role="navigation">
  <li><a href="/">首页</a></li>
  <li><a href="/about">关于</a></li>
  <li><a href="/contact">联系</a></li>
</ul>
```

第41.38节：role="note"

内容为附属或补充主资源内容的部分。

```
<p>Lorem ipsum...</p>
<p>Lorem ipsum...</p>
<p role="note">Lorem ipsum...</p>
```

第41.39节：role="option"

选择列表中的可选项。

```
<li role="menuitem">Help</li>
</ul>
```

Section 41.34: role="menuitem"

An option in a group of choices contained by a menu or menubar.

```
<ul role="menubar">
  <li role="menuitem">File</li>
  <li role="menuitem">Edit</li>
  <li role="menuitem">View</li>
  <li role="menuitem">Help</li>
</ul>
```

Section 41.35: role="menuitemcheckbox"

A checkable menuitem that has three possible values: true, false, or mixed.

```
<ul role="menu">
  <li role="menuitem">Console</li>
  <li role="menuitem">Layout</li>
  <li role="menuitemcheckbox" aria-checked="true">Word wrap</li>
</ul>
```

Section 41.36: role="menuitemradio"

A checkable menuitem in a group of menuitemradio roles, only one of which can be checked at a time.

```
<ul role="menu">
  <li role="menuitemradio" aria-checked="true">Left</li>
  <li role="menuitemradio" aria-checked="false">Center</li>
  <li role="menuitemradio" aria-checked="false">Right</li>
</ul>
```

Section 41.37: role="navigation"

A collection of navigational elements (usually links) for navigating the document or related documents.

```
<ul role="navigation">
  <li><a href="/">Home</a></li>
  <li><a href="/about">About</a></li>
  <li><a href="/contact">Contact</a></li>
</ul>
```

Section 41.38: role="note"

A section whose content is parenthetic or ancillary to the main content of the resource.

```
<p>Lorem ipsum...</p>
<p>Lorem ipsum...</p>
<p role="note">Lorem ipsum...</p>
```

Section 41.39: role="option"

A selectable item in a select list.

```
<ul role="listbox">
  <li role="option">选项1</li>
  <li role="option">选项2</li>
  <li role="option">选项3</li>
</ul>
```

第41.40节：role="progressbar"

显示长时间任务进度状态的元素。

```
<progress role="progressbar" value="25" max="100">25%</progress>
```

第41.41节：role="radio"

一组单选按钮中的可选输入，一次只能选中其中一个。

```
<div role="radiogroup">
  <input role="radio" type="radio" aria-checked="true"> 一<br>
  <input role="radio" type="radio" aria-checked="false"> 二<br>
  <input role="radio" type="radio" aria-checked="false"> 三
</div>
```

第41.42节：role="region"

网页或文档中一个较大的可感知区域，作者认为该区域足够重要，应包含在页面摘要或目录中，例如包含实时体育赛事统计数据的页面区域。

```
<div role="region">
  主队：4<br>
  客队：2
</div>
```

第41.43节：role="radiogroup"

一组单选按钮。

```
<div role="radiogroup">
  <input role="radio" type="radio" aria-checked="true"> 一<br>
  <input role="radio" type="radio" aria-checked="false"> 二<br>
  <input role="radio" type="radio" aria-checked="false"> 三
</div>
```

第41.44节：role="row"

表格容器中的一行单元格。

```
<table>
  <thead>
<!-- 等等 -->
  </thead>
  <tbody>
    <tr role="row">
      <!-- 等等 -->
    </tr>
  </tbody>
```

```
<ul role="listbox">
  <li role="option">Option 1</li>
  <li role="option">Option 2</li>
  <li role="option">Option 3</li>
</ul>
```

Section 41.40: role="progressbar"

An element that displays the progress status for tasks that take a long time.

```
<progress role="progressbar" value="25" max="100">25%</progress>
```

Section 41.41: role="radio"

A checkable input in a group of radio roles, only one of which can be checked at a time.

```
<div role="radiogroup">
  <input role="radio" type="radio" aria-checked="true"> One<br>
  <input role="radio" type="radio" aria-checked="false"> Two<br>
  <input role="radio" type="radio" aria-checked="false"> Three
</div>
```

Section 41.42: role="region"

A large perceivable section of a web page or document, that the author feels is important enough to be included in a page summary or table of contents, for example, an area of the page containing live sporting event statistics.

```
<div role="region">
  Home team: 4<br>
  Away team: 2
</div>
```

Section 41.43: role="radiogroup"

A group of radio buttons.

```
<div role="radiogroup">
  <input role="radio" type="radio" aria-checked="true"> One<br>
  <input role="radio" type="radio" aria-checked="false"> Two<br>
  <input role="radio" type="radio" aria-checked="false"> Three
</div>
```

Section 41.44: role="row"

A row of cells in a tabular container.

```
<table>
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <tr role="row">
      <!-- etc -->
    </tr>
  </tbody>
```

```
</table>
```

第41.45节：role="rowgroup"

包含网格中一个或多个行元素的组。

```
<table>
  <thead role="rowgroup">
    <!-- 等等 -->
  </thead>
  <tbody role="rowgroup">
    <!-- 等等 -->
  </tbody>
</table>
```

第41.46节：role="rowheader"

包含网格中某行标题信息的单元格。

```
<table role="grid">
  <thead>
<!-- 等等 -->
  </thead>
  <tbody>
    <tr>
      <th role="rowheader">第1天</th>
      <td>65</td>
    </tr>
    <tr>
      <th role="rowheader">第2天</th>
      <td>74</td>
    </tr>
  </tbody>
</table>
```

第41.47节：role="scrollbar"

一个图形对象，用于控制查看区域内内容的滚动，无论内容是否完全显示在查看区域内。

```
<div id="content1">Lorem ipsum...</div>
<div
  role="scrollbar"
  aria-controls="content1"
  aria-orientation="vertical"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25">
  <div class="scrollhandle"></div>
</div>
```

第41.48节：role="search"

一个地标区域，包含一组项目和物体，整体结合形成一个搜索设施。

```
<div role="search">
```

```
</table>
```

Section 41.45: role="rowgroup"

A group containing one or more row elements in a grid.

```
<table>
  <thead role="rowgroup">
    <!-- etc -->
  </thead>
  <tbody role="rowgroup">
    <!-- etc -->
  </tbody>
</table>
```

Section 41.46: role="rowheader"

A cell containing header information for a row in a grid.

```
<table role="grid">
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <tr>
      <th role="rowheader">Day 1</th>
      <td>65</td>
    </tr>
    <tr>
      <th role="rowheader">Day 2</th>
      <td>74</td>
    </tr>
  </tbody>
</table>
```

Section 41.47: role="scrollbar"

A graphical object that controls the scrolling of content within a viewing area, regardless of whether the content is fully displayed within the viewing area.

```
<div id="content1">Lorem ipsum...</div>
<div
  role="scrollbar"
  aria-controls="content1"
  aria-orientation="vertical"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25">
  <div class="scrollhandle"></div>
</div>
```

Section 41.48: role="search"

A landmark region that contains a collection of items and objects that, as a whole, combine to create a search facility.

```
<div role="search">
```



```
<input role="searchbox" type="text">
<button role="button">搜索</button>
</div>
```

第41.49节：role="searchbox"

一种用于指定搜索条件的文本框类型。

```
<div role="search">
  <input role="searchbox" type="text">
  <button role="button">搜索</button>
</div>
```

第41.50节：role="separator"

用于分隔和区分内容部分或菜单项组的分隔符。

```
<p>Lorem ipsum...</p>
<hr role="separator">
<p>Lorem ipsum...</p>
```

第41.51节：role="slider"

用户输入控件，用户可从给定范围内选择一个值。

```
<div
  role="slider"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25">
  <div class="sliderhandle"></div>
</div>
```

第41.52节：role="spinbutton"

一种范围形式，期望用户从离散选项中进行选择。

```
<input
  role="spinbutton"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25"
  type="number"
  value="25">
```

第41.53节：role="status"

一个容器，其内容是给用户的建议信息，但重要性不足以触发警报，通常但不一定以状态栏形式呈现。

```
<div role="status">在线</div>
```

第41.54节：role="switch"

一种表示开/关值的复选框类型，而不是选中/未选中值。

```
<input role="searchbox" type="text">
<button role="button">Search</button>
</div>
```

Section 41.49: role="searchbox"

A type of textbox intended for specifying search criteria.

```
<div role="search">
  <input role="searchbox" type="text">
  <button role="button">Search</button>
</div>
```

Section 41.50: role="separator"

A divider that separates and distinguishes sections of content or groups of menuitems.

```
<p>Lorem ipsum...</p>
<hr role="separator">
<p>Lorem ipsum...</p>
```

Section 41.51: role="slider"

A user input where the user selects a value from within a given range.

```
<div
  role="slider"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25">
  <div class="sliderhandle"></div>
</div>
```

Section 41.52: role="spinbutton"

A form of range that expects the user to select from among discrete choices.

```
<input
  role="spinbutton"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25"
  type="number"
  value="25">
```

Section 41.53: role="status"

A container whose content is advisory information for the user but is not important enough to justify an alert, often but not necessarily presented as a status bar.

```
<div role="status">Online</div>
```

Section 41.54: role="switch"

A type of checkbox that represents on/off values, as opposed to checked/unchecked values.

```
<select role="switch" aria-checked="false">
  <option>开</option>
  <option selected>关</option>
</select>
```

第41.55节：role="tab"

一个分组标签，提供选择要呈现给用户的标签内容的机制。

```
<ul role="tablist">
  <li role="tab">介绍</li>
  <li role="tab">第1章</li>
  <li role="tab">第2章</li>
</ul>
```

第41.56节：role="table"

包含按行和列排列的数据的部分。table角色用于非交互式的表格容器。

```
<table role="table">
  <thead>
<!-- 等等 -->
  </thead>
  <tbody>
<!-- 等等 -->
  </tbody>
</table>
```

第41.57节：role="tablist"

标签元素列表，这些元素是对tabpanel元素的引用。

```
<ul role="tablist">
  <li role="tab">介绍</li>
  <li role="tab">第1章</li>
  <li role="tab">第2章</li>
</ul>
```

第41.58节：role="tabpanel"

与标签相关联的资源容器，每个标签包含在一个tablist中。

```
<ul role="tablist">
  <li role="tab">介绍</li>
  <li role="tab">第1章</li>
  <li role="tab">第2章</li>
</ul>
<div role="tabpanel">
  <!-- 等等 -->
</div>
```

第41.59节：role="textbox"

允许自由格式文本作为其值的输入。

```
<select role="switch" aria-checked="false">
  <option>On</option>
  <option selected>Off</option>
</select>
```

Section 41.55: role="tab"

A grouping label providing a mechanism for selecting the tab content that is to be rendered to the user.

```
<ul role="tablist">
  <li role="tab">Introduction</li>
  <li role="tab">Chapter 1</li>
  <li role="tab">Chapter 2</li>
</ul>
```

Section 41.56: role="table"

A section containing data arranged in rows and columns. The table role is intended for tabular containers which are not interactive.

```
<table role="table">
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <!-- etc -->
  </tbody>
</table>
```

Section 41.57: role="tablist"

A list of tab elements, which are references to tabpanel elements.

```
<ul role="tablist">
  <li role="tab">Introduction</li>
  <li role="tab">Chapter 1</li>
  <li role="tab">Chapter 2</li>
</ul>
```

Section 41.58: role="tabpanel"

A container for the resources associated with a tab, where each tab is contained in a tablist.

```
<ul role="tablist">
  <li role="tab">Introduction</li>
  <li role="tab">Chapter 1</li>
  <li role="tab">Chapter 2</li>
</ul>
<div role="tabpanel">
  <!-- etc -->
</div>
```

Section 41.59: role="textbox"

Input that allows free-form text as its value.

```
<textarea role="textbox"></textarea>
```

第41.60节：role="timer"

一种包含数字计数器的实时区域，指示从起点经过的时间量，或直到终点剩余的时间。

```
<p>
  <span role="timer">60</span> 秒剩余。
</p>
```

第41.61节：role="toolbar"

一组以紧凑视觉形式表示的常用功能按钮集合。

```
<ul role="toolbar">
  <li></li>
  <li></li>
  <li></li>
  <li></li>
</ul>
```

第41.62节：role="tooltip"

显示元素描述的上下文弹出框。

```
<span aria-describedby="slopedesc">斜率</span>
<div role="tooltip" id="slopedesc">y=mx+b</div>
```

通常，工具提示会被隐藏。使用JavaScript，当用户悬停在其描述的元素上时，工具提示会在延迟后显示。

第41.63节：role="tree"

一种列表类型，可能包含可折叠和展开的子级嵌套组。

```
<ul role="tree">
  <li role="treeitem">
    第1部分
    <ul>
      <li role="treeitem">第1章</li>
      <li role="treeitem">第2章</li>
      <li role="treeitem">第3章</li>
    </ul>
  </li>
  <li role="treeitem">
    第二部分
    <ul>
      <li role="treeitem">第4章</li>
      <li role="treeitem">第5章</li>
      <li role="treeitem">第6章</li>
    </ul>
  </li>
  <li role="treeitem">
    第三部分
    <ul>
```

```
<textarea role="textbox"></textarea>
```

Section 41.60: role="timer"

A type of live region containing a numerical counter which indicates an amount of elapsed time from a start point, or the time remaining until an end point.

```
<p>
  <span role="timer">60</span> seconds remaining.
</p>
```

Section 41.61: role="toolbar"

A collection of commonly used function buttons represented in compact visual form.

```
<ul role="toolbar">
  <li></li>
  <li></li>
  <li></li>
  <li></li>
</ul>
```

Section 41.62: role="tooltip"

A contextual popup that displays a description for an element.

```
<span aria-describedby="slopedesc">Slope</span>
<div role="tooltip" id="slopedesc">y=mx+b</div>
```

Typically, the tooltip would be hidden. Using JavaScript, the tooltip would be displayed after a delay when the user hovers over the element that it describes.

Section 41.63: role="tree"

A type of list that may contain sub-level nested groups that can be collapsed and expanded.

```
<ul role="tree">
  <li role="treeitem">
    Part 1
    <ul>
      <li role="treeitem">Chapter 1</li>
      <li role="treeitem">Chapter 2</li>
      <li role="treeitem">Chapter 3</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 2
    <ul>
      <li role="treeitem">Chapter 4</li>
      <li role="treeitem">Chapter 5</li>
      <li role="treeitem">Chapter 6</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 3
    <ul>
```

```
<li role="treeitem">第7章</li>
<li role="treeitem">第8章</li>
<li role="treeitem">第9章</li>
</ul>
</li>
</ul>
```

第41.64节：role="treegrid"

一种网格，其行可以像树形结构一样展开和折叠。

第41.65节：role="treeitem"

树的选项项。如果包含子级树项组，则这是树中的一个元素，可以展开或折叠。

```
<ul role="tree">
  <li role="treeitem">
    第1部分
    <ul>
      <li role="treeitem">第1章</li>
      <li role="treeitem">第2章</li>
      <li role="treeitem">第3章</li>
    </ul>
  </li>
  <li role="treeitem">
    第二部分
    <ul>
      <li role="treeitem">第4章</li>
      <li role="treeitem">第5章</li>
      <li role="treeitem">第6章</li>
    </ul>
  </li>
  <li role="treeitem">
    第三部分
    <ul>
      <li role="treeitem">第7章</li>
      <li role="treeitem">第8章</li>
      <li role="treeitem">第9章</li>
    </ul>
  </li>
</ul>
```

```
<li role="treeitem">Chapter 7</li>
<li role="treeitem">Chapter 8</li>
<li role="treeitem">Chapter 9</li>
</ul>
</li>
</ul>
```

Section 41.64: role="treegrid"

A grid whose rows can be expanded and collapsed in the same manner as for a tree.

Section 41.65: role="treeitem"

An option item of a tree. This is an element within a tree that may be expanded or collapsed if it contains a sub-level group of treeitems.

```
<ul role="tree">
  <li role="treeitem">
    Part 1
    <ul>
      <li role="treeitem">Chapter 1</li>
      <li role="treeitem">Chapter 2</li>
      <li role="treeitem">Chapter 3</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 2
    <ul>
      <li role="treeitem">Chapter 4</li>
      <li role="treeitem">Chapter 5</li>
      <li role="treeitem">Chapter 6</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 3
    <ul>
      <li role="treeitem">Chapter 7</li>
      <li role="treeitem">Chapter 8</li>
      <li role="treeitem">Chapter 9</li>
    </ul>
  </li>
</ul>
```

学分

非常感谢所有来自Stack Overflow Documentation的人员提供此内容，
更多更改可发送至web@petercv.com以发布或更新新内容

AA2992	第12章
阿比谢克·潘迪	第17章和第33章
阿布拉尔·贾欣	第4章
阿吉特	第29章
ahmednawazbutt	第1章
Al.G.	第2章、第6章和第17章
阿尔伯特	第8章
亚历克斯	第15章
亚历山大·威格莫尔	第33章
亚历山大·N.	第8、13、15、28和29章
阿里·阿尔穆利姆	第18章和第27章
阿曼达·安	第1章
amflare	第1章和第27章
阿米泰·斯特恩	第1章和第17章
andreaem	第13章
andreas	第31章
安德鲁·布鲁克	第20章
安吉洛斯·查拉里斯	第2、10、17和33章
阿尼·梅农	第2、3、9、17、18和33章
阿尼尔	第20章
animuson	第1、2、5、6、7、8、9、15、17、18、27、29、33、37和40章
安瑟姆	第6章、第12章和第13章
安东尼·范	第6章
阿翁·穆罕默德	第18章
阿拉克尼德	第19章
阿拉文德·苏雷什	第12章
阿什温·拉马斯瓦米	第9章
斧头	第33章
巴林特	第33章
ban17	第6章
bdkopen	第9章、第14章和第33章
贝卡	第37章
本·里斯	第5章
bhansa	第17章
巴维亚·辛格	第41章
书食者	第33章
鲍里斯	第33章
博伊森贝里	第15章
布兰达蒙	第10章
卡勒布·克莱维特	第10章和第15章
卡兰·赫德	第1章和第12章
塞德里克·佐波洛	第8章
查尔斯	第13章
克里斯	第2章和第19章
克里斯·鲁瑟福德	第12章
克里斯D	第1章、第9章、第24章、第29章和第31章
克里斯蒂安·特努斯	第14章

Credits

Thank you greatly to all the people from Stack Overflow Documentation who helped provide this content,
more changes can be sent to web@petercv.com for new content to be published or updated

AA2992	Chapter 12
Abhishek Pandey	Chapters 17 and 33
Abrar Jahin	Chapter 4
Adjit	Chapter 29
ahmednawazbutt	Chapter 1
Al.G.	Chapters 2, 6 and 17
albert	Chapter 8
Alex	Chapter 15
Alexander Wigmore	Chapter 33
Alexandre N.	Chapters 8, 13, 15, 28 and 29
Ali Almoullim	Chapters 18 and 27
Amanda Ahn	Chapter 1
amflare	Chapters 1 and 27
Amitay Stern	Chapters 1 and 17
andreaem	Chapter 13
andreas	Chapter 31
Andrew Brooke	Chapter 20
Angelos Chalaris	Chapters 2, 10, 17 and 33
Ani Menon	Chapters 2, 3, 9, 17, 18 and 33
Anil	Chapter 20
animuson	Chapters 1, 2, 5, 6, 7, 8, 9, 15, 17, 18, 27, 29, 33, 37 and 40
Anselm	Chapters 6, 12 and 13
Anthony Pham	Chapter 6
Aown Muhammad	Chapter 18
Araknid	Chapter 19
Aravind Suresh	Chapter 12
Ashwin Ramaswami	Chapter 9
Axe	Chapter 33
Bálint	Chapter 33
ban17	Chapter 6
bdkopen	Chapters 9, 14 and 33
Becca	Chapter 37
Ben Rhys	Chapter 5
bhansa	Chapter 17
Bhavya Singh	Chapter 41
Bookeater	Chapter 33
Boris	Chapter 33
Boysenb3rry	Chapter 15
brandaemon	Chapter 10
Caleb Kleveter	Chapters 10 and 15
Callan Heard	Chapters 1 and 12
Cedric Zoppolo	Chapter 8
Charles	Chapter 13
Chris	Chapters 2 and 19
Chris Rutherfurd	Chapter 12
ChrisD	Chapters 1, 9, 24, 29 and 31
Christian Ternus	Chapter 14

克里斯托夫·斯特罗贝	第17章
cone56	第12章和第32章
内容解决方案	第2章、第19章、第26章、第35章和第36章
Cullub	第18章
C_____yN	第17章
D M	第19章
丹尼尔	第17章
丹尼尔·凯弗	第7章
戴夫·埃弗里特	第1章
黎明圣骑士	第6、12和17章
迪彭·沙阿	第17章
dippas	第3章
多梅尼克	第12和33章
德鲁齐翁	第17章
埃马纽埃尔·温蒂尔ă	第29章
埃米尔	第5、6和10章
埃文	第3章
法哈德	第38章
feeela	第12和25章
费利佩·阿尔斯	第30章
FlyingPiMonster	第6章
gabe3886	第17章
李志康	第15章和第18章
加尔·拉特斯金	第19章
geek1011	第1章和第33章
geeksal	第17章
盖罗尔德·布罗泽	第1章和第30章
GoatsWearHats	第10章
格蕾丝·诺特	第7章
格兰特·帕林	第8章
古斯塔沃·亨克	第5章
吉蒂斯·特诺维马斯	第17章
H·米尔扎	第1章
H·保韦林	第1章
哈贝尔·菲利普	第33章
哈菲兹·伊尔哈姆·阿吉·佩尔马纳	第33章
哈尼夫·福莫利	第20章
赫尔夫克	第2章
亨里克·巴塞洛斯	第12章和第17章
阮辉	第17章
Infuzion	第3章和第12章
insertusernamehere	第6章
intboolstring	第1章
伊萨克·康布林克	第25章和第30章
ivn	第15章
J F	第5章和第8章
j08691	第8章和第18章
jhnance	第10章
jhoanna	第33章
JHS	第6章、第8章、第9章、第12章和第19章
jkdev	第9章
joe_young	第1章、第3章和第8章
约翰·斯莱格斯	第10章和第17章

Christophe Strobbe	Chapter 17
cone56	Chapters 12 and 32
Content Solutions	Chapters 2, 19, 26, 35 and 36
Cullub	Chapter 18
C□□□□yN□□□□	Chapter 17
D M	Chapter 19
Daniel	Chapter 17
Daniel Käfer	Chapter 7
Dave Everitt	Chapter 1
DawnPaladin	Chapters 6, 12 and 17
Dipen Shah	Chapter 17
dippas	Chapter 3
Domenic	Chapters 12 and 33
Druzion	Chapter 17
Emanuel Vintilă	Chapter 29
Emil	Chapters 5, 6 and 10
Evan	Chapter 3
Farhad	Chapter 38
feeela	Chapters 12 and 25
FelipeAls	Chapter 30
FlyingPiMonster	Chapter 6
gabe3886	Chapter 17
Gabriel Chi Hong Lee	Chapters 15 and 18
Gal Ratzkin	Chapter 19
geek1011	Chapters 1 and 33
geeksal	Chapter 17
Gerold Broser	Chapters 1 and 30
GoatsWearHats	Chapter 10
Grace Note	Chapter 7
Grant Palin	Chapter 8
gustavohenke	Chapter 5
Gytis Tenovimas	Chapter 17
H Mirza	Chapter 1
H. Pauwelyn	Chapter 1
Habel Philip	Chapter 33
Hafidz Ilham Aji Permana	Chapter 33
Hanif Formoly	Chapter 20
heerfk	Chapter 2
Henrique Barcelos	Chapters 12 and 17
Huy Nguyen	Chapter 17
Infuzion	Chapters 3 and 12
insertusernamehere	Chapter 6
intboolstring	Chapter 1
Isak Combrinck	Chapters 25 and 30
ivn	Chapter 15
J F	Chapters 5 and 8
j08691	Chapters 8 and 18
jhnance	Chapter 10
jhoanna	Chapter 33
JHS	Chapters 6, 8, 9, 12 and 19
jkdev	Chapter 9
joe_young	Chapters 1, 3 and 8
John Slegers	Chapters 10 and 17

Jojodmo	第6章
乔恩·埃里克森	第7章
乔纳森·蓝姆	第1章、第6章、第12章和第16章
琼斯·约瑟夫	第6章和第14章
Kake Fisk	第15章
kcpike	第7章
kelvinelove	第1章和第33章
Kimmax	第18章
拉希鲁·阿山	第17章
莱奥尼达斯·梅嫩德斯	第1章
卢卡·兰格拉	第18章
卢卡·普楚	第10章
玛乔丽·皮卡德	第13章
马文	第7章和第17章
马塔斯·瓦伊特凯维休斯	第6、7、12和17章
马特	第1章
马克西米利安·劳迈斯特 (Maximillian Laumeister)	第15和17章
默夫S	第13和40章
迈克尔·莫里亚蒂 (Michael Moriarty)	第33章
Michael B	第10章
mnoronha	第1章
莫哈德·萨米尔·汗 (Mohd Samir Khan)	第15章
morewry	第17章
莫什·费乌	第27章
莫蒂	第8章
利斯特先生	第8章
艾伦先生	第33章
蒙塔西尔	第30章
我的速度	第19章
m_callens	第22章和第33章
nalply	第20章
娜塔莉	第10章
内森·塔吉	第6章
南	第12章和第15章
尼克·布劳沃尔	第18章
Nijin22	第8章
尼尔·利斯特里	第17章
尼什查伊	第1章和第33章
新手程序员	第17章
奥仁	第29章
奥尔托马拉·洛克尼	第17章和第33章
帕雷什·马戈迪亚	第39章
保罗·斯威特	第41章
佩尔辛	第1章
彼得·L.	第5章
Pi程序	第17章
pinjasaur	第2章
platy11	第7章
普拉纳夫	第6章
普拉蒂克	第6、7、8和17章
普拉蒂克·洛查瓦拉	第8章
普拉文·库马尔	第8章
普萨尼科	第17和36章

Jojodmo	Chapter 6
Jon Ericson	Chapter 7
Jonathan Lam	Chapters 1, 6, 12 and 16
Jones Joseph	Chapters 6 and 14
Kake Fisk	Chapter 15
kcpike	Chapter 7
kelvinelove	Chapters 1 and 33
Kimmax	Chapter 18
Lahiru Ashan	Chapter 17
Leonidas Menendez	Chapter 1
Luca Iangella	Chapter 18
Luca Putzu	Chapter 10
Marjorie Pickard	Chapter 13
Marvin	Chapters 7 and 17
Matas Vaitkevicius	Chapters 6, 7, 12 and 17
Matt	Chapter 1
Maximillian Laumeister	Chapters 15 and 17
MervS	Chapters 13 and 40
Michael Moriarty	Chapter 33
Michael B	Chapter 10
mnoronha	Chapter 1
Mohd Samir Khan	Chapter 15
morewry	Chapter 17
Mosh Feu	Chapter 27
Mottie	Chapter 8
Mr Lister	Chapter 8
Mr. Alien	Chapter 33
Muntasir	Chapter 30
MySpeed	Chapter 19
m_callens	Chapters 22 and 33
nalply	Chapter 20
Natalie	Chapter 10
Nathan Tuggy	Chapter 6
Nhan	Chapters 12 and 15
Niek Brouwer	Chapter 18
Nijin22	Chapter 8
Nil Llisterri	Chapter 17
Nishchay	Chapters 1 and 33
NoobCoder	Chapter 17
Ojen	Chapter 29
Ortomala Lokni	Chapters 17 and 33
Paresh Maghodiya	Chapter 39
Paul Sweatte	Chapter 41
Persijn	Chapter 1
Peter L.	Chapter 5
Pi Programs	Chapter 17
pinjasaur	Chapter 2
platy11	Chapter 7
Pranav	Chapter 6
Prateek	Chapters 6, 7, 8 and 17
PrAtik Lochawala	Chapter 8
Praveen Kumar	Chapter 8
Psaniko	Chapters 17 and 36

化名帕特尔	第3和7章
拉西尔·希兰	第11章
rajarshig	第11章
拉面厨师	第16章和第17章
兰吉特·辛格	第17章
雷斯塔法里安	第5章和第6章
理查德·汉密尔顿	第7章、第10章、第26章和第32章
罗伯特·哥伦比亚	第6章
罗科·C·布尔扬	第13章、第17章、第22章和第32章
萨福尔·萨夫达尔	第33章
萨沙	第3章
SeinopSys	第17章
森朱蒂·马哈帕特拉	第33章
香农·杨	第20章和第41章
沙拉夫南·Kv	第17章
湿婆	第25章
希万吉·乔拉西亚	第15和17章
西尔多雷斯	第8章和第13章
西蒙·卡莱蒂	第10章
SJDS	第17章
栈	第40章
斯蒂芬·莱皮克	第17章和第33章
斯图尔特赛德	第7章和第17章
斯坦·范·埃斯费尔德	第6章
萨姆纳·埃文斯	第2章
桑尼·R·古普塔	第1章和第17章
超级风暴者	第13章、第19章和第34章
sv3k	第17章
svarog	第8章
泰德·戈阿斯	第8章和第33章
the12	第1章和第17章
think123	第17章
托马斯·格罗特	第24章
托马斯·兰道尔	第8章
thouusten	第14章
提蒙	第17章
蒂莫西·米勒	第31章
tmg	第2章、第7章和第17章
汤姆·约翰逊	第7章
tonethar	第32章
托特·扎姆	第17章和第27章
特拉维斯	第41章
特雷弗·克拉克	第32章
trungk18	第17章
泰勒·齐卡	第10章
乌里希·施瓦茨	第9章
撤销	第17章
unor	第1、5、6、7、12、30、33、34、35和37章
user3130333	第1章
user5389107	第17章
V4karian	第12章
瓦洛尔·纳拉姆	第1、4和38章
vkopio	第3章

Pseudonym Patel	Chapters 3 and 7
Racil Hilan	Chapter 11
rajarshig	Chapter 11
RamenChef	Chapters 16 and 17
Ranjit Singh	Chapter 17
Raystafarian	Chapters 5 and 6
Richard Hamilton	Chapters 7, 10, 26 and 32
Robert Columbia	Chapter 6
Roko C. Buljan	Chapters 13, 17, 22 and 32
Safoor Safdar	Chapter 33
sasha	Chapter 3
SeinopSys	Chapter 17
Senjuti Mahapatra	Chapter 33
Shannon Young	Chapters 20 and 41
Sharavnan Kv	Chapter 17
Shiva	Chapter 25
Shivangi Chaurasia	Chapters 15 and 17
Sildoreth	Chapters 8 and 13
Simone Carletti	Chapter 10
SJDS	Chapter 17
stack	Chapter 40
Stephen Leppik	Chapters 17 and 33
Stewartside	Chapters 7 and 17
Steyn van Esveld	Chapter 6
Sumner Evans	Chapter 2
Sunny R Gupta	Chapters 1 and 17
SuperStormer	Chapters 13, 19 and 34
sv3k	Chapter 17
svarog	Chapter 8
Ted Goas	Chapters 8 and 33
the12	Chapters 1 and 17
think123	Chapter 17
Thomas Gerot	Chapter 24
Thomas Landauer	Chapter 8
thouusten	Chapter 14
Timon	Chapter 17
Timothy Miller	Chapter 31
tmg	Chapters 2, 7 and 17
Tom Johnson	Chapter 7
tonethar	Chapter 32
Tot Zam	Chapters 17 and 27
Travis	Chapter 41
Trevor Clarke	Chapter 32
trungk18	Chapter 17
Tyler Zika	Chapter 10
Ulrich Schwarz	Chapter 9
Undo	Chapter 17
unor	Chapters 1, 5, 6, 7, 12, 30, 33, 34, 35 and 37
user3130333	Chapter 1
user5389107	Chapter 17
V4karian	Chapter 12
Valor Naram	Chapters 1, 4 and 38
vkopio	Chapter 3

[vladdobra](#)
[w5m](#)
[沃尔夫冈](#)
[xims](#)
[亚西尔·T](#)
[约西·阿哈龙](#)
[扎克](#)
[赞格](#)
[扎兹](#)
[zer00ne](#)
[齐吉曼图斯](#)
[Zze](#)
[zzzzBov](#)

第12章和第17章
第31章
第6、9和15章
第33章
第11章
第20章和第25章
第7章
第37章
第1章
第8章和第23章
第1章
第1章
第17章

[vladdobra](#)
[w5m](#)
[Wolfgang](#)
[xims](#)
[Yasir T](#)
[Yossi Aharon](#)
[Zack](#)
[Zange](#)
[Zaz](#)
[zer00ne](#)
[zygimantus](#)
[Zze](#)
[zzzzBov](#)

Chapters 12 and 17
Chapter 31
Chapters 6, 9 and 15
Chapter 33
Chapter 11
Chapters 20 and 25
Chapter 7
Chapter 37
Chapter 1
Chapters 8 and 23
Chapter 1
Chapter 1
Chapter 17

你可能也喜欢

