高等学校计算机应用规划教材

HTML+CSS+JavaScript 网页设计

夏魁良　王丽红　编著

清华大学出版社

北　京

内 容 简 介

本书系统全面地介绍 HTML、CSS 和 JavaScript 的基本知识和使用技巧。全书共分为 14 章，主要内容包括网页设计基础知识、HTML 基础、HTML5 快速入门、HTML 表单、网页中的多媒体、CSS 概述、CSS 选择器、使用 CSS 设置文本样式、设置元素的背景、边框和边距、变形处理、CSS 动画、网页布局、JavaScript 基本语法、JavaScript 函数、事件和对象、使用 jQuery 等内容。最后综合运用全书所学知识，介绍企业网站建设的基本流程和风格设计。

本书内容丰富、结构合理、思路清晰、语言简练流畅、示例翔实，主要面向网页设计与制作的学习人员，适合作为高等院校相关专业网页设计课程的教材，还可作为网页设计与开发从业人员的参考资料。

本书对应的课件、习题答案和实例源文件可以到 http://www.tupwk.com.cn/downpage 网站下载，也可通过扫描前言中的二维码下载。

图书在版编目(CIP)数据

HTML+CSS+JavaScript 网页设计 / 夏魁良，王丽红 编著. —北京：清华大学出版社，2019
(高等学校计算机应用规划教材)
ISBN 978-7-302-52523-3

Ⅰ. ①H…　Ⅱ. ①夏… ②王…　Ⅲ. ①超文本标记语言—程序设计—高等学校—教材 ②网页制作工具－高等学校－教材 ③JAVA 语言—程序设计—高等学校—教材　Ⅳ. ①TP312.8②TP393.092

中国版本图书馆 CIP 数据核字(2019)第 043606 号

责任编辑：胡辰浩
装帧设计：孔祥峰
责任校对：成凤进
责任印制：李红英

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：清华大学印刷厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**22　　**字　　数：**563 千字
版　　次：2019 年 4 月第 1 版　　**印　　次：**2019 年 4 月第 1 次印刷
定　　价：68.00 元

产品编号：057180-01

前　言

在“互联网+”时代，对各种网站的需求越来越多，规范性标准越来越高，技术越来越先进，传统的网站制作教材从技术实现的角度看，使用的技术比较落后；从代码结构看，没有将页面内容和样式分离，导致代码过于烦琐，不便于维护和扩展。为了适应现代技术的飞速发展，帮助众多网页制作爱好者学习标准的网页设计规范，提高网站的设计及编码水平，我们在潜心研究网站制作的前沿技术后，精心编写了本书。

本书采用全新的 Web 标准及技术，由浅入深、系统全面地介绍 HTML、CSS 和 JavaScript 的基本知识和常用技巧。这 3 项技术分别对应网页的 3 个主要部分：结构、表现和行为。HTML 是网页的结构，是网页制作的主要语言，作为网页内容的载体，HTML 包含用户需要浏览的内容，包括图文、视频，它们是构成网页的基本元素；CSS 用来设定网页的表现样式，中文名叫层叠样式表，它的出现是为了解决内容和表现分离的问题，CSS 的存在使得 HTML 变得丰富多样；如果只有“结构”和“表现”，而缺少用户与网页的交互，那么这样的网页就如一潭死水，无法使用户获得良好体验，JavaScript 的出现就是为了控制网页的行为，增强用户的可操作性，JavaScript 是脚本语言，是连接前台(HTML)和后台服务器的桥梁，更是操纵 HTML 的能手。

全书共 14 章，第 1 章介绍网页制作与设计相关的基础知识，主要讲述一些基本概念、相关技术、常用的开发工具等；第 2~5 章是 HTML 部分，主要介绍 HTML 的基本语法和常用标签的使用，HTML5 新增的标签、属性和事件，HTML 表单以及网页中的多媒体元素等；第 6~10 章是 CSS 部分，主要介绍为什么要使用 CSS，如何在 HTML 中使用 CSS，CSS 的继承和优先级，CSS 选择器，CSS 的常用属性(包括字体相关的属性、文本格式化属性、颜色与背景、边框和边距等)，变形处理和动画设计；第 11~13 章是 JavaScript 部分，主要介绍 JavaScript 的发展历程，文档对象模型(DOM)，JavaScript 的基本语法，JavaScript 函数、事件和对象，使用 jQuery(包括为什么使用 jQuery、jQuery 的基本语法、选择器、筛选器、事件处理、文档处理、jQuery 动画等)。JavaScript 本身是一个庞大的主题，本书虽然并未以 HTML 和 CSS 同样的深度进行介绍，但也能使读者编写自己的脚本并有效地使用 jQuery。第 14 章是综合实例，讲述典型的企业网站建设流程和风格设计，并通过具体的实例开发，引领读者学以致用，熟悉实际项目的开发流程，逐步成长为一名优秀的网页设计与开发人员。

本书内容丰富、结构合理、思路清晰、语言简练流畅、示例翔实。每一章的引言部分概述该章的内容和学习目标。在每一章的正文中，结合所讲述的关键技术和难点，穿插大量极富实用价值的示例，并配有相应的效果图。每一章末尾都安排了有针对性的思考和练习，帮助读者巩固该章所学的知识点，培养读者的实际动手能力，加深读者对关键技术和难点的理解。

本书主要面向网页制作与开发的学习人员，适合作为高等院校相关专业的教材，也适合从事网页设计制作和网站建设的从业人员阅读和参考。

本书分为14章，其中黑河学院的夏魁良编写了第1~7章，王丽红编写了第8~14章。另外，参加本书编写的人员还有肖茜、徐晓明、薛继军、岳殿召、陈添荣、侯铁国、刘军勇、李淑萍、尹志亮、陈光训、吴超群、郑玉祥、付君泽、黄怀春、靳廷喜等。由于作者水平有限，本书难免有不足之处，欢迎广大读者批评指正。我们的信箱是 huchenhao@263.net，电话是010-62796045。

本书对应的课件、习题答案和实例源文件可以到 http://www.tupwk.com.cn/downpage 网站下载，也可通过扫描下方的二维码下载。

作　者

2018 年 12 月

目　录

第 2 章

HTML基础

作为一种网页制作语言，HTML 有自己的语法规则，本章从 HTML 的历史变迁讲起，带领读者从基本的标签和元素开始，慢慢学习 HTML 的基本语法，尝试制作简单的网页。

本章的学习目标：

- 了解HTML的历史变迁
- 了解HTML与XHTML的关系
- 掌握HTML中标签和元素的基本概念
- 掌握 HTML5 的文档类型声明
- 掌握常用的文本格式化标签
- 掌握HTML中列表的创建与使用
- 掌握链接的创建和应用
- 掌握表格的创建与使用

2.1 HTML 简介

HTML 是目前互联网上应用最为广泛的语言，也是构成网页文档的主要语言。HTML 文档是由 HTML 标签组成的描述性文本，HTML 标签可以标识文字、图形、动画、声音、视频、表格、链接等。

2.1.1 HTML 的历史变迁

1982 年，美国人蒂姆 • 伯纳斯 • 李为了方便世界各地的物理学家能够进行合作研究以及信息共享，创造了 HTML 语言。1990 年，他又发明了世界上的第一个浏览器。随后的几年里，Netscape 和 Microsoft 两个软件巨头掀起了一场互联网浏览器大战。这场大战最后以 Microsoft 的 Internet Explorer 完胜告终，Internet Explorer 极大地推动了互联网的发展，把网络带到了千千万万普通用户面前。1993 年，互联网工程工作小组(Internet Engineering Tast Force，IETF)工作草案发布，可以算作 HTML 的第一个版本，但它却不是一个正式的版本。第一个正式版本 HTML 2.0 也不是出自 W3C 之手，而是由 IETF 制定的，从第三个版本开始，W3C 开始接手并负责后

续版本的制定工作。

HTML 3.0 规范由 W3C 于 1995 年 3 月提出，提供了很多新的特性，例如表格、文字绕排和复杂数学元素的显示。虽然它是被设计用来兼容 2.0 版本的，但是实现这个标准的工作在当时过于复杂，在草案于 1995 年 9 月过期时，标准开发也因为缺乏浏览器支持而中止了。3.1 版本从未被正式提出，而下一个被提出的版本是开发代号为 Wilbur 的 HTML 3.2，去掉了大部分 3.0 版本中的新特性，但是加入了很多特定浏览器(例如 Netscape 和 Mosaic)的元素和属性。

20 世纪 90 年代，HTML 有过几次快速发展。众所周知，那时构建网站是一项十分复杂的工程，从 1997 年到 1999 年，HTML 的版本从 3.2 到 4.0，再到 4.01，经历了非常快的发展。

问题是到了 4.01 版本的时候，W3C 的认识发生倒退，W3C 并没有停止开发这门语言，只不过他们对 HTML 不再感兴趣了。在 HTML 4.01 之后，W3C 提出了 XHTML 1.0 的概念。虽然听起来完全不同，但 XHTML 1.0 和 HTML 4.01 其实是一样的。唯一不同的是 XHTML 1.0 要求使用 XML 语法。

从规范本身的内容看，本质是相同的，不同之处在于编码风格，因为浏览器读取符合 HTML 4.01、HTML 3.2 或 XHTML 1.0 规范的网页都没有问题。对于浏览器来说这些网页都是一样的，都会生成相同的 DOM 树，只不过用户更喜欢 XHTML 1.0，因为不少人认同它比较严格的编码风格。

到了 2000 年，Web 标准项目的活动开展得如火如荼，开发人员对浏览器里包含的那些乱七八糟的专有特性已经忍无可忍了。当时 CSS 有了长足的发展，而且与 XHTML 1.0 的结合也很紧密，CSS+XHTML 1.0 可以算是最佳实践了。虽然 HTML 4.01 与 XHTML 1.0 没有本质上的区别，但是大部分开发人员接受了这种组合。

XHTML 1.0 之后是 XHTML 1.1，规范本身没有什么新内容，元素、属性也都相同，唯一的变化就是把文档标记为 XML 文档。而在使用 XHTML 1.0 的时候，还可以把文档标记为 HTML 文档。但是，这样做带来了很多问题。首先，把文档标记为 XML 文档后，IE 浏览器不能处理。当然，IE9 及其以上版本是可以处理的。作为全球领先的浏览器，IE 无法处理接收到的 XML 类型的文档，而规范又要求以 XML 类型来发送文档，这对于广大用户来说，是一件很痛苦的事。

接下来，新的版本是 XHTML 2，但是这个版本并没有完成。理论上，XHTML 2 是一个非常好的规范。如果所有人都同意使用的话，也一定是非常好的格式，只不过它还不够实用。首先，XHTML 2 仍然使用 XML 错误处理模型，用户必须保证以 XML 类型发送文档；其次，XHTML 2 中有意不再向后兼容已有的 HTML 版本，甚至曾经讨论废除 img 元素，这对于每天都在做 Web 开发的人员来说确实有点难以接受。

因此，无论 XHTML 2 在理论上是多么完美的一种格式，却从未有机会付诸实践。之所以难以付诸实践，就是因为开发人员永远不会支持它，它向后不兼容。同样，浏览器厂商也不会支持它。

在 2004 年 W3C 成员内部的一次研讨会上，Opera 公司的代表伊恩 • 希克森(Ian Hickson)提出了扩展和改进 HTML 的建议。他建议新的任务组可以跟 XHTML 2 并行，但是在已有 HTML 的基础上开展工作，目标是对 HTML 进行扩展。但是 W3C 投票表示反对，因为他们觉得 XHTML 2 才是未来的方向。然后，Opera、Apple 等浏览器厂商以及其他一些成员脱离了 W3C，成立了 WHATWG(Web Hypertext Applications Technology Working Group，Web 超文本应用技术工作

组)，在 HTML 的基础上开展工作，向其中添加新东西。

WHATWG 的工作不久就初见成效，而 W3C 的 XHTML 2 并没有实质性进展。于是，W3C 于 2007 年组建了 HTML5 工作组，在 WHATWG 工作成果的基础上继续开展工作，由伊恩 • 希克森担任 W3C HTML5 规范的编辑，同时兼任 WHATWG 的编辑，以方便新工作组开展工作。

在 XHTML 2 失败之时，HTML5 已经取得了成功，因为它在开发时考虑到了当前和未来的浏览器开发，以及过去、现在和将来的 Web 开发任务。

Web 发展不断告诉我们的经验之一是，相比理论完美性它更重视实用型开发。HTML5 的发展核心就在于支持所有现有内容的重要性。HTML5 是向后兼容的。它包含 HTML4 规范的全部特性，并包括少量修改和完善。基于这一思想，W3C 指出：应当尽可能将现有 HTML 文档处理成 HTML5，并根据现有浏览器的行为，得到与用户和作者的当前期望相兼容的结果。

2.1.2　XHTML 基础

在 HTML 的历史变迁中，XHTML 是想作为 HTML 的替代者出现的，但是发展到 XHTML 2 的时候失败了，而 HTML5 取得了成功。为了更好地学习和理解 HTML5，有必要对 XHTML 做更深入的学习和理解。

1. XHTML 结构

XHTML 是在 HTML 语言基础上发展而来的，但是为了兼容数以万计的现存网页和不同浏览器，XHTML 文档与 HTML 文档没有太大区别，只是添加了 XML 语言的基本规范和要求。

下面是 Dreamweaver 自动生成的标准 XHTML 页面模板文件，包含以下代码：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
        <meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
        <title>无标题文档</title>
</head>
<body>
</body>
</html>
```

XHTML 代码不排斥 HTML 规则，在结构上也基本相似，但如果仔细比较，它们有两点不同。

(1) 定义文档类型

在 XHTML 文档的第一行新增了<!DOCTYPE>元素，该元素用来定义文档类型。DOCTYPE 是 document type(文档类型)的英文简写，用于设置 XHTML 文档的版本。使用时应注意该元素的名称和属性必须大写。

DTD(如 xhtml1-transitional.dtd)表示文档类型定义，里面包含文档的规则，网页浏览器会根据预定义的 DTD 来解析页面元素，并把这些元素所组织的页面显示出来。要建立符合网页标准的文档，DOCTYPE 声明是必不可少的关键组成部分，除非所编写的 XHTML 确定了正确的

DOCTYPE，否则页面上的元素和 CSS 不能正确生效。

(2) 声明命名空间

在 XHTML 文档的根元素中必须使用 xmlns 属性声明文档的命名空间。xmlns 是 XHTML NameSpace 的英文缩写，中文译为命名空间。命名空间是收集元素类型和属性名称的一个详细 DTD，它允许通过一个 URL 地址指向来识别命名空间。

XHTML 是 HTML 向 XML 过渡的标识语言，它需要符合 XML 规则，因此也需要定义命名空间。又因为 XHTML 1.0 还不允许用户自定义元素，因此它的命名空间都相同，就是 http://www.w3.org/1999/xhtml，这也是每个 XHTML 文档的 xmlns 属性值都相同的原因。

2. XHTML 语法

XHTML 是根据 XML 语法简化而来的，因此它遵循 XML 文档规范。同时 XHTML 又大量继承 HTML 语言的语法规范，因此与 HTML 语言非常相似，不过它对代码的要求更加严谨。遵循以下这些要求，对于培养良好的 XHTML 代码书写习惯是非常重要的：

- 在文档的开头必须定义文档类型。
- 在根元素中声明命名空间，即设置 xmlns 属性。
- 所有标签都必须是闭合的。在 HTML 中，用户可能习惯书写独立的标签，如 、
，而不习惯书写对应的 和
来关闭它们，但在 XHTML 中这是不合法的。XHTML 要求有严谨的结构，所有标签都必须关闭。如果是单独不成对的标签，应在标签的最后加上“/”来关闭，如
。
- 所有元素和属性都必须小写。XHTML 是大小写敏感的，<title>和<TITLE>表示不同的标签，而 HTML 不区分大小写。
- 所有属性必须用引号括起来。在 HTML 中，可以不给属性值加引号，但是在 XHTML 中必须加引号，如<table height="80"></table>。特殊情况下，可以在属性值里使用双引号或单引号。
- 所有标签都必须合理嵌套。这是因为 XHTML 要求有严谨的结构，所有的嵌套都必须按顺序进行。
- 所有属性都必须被赋值，没有值的属性就用自身来赋值。例如，<td nowrap>是错误的，正确的写法是<td nowrap="nowrap">。
- XHTML 规范废除了 name 属性，而使用 id 属性作为统一的名称。

3. XHTML 类型

从上面的介绍可知，XHTML 文档有 3 个主要部分：DOCTYPE、head、body。DOCTYPE 表示文档类型。在 XHTML 文档中，文档类型声明总是位于首行，例如：

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

XHTML 文档类型有 3 种：STRICT(严格类型)、TRANSITIONAL(过渡类型)和 FRAMESET(框架类型)。上面所示的为严格类型，另外两种分别如下：

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

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

这 3 种文档类型的区别如下：

- 严格型文档对文档中的代码要求比较严格，不允许使用任何表现层的标签和属性。在严格型文档中，诸如 center、font、strike、s、u、iframe、isindex、dir、menu、basefont、applet 等元素和 align、language、background、bgcolor、border、height、hspace、name、noshade、nowrap、target、text、link、vlink、alink、vspace、width 等属性将不被支持。
- 过渡型文档对标签和属性的语法要求不是很严格，允许在页面中使用 HTML 4.01 的标签。
- 框架型文档专门针对框架页面所使用的 DTD，当页面中含有框架元素时，就应该采用这种 DTD。

4. DTD 解析

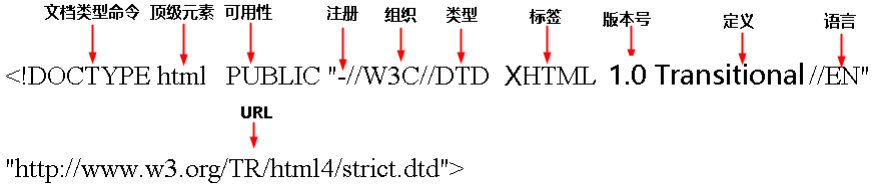
在 XHTML 文档中，只有使用正确的 DOCTYPE(文档类型)，HTML 文档的结构和样式才能被正常解析和呈现。

DTD 是一套关于标签的语法规则。DTD 文件是 ASCII 文本文件，后缀名为.dtd。利用 DOCTYPE 声明中的 URL 可以访问指定类型的 DTD 详细信息。例如，在 XHTML 1.0 中，过渡型 DTD 的 URL 为 http://www.w3.org/TR/XHTML1/DTD/xhtml1-transitional.dtd。

文档类型不同，对应的 DTD 也不同。DTD 文档包含元素的定义规则，元素之间关系的定义规则，元素可使用的属性、实体或符号规则。这些规则用于标识 Web 文档的内容。此外还包括一些其他规则，它们规定哪些标签能出现在其他标签中。

如果页面中没有显式声明 DOCTYPE，那么不同的浏览器就会自动采用各自默认的 DOCTYPE 规则来解析文档中的各种标签和 CSS 样式。

DOCTYPE 声明语句的结构含义如图 2-1 所示。



- 顶级元素：指定 DTD 中声明的顶级元素的类型，这与声明的 SGML 文档类型相对应。HTML 文档默认的顶级元素为 html。
- 可用性：指定是可公开访问的对象 PUBLIC 还是系统资源 SYSTEM。默认为 PUBLIC，SYSTEM 系统资源包括本地文件和 URL。
- 注册：指定组织是否由国际标准化组织 ISO 注册。+表示组织名称已注册，是默认选项。-表示组织名称未注册。W3C 是未注册 ISO 的组织，因此显示为-符号。
- 组织：指定在 DOCTYPE 声明中引用的 DTD 的创建和维护团体或组织的名称。XHTML 语言规范的创建和维护组织为 W3C。

- 类型：指定公开文本的类，即引用的对象类型。
- 标签：指定公开文本的描述，即对引用的公开文本的唯一描述性名称，后面可附带版本号。
- 定义：指定文档类型的定义。
- 语言：指定公开文本的语言。
- URL：指定所引用对象的位置。

由此可见，DOCTYPE 声明语句的写法严格遵循一定的规则，只有这样，浏览器才能够调用对应的文档类型的规则集来解释文档中的标签。

5. 命名空间

在 XHTML 文档中，还有一句常见的代码：

```
<html xmlns="http://www.w3.org/1999/xhtml" >
```

xmlns 属性声明了 html 顶级元素的命名空间，用来定义顶级元素及其包含的各级子元素的唯一性。由于 XML 语言允许用户自定义标签，因此使用命名空间可以避免自己定义的标签和别人定义的标签发生冲突。比如，两个人定义了一模一样的文档，如果文档头部没有用 xmlns 命名空间加以区分，就会发生冲突；如果在文档头部加上不同的命名空间，文档就不会冲突。通俗地讲，命名空间就是给文档做标记，标明文档属于哪个网站。对于 HTML 文档来说，由于元素是固定的，不允许用户进行定义，因此指定的命名空间永远是 http://www.w3.org/1999/xhtml。

2.1.3 Web 开发新时代：HTML5

2014 年 10 月 29 日，万维网联盟宣布，经过几乎 8 年的艰辛努力，HTML5 标准规范终于最终制定完成并公开发布。

1. HTML5 的目标

HTML5 的目标是创建更简单的 Web 程序，书写出更简洁的 HTML 代码。例如，为了使 Web 应用程序的开发变得更容易，提供了很多 API；为了使 HTML 变得更简洁，开发出了新的属性、元素，等等。总体来说，HTML5 为下一代 Web 平台提供了许许多多新的功能。

2. HTML5 新特性

虽然 HTML5 宣称的立场是“非革命性的发展”，但是它所带来的功能是让人渴望的，使用它进行设计也是简单的，因此深受 Web 设计及开发人员的欢迎。

(1) 兼容性

考虑到互联网上 HTML 文档已经存在二十多年了，因此支持所有现存 HTML 文档是非常重要的。HTML5 不是颠覆性的创新，它的核心理念就是要保持与过去技术的兼容和过渡。一旦浏览器不支持 HTML5 的某项功能，针对该功能的备选行为就会悄悄进行。

(2) 合理性

HTML5 新增加的元素都是经过对现有网页和用户习惯进行跟踪、分析和概括而推出的。例如，Google 分析了上百万个页面，从中分析出 标签的通用 ID 名称，并且发现其重复量

很大，如很多开发人员使用 来标记页眉区域。为了解决实际问题，HTML5 直接添加了<header>标签。也就是说，HTML5 新增的很多元素、属性或功能都是根据现实互联网中已经存在的各种应用进行的技术精炼，而不是在实验室中理想化地虚构新功能。

(3) 效率

HTML5 规范是基于用户优先准则编写的，宗旨是“用户即上帝”，这意味着在遇到无法解决的冲突时，HTML5 规范会把用户放在第一位，其次是页面作者，再次是实现者(或浏览器)，接着是规范制定者(W3C/WHATWG)，最后才考虑理论的纯粹性。因此，HTML5 的绝大部分功能是实用的，只是在有些情况下还不够完美。例如，下面的几种代码写法在 HTML5 中都能被识别：

```
id="prohtml5"
id=prohtml5
ID="prohtml5"
```

当然，上面几种写法比较混乱，不够严谨，但是从用户开发角度考虑，用户不在乎代码怎么写，根据个人习惯书写反而能提高代码编写效率。当然，我们并不提倡初学者一开始写代码就这样随意、不严谨。

(4) 安全性

为保证安全，HTML5 规范引入了一种新的基于来源的安全模型，该安全模型不仅易用，而且各种不同 API 都可通用。这个安全模型不需要借助任何所谓聪明、有创意却不安全的 hack 技术就能跨域进行安全对话。

(5) 分离

在清晰分离表现与内容方面，HTML5 迈出了很大一步。HTML5 在所有可能的地方都努力进行了分离，包括 HTML 和 CSS。实际上，HTML5 规范已经不支持旧版 HTML 的大部分表现功能了。

(6) 简单

HTML5 要的就是简单，避免不必要的复杂性。为了尽可能简单，HTML5 做了如下改进：

- 以浏览器原生能力替代复杂的 JavaScript 代码。
- 简化的 DOCTYPE。
- 简化的字符集声明。
- 简单而强大的 HTML5 API。

(7) 通用

通用访问的原则可以分成如下 3 个概念。

- 可访问性：出于对残疾人士的考虑，HTML5 与 WAI(Web Accessibility Initiative，Web 可访问性倡议)和 ARIA(Accessible Rich Internet Application，可访问的富 Internet 应用)做到了紧密结合，WAI-ARIA 中以屏幕阅读器为基础的元素已经被添加到 HTML 中。
- 媒体中立：如果可能的话，HTML5 的功能在所有不同的设备和平台上应该都能正常运行。
- 支持所有语种：例如，新的<body>元素支持在东亚地区页面排版中会用到的 Ruby 注释。

(8) 无插件

在传统 Web 应用中，很多功能只能通过插件或复杂的 hack 技术来实现，但在 HTML5 中提供了对这些功能的原生支持。插件方式存在很多问题：

- 插件安装可能失败。
- 插件可以被禁用或屏蔽(如 Flash 插件)。
- 插件自身会成为被攻击的对象。
- 插件不容易与 HTML 文档的其他部分集成，因为存在插件边界、剪裁和透明度问题。

以 HTML5 的<canvas>元素为例，以前在 HTML4 页面中较难画出对角线，而有了<canvas>元素就可以轻易地实现了。基于 HTML5 的各类 API 的优秀设计，可以轻松地对它们进行组合应用。

3. HTML5 的构成

HTML5 主要包括如下功能：Canvas(2D 和 3D)、Channel 消息传递、Cross-Document 消息传送、Geolocation、MathML、Microdata、Server-Send Events、Scalable Vector Graphics(SVG)、WebSocket API 及协议、Web Origin Concept、Web Storage、Web SQL Database、Web Workers、XMLHttpRequest Level 2。

2.2 HTML 基本语法

HTML 是超文本标记语言，作为一种网页制作语言，它有自己的语法规则，本节就来学习这些基础的语法规则。

2.2.1 标签与元素

在上一章中我们已经制作了一个简单的 HTML 页面，在 HTML 文档中出现了很多用尖括号括起来的字符，这些带尖括号的字符就是 HTML 的“标签”。

通常，一个 HTML 文档中有很多标签，且标签大都成对出现。它们中有“开标签”和“闭标签”。尖括号中没有斜线(/)的标签是开标签，而尖括号中第一个字符为斜线(/)的标签为闭标签，如</html>。

一对标签及二者之间包含的内容称作“元素”(element)。如图 2-2 所示，是例 1-2 所示页面中的<title>元素。

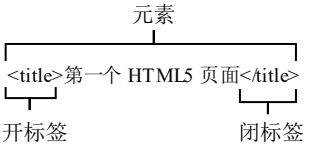


图 2-2　元素与标签示意图

说明：标签通常包含左尖括号、右尖括号以及二者间的字母和数字，如<title>，而元素则是指开标签、闭标签以及二者之间的任何内容。

实际上，整个 HTML 文档都包含在开标签<html>和闭标签</html>之间。大多数 HTML 元素都可以包含其他 HTML 元素，即 HTML 元素可以嵌套。包含另一个元素的元素称作“父元素”，而被包含的那个元素则称为父元素的“子元素”。因此，<title>元素是<head>元素的子元素，而<head>元素是<title>元素的父元素，以此类推。

2.2.2　核心元素

通常我们的 HTML 文档都包含于开标签<html>和闭标签</html>之间(除了第一行的 DOCTYPE)，在<html>元素内部，存在页面的两个主要部分。

- <head>元素：经常被称为页面的头部，包含页面的相关信息(此处不是页面的主体内容)。例如，它可能包含一个<title>元素和一段页面描述或指示信息，告知浏览器从哪里可以找到用于解释文档外观的 CSS 规则。
- <body>元素：经常被称为页面的主体，包含实际希望在浏览器主窗口中显示的信息。

<html>、<head>以及<body>元素构成了一个 HTML 文档的框架——它们是所有网页构建的基础。

1. 关于 DOCTYPE

前面介绍了 XHTML 的 DOCTYPE 是非常复杂的一长串字符，在编写 HTML5 规范时，WHATWG 意识到了这一点，并将它改为构成有效 DOCTYPE 的最短字符序列：

```
<!doctype html>
```

看上去非常简单，而且很好记。这一文档类型声明也是所有 HTML5 页面的第一行代码。

2. <html>元素

<html>元素是整个 HTML 文档的包含元素，出现在 DOCTYPE 声明之后。
<html>元素可以包含以下几个属性：id、dir、lang。

3. <head>元素

<head>元素仅仅是所有其他头部元素的容器。它是开标签<html>后出现的第一个标签。

通常<head>元素内都包含一个<title>元素，用以指定文档的标题。不过，它还可以包含以下元素的任意一种按任意顺序出现的组合。

- <base>元素：用来为页面指定基础 URL 地址。通过这种设置，浏览器会以在相对地址前加上基础地址的方式得到完整的绝对地址。基础 URL 地址的值可以在<base>元素的 href 属性中进行设置。
- <link>元素：用于链接外部文件，例如 CSS 样式表文件等。在学习 CSS 时详细介绍。
- <style>元素：用于在文档内包含 CSS 规则。在学习 CSS 时详细介绍。
- <script>元素：用于在文档内包含脚本。在学习 JavaScript 时详细介绍。
- <meta>元素：包含文档的相关信息，比如一段描述或作者姓名等。

<head>开标签可包含如下几个属性：id、dir、lang。

4. <title>元素

<title>元素用来为网页指定标题，它是<head>元素的一个子元素。它以如下几种方式呈现和使用：

- 在浏览器窗口的标题栏上显示。
- 在 IE、Firefox、Chrome 等浏览器中作为书签的默认名称。
- 搜索引擎使用其内容帮助建立页面索引。

因此，必须使用能够描述网站内容的标题。检验标题好坏的标准是：访问者能否在不必查看网页实际内容的情况下，仅通过读标题就能说出他们将在页面中找到什么；以及是否使用了人们搜索此类信息时常用的字眼。

<title>元素应该只包含标题文本，不可以包含任何其他元素。<title>元素可以包含如下属性：id、dir、lang。

5. 链接与样式表

<link>元素用来添加样式表。该元素可以使用 href 属性指向 Web 上的某个资源。这里的 href 不是为了在单击链接时打开一个新的页面或网站，而是指向为当前页面提供样式信息的文件的位置。使用 rel 属性指明链接的文档是样式表，并且浏览器应据此做相应处理。

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

向页面添加脚本则更加简单。在页面中添加一个<script>元素，然后添加 src 属性，指向需要使用的 JavaScript 文件的位置。

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

6. <body>元素

<body>元素出现在<head>元素之后。它包含实际想在浏览器主窗口中显示的部分，有时也被称为“主体内容”。

7. 常见的内容元素

除了上述主要元素，还有很多用来描述文本结构的不同元素。主要包括如下几类。

- 标题的 6 个级别：<h1>、<h2>、<h3>、<h4>、<h5>及<h6>。
- 段落<p>、预格式化小节<pre>、断行
、块引用<blockquote>以及地址<address>。
- 分组元素： 、<header>、<hgroup>、<nav>、<section>、<article>、<footer>、<aside>以及<hr>。
- 呈现性元素：<b>、<i>、<sup>以及<sub>。
- 短语元素：、、<abbr>、<dfn>、<blockquote>、<q>、<cite>、<code>、kbd>、<var>以及<samp>。
- 列表：如使用<ul>、<li>的无序列表，使用<ol>、<li>的有序列表，以及使用<dl>、<dt>及<dd>的定义列表。
- 编辑元素：<ins>和<del>。

2.2.3　HTML 属性

属性为 HTML 元素提供了更多附加信息。HTML 属性通常以名称/值对的形式出现在开始标签中，例如下面的<style>标签中的 type 属性，值为 text/css：

```
<style type="text/css">
```

有些属性则只含有一个名称，如 required 或 checked 属性。这些属性被称作“布尔属性”。在一个标签中如果只出现一个布尔属性的名称而没有值，则代表该布尔属性的值为 true。因此，以下两行是等价的：

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

1. 核心属性

可以在多数(尽管不是全部)HTML 元素中使用的 4 个核心属性是：id、title、class 和 style。

(1) id 属性

id 属性用来唯一标识页面中的一个元素，或者用来指定一个 CSS 样式或一段 JavaScript 代码应该只被应用于文档中的该元素。

id 属性的使用语法如下(此处的“string”是为该属性选定的值)：

```
id="string"
```

例如，可以使用 id 属性区分两个段落元素，如下所示：

```
<p id="accounts">这个段落是账户信息。</p>
<p id="sales">这个段落是销售信息。</p>
```

id 属性的取值需要遵循下列特殊规则：

- 必须以字母(A~Z 或 a~z)开头。之后可接任意数量的字母、数字(0~9)、横线(-)、下画线(_)、分号(;)以及句号(.)。不能以数字、横线、下画线、分号或句号开头。
- 在文档中必须保持唯一性。同一 HTML 页面中不允许存在两个取值相同的 id 属性。这种情况应该由 class 属性处理。

(2) class 属性

可以使用 class 属性指定某元素属于某一特定“类型”。这种用法在 CSS 中运用非常普遍。后面学习 CSS 时会学习更多有关 class 属性的知识。class 属性的语法如下：

```
class="className"
```

class 属性的取值也可以是一个以空格分隔的 class 名称列表，例如：

```
class="className1 className2 className3"
```

(3) title 属性

title 属性为元素提供标题。title 属性的语法如下：

```
title="string"
```

该属性的行为取决于包含它的元素。不过它经常会作为提示标签或在元素载入时显示。

(4) style 属性

style 属性用来在元素内部指定 CSS 规则。本书第 6 章在介绍 CSS 时会用到该属性，不过，作为总体规则，最好使用一个独立的样式表取而代之。

2. 国际化

Web 是一项全球化技术。因此，很多机制被内置于驱动 Web 的工具中，以允许作者们使用不同语言创建文档。这一过程被称为“国际化”(internationalization)。

有两个常见的国际化属性可以帮助用户使用不同的语言及字符集编写网页：dir 和 lang。

(1) dir 属性

dir 属性用来指定浏览器中文本的显示方向：从左向右或从右向左。当需要为整个文档(或文档的大部分)指定行文方向时，应该在<html>元素而非<body>元素中使用该属性。该属性的两个可能的取值如下。

- ltr：从左向右(默认值)显示
- rtl：从右向左(用于希伯来文或阿拉伯文等从右向左朗读的语言)显示

(2) lang 属性

使用 lang 属性可以指定文档中使用的主要语言。

lang 属性的设计初衷是为用户提供基于语言的显示方式。然而，它在主流浏览器中的效果并不明显。使用 lang 属性的好处主要体现在：搜索引擎(可以告知用户编写文档所用的语言)、屏幕阅读器(可能需要对不同语言使用不同发音)以及应用程序(当不支持该语言或页面语言与默认语言不同时，可以警告用户)。

lang 属性的取值是 ISO-639-1 的标准双字符语言代码。如果想指定该语言的某种方言，可以在语言代码后附上一个横线和次级语言代码。常见的取值如下。

- ar：阿拉伯语
- en：英语
- en-us：美国英语
- zh：中文

3. 全局属性

在 HTML5 中增加了全局属性的概念。全局属性是指可以对任何元素都使用的属性，这些属性如表 2-1 所示。

表 2-1　HTML 全局属性

属　性	描　述
accesskey	规定激活元素的快捷键
contenteditable	规定元素内容是否可编辑
contextmenu	规定元素的上下文菜单。上下文菜单在用户单击元素时显示
data-*	用于存储页面或应用程序的私有定制数据
draggable	规定元素是否可拖动
designMode	指定整个页面是否可编辑，当页面可编辑时，页面中任何支持 contentEditable 属性的元素都变成可编辑状态

(续表)

属　　性	描　　述
dropzone	规定在拖动被拖动数据时是否进行复制、移动或链接
hidden	规定元素仅仅在视觉上看不见，占据的空间位置仍然存在
spellcheck	对用户输入的文本内容进行拼写检查和语法检查
tabindex	规定元素的 Tab 键次序
translate	规定是否应该翻译元素内容

2.2.4　文本格式化

几乎所有的页面都包含某种形式的文本，本节将学习文本格式化相关的 HTML 元素，主要包括如下几个元素：

- <h1>、<h2>、<h3>、<h4>、<h5>以及<h6>元素
- <p>、
以及<pre>元素

在开始学习这些元素之前，先了解一下在没有任何元素时文本的默认显示方式会很有帮助。如果希望浏览器使用不同方式处理文本，这样做有助于展示通过使用标记的重要性。

1. 空格

在开始标记文本之前，先了解一下当 HTML 遇到空格时是如何处理的，以及浏览器是如何处理长句子和文本段落的。

当一段文本中的两个字之间出现多个连续的空格时，默认情况下，屏幕上只有一个空格会被显示。这种处理方式被称为“空格压缩”。类似地，在 HTML 文档中另起一个新行，或者当有多个连续的空行时，这些都会被忽略，并且会作为一个空格处理。

【例 2-1】 HTML 的空格压缩功能。

在 Dreamweaver 中新建一个 HTML 页面，保存为 2-1.html，存放在 Apache 的 htdocs/exam/ch02 目录下，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>HTML 空格压缩</title>
  </head>
  <body>
      <p>这是正文段落，后面有多个空格        和多个空行。


      空格和空行后的文本。
      </p>
  </body>
</html>
```

通过浏览器查看页面，效果如图 2-3 所示，浏览器将多个空格以及若干换行都以单个空格

处理。它还使文本行完全占据浏览器窗口的全部宽度。

现在再来看一遍本例中的代码，并将代码中每个新行的起始处与屏幕中每行的起始处做比较。除非在浏览器显示文本时另行指定，否则将自动占据屏幕的全部宽度，并在空间不足时将文本自动换行以继续显示。为了更好地观察这一效果，可以尝试调整浏览器窗口宽度，并注意文本在屏幕上的新位置。

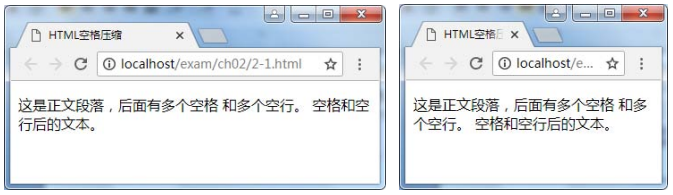


图 2-3　HTML 的空格压缩效果

空格压缩有时非常有用，因为它允许你在 HTML 中加入额外的空格，而在浏览器中查看时又不会显示。可以利用这一特性对代码进行缩进，从而提高代码的可读性。

如果页面中确实需要显示多个连续的空格，该怎么办呢？这时有两种方法：使用<pre>标签(本节后面会介绍)或使用空格符号 ， 代表一个空格，需要多少个空格就添加多少个 。

【例 2-2】 在 HTML 页面中添加多个连续的空格。

新建一个名为 2-2.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>多个连续空格</title>
  </head>
  <body>
      <p>         这是正文段落，后面有多个空格     
   空格后的文本。
      </p>
  </body>
</html>
```

在浏览器中的显示效果如图 2-4 所示。

除了空格以外，在网页制作过程中，还有一些特殊符号也需要使用代码进行代替，一般情况下，这些特殊符号的代码都由前缀“&”、字符名称和后缀“;”组成。常用的特殊符号及对应的代码如表 2-2 所示。

表 2-2　特殊符号及对应的代码

特殊符号	代　　码	特殊符号	代　　码
"	&quot;	&	&amp;
<	&lt;	>	&gt;
×	&times;	§	&sect;
©	&copy;	®	&reg;

2. 使用标题

HTML 提供了 6 种级别的标题，它们对应元素<h1>、<h2>、<h3>、<h4>、<h5>以及<h6>。浏览器以上述 6 个元素中的最大字体显示<h1>，而<h6>对应显示的字体则最小。<h1>通常表示一段文字的标题或主题，所以不宜多用，一个就足够了；<h2>~<h6>使用数目不限，以体现多层次的内容结构。

在较长的文本片段中，标题可以帮助组织文档结构。如果查看一下本书的目录，就能看到不同级别的标题是如何组织的。

【例 2-3】 查看标题的不同级别的显示效果。

新建一个名为 2-3.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>不同级别标题的效果</title>
  </head>
  <body>
    <h1>一级标题 h1</h1>
    <h2>二级标题 h2</h2>
    <h3>三级标题 h3</h3>
    <h4>四级标题 h4</h4>
    <h5>五级标题 h5</h5>
    <h6>六级标题 h6</h6>
  </body>
</html>
```

在浏览器中的显示效果如图 2-5 所示。

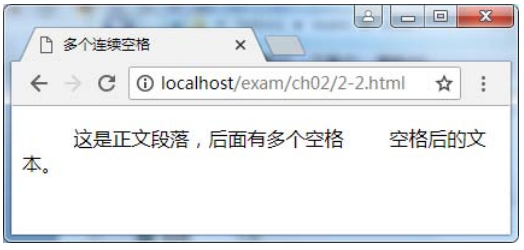


图 2-4　在 HTML 中添加多个连续空格

图 2-5　不同级别标题的显示效果

当然，这 6 个标题元素都可以包含以下通用属性：class、id、style、title、dir 和 lang。

3. 使用<p>元素创建段落

使用段落标签<p>可以分段显示网页中的文本，让文章具有段落之分。合理使用<p>元素，

不仅可以减轻阅读者的视觉疲劳，而且可以让文章更有条理，也利于搜索引擎优化。

开标签 与闭标签 之间的所有文本都在一个段落内，如果要分成多个段落，则需要使用多个 标签。

【例 2-4】 分段显示网页文本。

新建一个名为 2-4.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>分段显示</title>
  </head>
  <body>
      <h2>下面是多个段落文本</h2>
      <p>伤害让一个人成长，时间让一个人坚强</p>
      <p>每个人心中都会有一些记忆，在某个夜深人静的时刻，翻涌而出</p>
      <p>有时候，必须有前面的苦心经营，才有后面的偶然相遇</p>
      <p>我们每个人，至少都有那么一次，付出了很多辛苦，也很真诚，掏心挖肺的，觉得什么都拎出来了，却发现不仅没有回报，还被别人当白痴对待</p>
      <p>倘若你问心有愧呢</p>
      <p>不是所有事情都能如愿以偿，但是任何事情都值得尝试</p>
  </body>
</html>
```

当浏览器显示一个段落时，通常会在下一个段落前插入一个空行，并加入少许额外的纵向空间，如图 2-6 所示。

4. 使用
和<hr/>标签

使用
元素，可以将段落文本换行显示。
元素是一个“空元素”，不需要开闭标签对，因为二者之间不会有任何内容，通常直接在开标签的后面加一条斜线，写作
。一个
代表一次换行，多个
可以实现多次换行。

与
类似的还有一个标签，也不需要开闭标签对，就是水平线标签<hr/>，使用该标签，将在网页中添加一条水平线。

【例 2-5】 使用
和<hr/>标签。

新建一个名为 2-5.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>文本换行与水平线</title>
  </head>
  <body>
      <h2>池上</h2> <hr/>
      小娃撑小艇，<br/>
```

```
        偷采白莲回。<br/>
        不解藏踪迹，<br/>
        浮萍一道开。<br/><hr/>
    </body>
</html>
```

在浏览器中的显示效果如图 2-7 所示。

图 2-6　分段显示文本

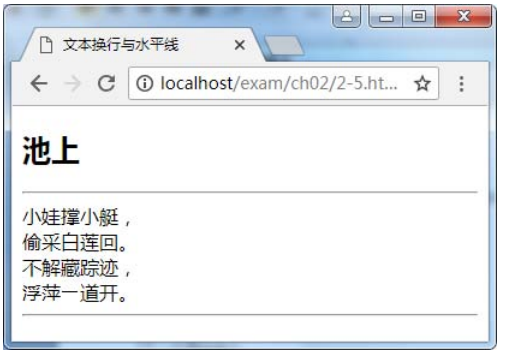


图 2-7　为文本添加换行和水平线

5. 使用<pre>元素预格式化文本

有时候，我们希望文本与它们写在 HTML 文档中的格式完全保持一致，不希望文本在到达浏览器边框时自动换行。也希望浏览器忽略多个空格，并且希望文本能够按照编写时的格式换行。这就需要使用<pre>元素了。任何位于<pre>开标签和</pre>闭标签之间的文本都会保持它们在源文件中的格式。

但是，需要注意的是，大多数浏览器默认会使用等宽字体显示这种文本(Courier 字体就是一种等宽字体，因为每个字母都占用相同的宽度。与之相对的是不等宽字体，这种字体中字母“i”的宽度通常小于字母“m”的宽度)。

<pre>元素经常用来显示源代码，例如下面的例 2-6。

【例 2-6】 使用<pre>元素。

新建一个名为 2-6.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="GB2312" />
        <title>使用 pre 预格式化文本</title>
    </head>
    <body>
        <p>以下内容演示&lt;pre&gt;的用法</p>
        <pre>
    function testFunction( strText ){
          console.log( strText )
```

```
    }
        </pre
    </body>
</html>
```

在浏览器中的显示效果如图 2-8 所示。

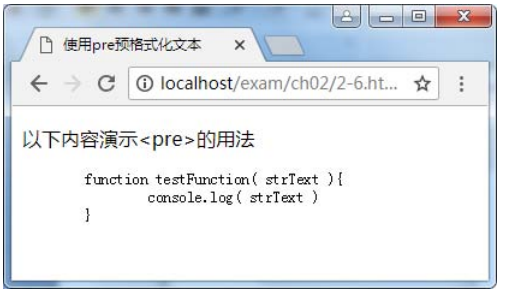


图 2-8　<pre>元素的显示效果

6. 使用字体样式标签

有时候，为了强调某些信息，需要对个别关键词使用不同的字体样式，如粗体或斜体等，这就用到如下几个字体样式标签。

- <b>：该标签将以粗体显示文本。
- ：该标签把文本定义为强调的内容，通常将文本显示为斜体。
- ：该标签把文本定义为语气更强的强调的内容。
- <i>：该标签显示文本斜体效果，和标签类似。它告诉浏览器将包含其中的文本以斜体(italic)或倾斜(oblique)字体显示。
- <sup>：该标签实现文本上标效果。
- <sub>：该标签实现文本下标效果。
- <u>：该标签内的文本将被添加下画线。

【例 2-7】 使用字体样式标签。

新建一个名为 2-7.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="GB2312" />
        <title>不同的字体样式</title>
    </head>
    <body>
        <p>不同样式效果演示</p>
        <b>粗体显示文本</b><br/>
        <em>强调内容，斜体显示文本</em><br/>
        <strong>语气更强的强调</strong><br/>
        <i>斜体显示文本</i><br/>
        <p>上标  a<sup>2</sup></p>
```

```
        <p>下标  H<sub>2</sub>O</p>
        <u>带下画线的文本,很少用,会把它混淆为一个超链接</u>
    </body>
</html>
```

在浏览器中的显示效果如图 2-9 所示。

图 2-9　字体样式显示效果

2.2.5　使用列表

HTML 支持如下 3 种类型的列表。

- 无序列表：无序列表是一列项目，列表项目使用粗体圆点(典型的小黑圆圈)进行标记。
- 有序列表：有序列表也是一列项目，列表项目使用数字进行标记。
- 自定义列表：自定义列表不仅是一列项目，而且是项目及其注释的组合。

1. 无序列表

在HTML中，无序列表的创建标签是 (ul是unordered list无序列表的英文缩写)，在
元素中需要写下的每一项或每一行都应该位于开标签
和闭标签
之间(li是list item的英文缩写)。<ul>和<li>元素可以包含所有通用属性以及UI事件属性。

【例 2-8】 创建无序列表。

新建一个名为 2-8.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="GB2312" />
        <title>无序列表</title>
    </head>
    <body>
        <p>下面是无序列表</p>
        <ul>
```

```
            <li>三国演义</li>
            <li>西游记</li>
            <li>水浒传</li>
            <li>红楼梦</li>
        </ul>
    </body>
</html>
```

在浏览器中的显示效果如图 2-10 所示。

2. 有序列表

有序列表使用的是<ol>标签。在有序列表中，不是在每个项目前放置圆点，而是可以使用数字(1、2、3)、字母(A、B、C)或罗马数字(i、ii、iii)来前置标识它们。

有序列表默认使用从 1 开始的数字标识每个项目，可将前面的无序列表改为有序列表。

【例 2-9】 创建有序列表。

新建一个名为 2-9.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="GB2312" />
        <title>有序列表</title>
    </head>
    <body>
        <p>下面是有序列表</p>
         <ol>
            <li>三国演义</li>
            <li>西游记</li>
            <li>水浒传</li>
            <li>红楼梦</li>
         </ol>
    </body>
</html>
```

在浏览器中的显示效果如图 2-11 所示。

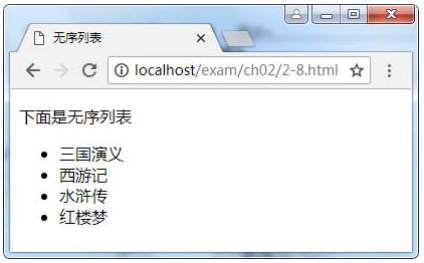


图 2-10　无序列表

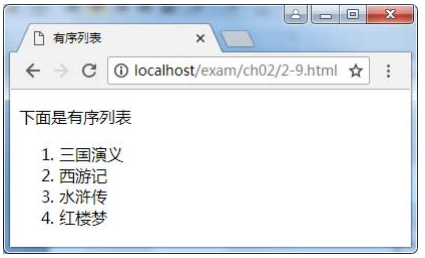


图 2-11　有序列表

如果要使用其他序列标识项目，可以使用 type 属性，该属性的可取值及描述如表 2-3 所示。

表 2-3　type 属性的可取值及描述

属　性　值	描　　述
1	数字，默认选项
a	小写拉丁字母
A	大写拉丁字母
i	小写罗马数字
I	大写罗马数字

修改例 2-9 中的代码，如下所示。在浏览器中查看效果，可以看到列表项目变成了使用大写罗马数字排序标识，如图 2-12 所示。

```
<ol type="I">
        <li>三国演义</li>
        <li>西游记</li>
        <li>水浒传</li>
        <li>红楼梦</li>
</ol>
```

除了指定序列的类型以外，还可以使用 start 属性修改有序列表的起始数字，该属性的值是与起始项对应的序列值，按如下所示修改代码，运行效果如图 2-13 所示。

```
<ol type="a"    start="4">
        <li>三国演义</li>
        <li>西游记</li>
        <li>水浒传</li>
        <li>红楼梦</li>
</ol>
```

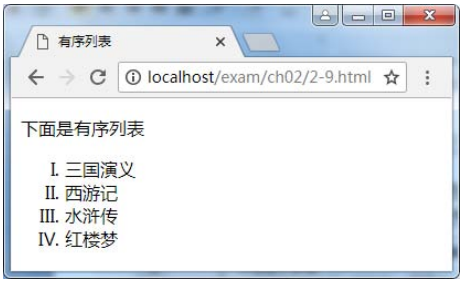


图 2-12　使用 type 属性

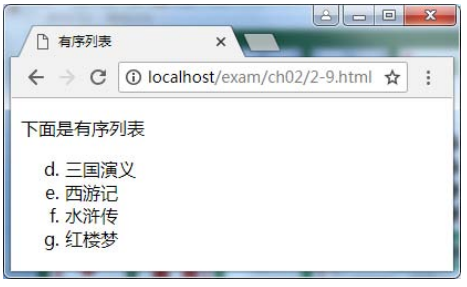


图 2-13　使用 start 属性

除了上面两个属性，HTML5 还新增了布尔属性 reversed，该属性可以使列表的序列反转，也就是从最大值开始向最小值倒数，演示代码如下，效果如图 2-14 所示。

```
<ol type="A"    start="7" reversed>
        <li>三国演义</li>
        <li>西游记</li>
        <li>水浒传</li>
        <li>红楼梦</li>
</ol>
```

3. 自定义列表

使用HTML的<dl>标签可以自定义列表，该标签用于结合<dt>(自定义列表中的项目)和<dd>(描述列表中的项目)。

HTML5 规范定义<dl>的含义是“描述列表”。<dl>元素代表一个描述列表，由 0 个或多个“术语-描述”(名称/值)组构成。每一组都与一个或多个“术语/名称”(<dt>元素的内容)以及一个或多个“描述/值”(<dd>元素的内容)相关联。

自定义列表是一种特殊类型的列表，它的列表项由术语和随后的简短文字定义或描述组成。自定义列表包含在<dl>元素内，之后在<dl>元素内部包含交替出现的<dt>和<dd>元素：<dt>元素的内容是所要定义的术语，<dd>元素中则包含之前<dt>中术语的定义。

【例 2-10】 创建自定义列表。

新建一个名为 2-10.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>自定义列表</title>
  </head>
  <body>
      <p>下面是自定义列表</p>
        <dl>
          <dt>乔峰</dt>
            <dd>小说《天龙八部》中的男主角</dd>
            <dd>生于辽国，长于大宋，实为契丹人</dd>
          <dt>小昭</dt>
            <dd>金庸武侠小说《倚天屠龙记》女主角之一，本名“韩昭”，紫衫龙王黛绮丝和银叶先生韩千叶之女。奉母之命扮作丑陋容貌混入光明顶，盗取乾坤大挪移心法。</dd>
        </dl>
  </body>
</html>
```

在浏览器中的显示效果如图 2-15 所示。

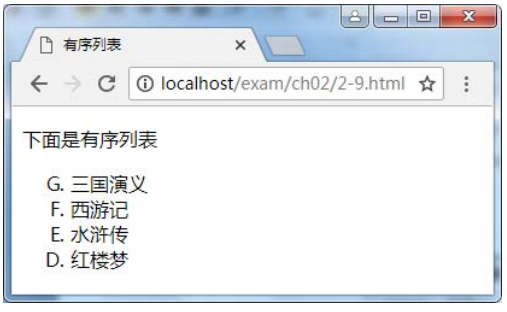


图 2-14　使用 reversed 属性

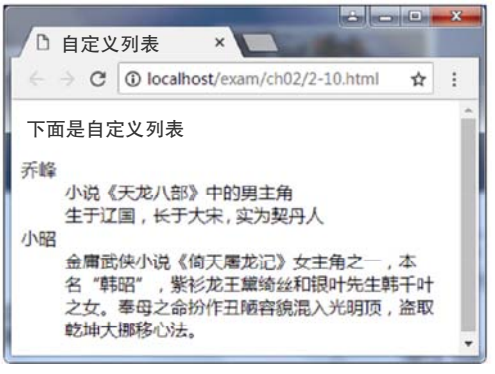


图 2-15　自定义列表

4. 列表的嵌套

可以在一个列表中嵌套另一列表。对于有序列表，除非使用 start 属性另行指定起始序列号，否则，每一个嵌套列表都将独立排序。

【例 2-11】 创建嵌套列表。

新建一个名为 2-11.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>嵌套列表</title>
  </head>
  <body>
  <p>下面是列表嵌套</p>
  <ol type="I">
    <li>我喜欢的水果</li>
    <ol >
      <li>香蕉</li>
      <li>苹果</li>
    </ol>
    <li>我喜欢的人物</li>
    <ol >
      <li>乔峰</li>
      <li>小昭</li>
    </ol>
    <li>我喜欢的蔬菜
    <ul >
      <li>西兰花</li>
      <li>土豆</li>
    </ul>
    </li>
    <li>其他</li>
  </ol>
  </body>
</html>
```

在浏览器中的显示效果如图 2-16 所示。

图 2-16　列表的嵌套

2.2.6 链接与导航

网络之所以有别于其他媒体，就是因为网页中可以包含链接(或者叫超链接)。通过链接，可以从一个页面跳转到另一个页面。链接的形式可以是单词、短语或一幅图片。

链接是网页中极重要的部分，单击文档中的链接，即可跳转至相应的位置。正是因为有了链接，用户才可以在不同的网页中来回跳转，从而方便地查阅各种各样的知识，享受网络带来的无穷乐趣。

1. 基本链接

创建超链接需要使用<a>元素。超链接包含两部分内容：一是链接地址，即链接的目标，可以是某个网址或文件的路径，对应为<a>元素的 href 属性；二是链接文本或图像，单击该文本或图像，将跳转到 href 属性指定的链接地址。超链接的基本语法如下：

```
<a href="链接地址" target="目标窗口位置">链接文本或图像</a>
```

其中，href 属性指定链接地址，target 属性指定链接在哪个窗口中打开，常用的取值是_self(自身窗口)和_blank(新建窗口)。

在<a>和标签对之间，既可以是文本，也可以是图像，例如下面例子中的两个超链接都在新窗口中打开清华大学主页。

【例 2-12】 创建超链接。

新建一个名为 2-12.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="GB2312" />
      <title>超链接</title>
  </head>
  <body>
      <p>下面的超链接都将打开清华大学主页</p>
    <a href="http://www.tsinghua.edu.cn" target="_blank">清华大学</a><br/><hr>
    <a href="http://www.tsinghua.edu.cn"><img src="tsinghua.png"></img></a>
  </body>
</html>
```

在浏览器中的显示效果如图 2-17 所示，单击网页中的文本“清华大学”或者下方的图片都会打开清华大学主页，所不同的是，前者会在新窗口中打开，而后者是在当前窗口中打开。这是因为对于由<a>元素创建的链接，默认情况下，浏览器会在同一窗口内打开目标地址；而使用 target 属性可以改变这一行为，该属性的 4 个可取值及描述如表 2-4 所示。

图 2-17 创建超链接

表 2-4　target 属性的 4 个可取值及描述

属　性　值	描　　述
_blank	浏览器总在一个新打开、未命名的窗口中载入目标地址
_self	这是默认目标，在当前窗口中加载目标地址
_parent	在父框架集中打开被链接文档
_top	在整个窗口中打开被链接文档

target 属性的这 4 个可取值都以下画线开始。任何其他使用一条下画线作为开头的窗口或目标都会被浏览器忽略，因此，不要将下画线作为文档中定义的任何框架名称或 id 的第一个字符。

本例中链接的目标地址是清华大学主页，这里给出的是 URL 地址，URL(Uniform Resource Locator，统一资源定位符) 是对可以从互联网上得到的资源的位置及访问方法的一种简洁表示，是互联网上资源的标准地址。互联网上的每个文件都有唯一的 URL，里面包含的信息指出了文件的位置以及浏览器应该怎么处理它。

URL 包含 3 个关键组成部分：协议(scheme)、主机地址(host address)以及文件路径(file path)。

- 协议：指明文件将以何种方式传输。大多数 Web 页面使用 HTTP(HyperText Transfer Protocol，超文本传输协议)传输信息，因此大多数网页的地址以“http://”开头。不过也有其他前缀，如使用电子银行时会看到“https://”(HTTP 的一种更安全形式)，以及下载大型文件时的“ftp://”。
- 主机地址：通常是网站的域名，如 baidu.com。我们经常会在域名前看到“www”前缀，但它实际上并非域名的一部分。主机地址还可以是数字形式的 IP 地址。
- 文件路径：以一个正斜线(/)开头，并可能包含一个或多个目录名称，然后以一个文件名结束，文件路径通常与网站的目录结构相对应。也可以没有文件名。

如果没有提供文件名，Web 服务器通常会做出以下三种反应之一：

(1) 寻找默认文件并返回。对于以 HTML 编写的网站，默认文件通常是 index.html。如果没有指定文件路径，服务器会在根目录下寻找 index.html 文件；如果指定了目录，服务器则会在该目录中寻找 index.html。

(2) 提供指定目录下的文件列表。

(3) 显示“无法找到页面”或“无法显示文件夹内文件”等提示信息。

2. 绝对地址和相对地址

根据链接地址是指向站外文件还是站内文件，链接地址可分为绝对地址和相对地址。

- 绝对地址：指向目标路径的完整描述，一般指向本站点外的文件或 URL，例如例 2-12 中<a>标签的 href 属性使用的就是绝对地址。
- 相当地址：相对于当前页面的路径，一般指向本站点内的文件，所以一般不需要完整的 URL 地址，例如例 2-12 中加载图片的<img>标签，通过 src 属性指向一个图片文件，用的就是相对地址。

使用相对地址的好处是，可以使代码看起来更简洁，而且当网站的整体域名发生变化时，链接中的相对地址无须修改。

根据目标地址与当前文件所在目录的关系，有如下几种类型的相对地址。

(1) 同一目录

当需要链接到或包含来自同一目录下的某一资源文件时，可以直接使用文件名进行引用，如例 2-12 中的图片文件就和 2-12.html 位于同一目录中。同一站点内多个页面的导航通常属于这一类型，例如，从首页(index.html)链接到登录页面(login.html)，因为这两个文件位于同一目录下，所以直接使用文件名即可。

(2) 子目录

如果网站的文件比较多，为了更好地管理和维护，通常会建立一些子目录，分类存放不同的文件，如图 2-18 所示，TV 和 Music 目录都是 Entertainment 目录的子目录。如果在 Entertainment 目录中建立一个页面，则可以在其中包含如下指向各子目录中 index.html 页面的链接：

```
TV/index.html
Music/index.html
```

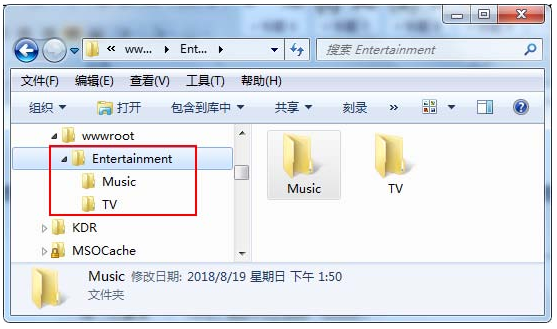


图 2-18　网站内的子目录结构

必须包含子目录名，紧跟一个正斜线(/)，然后是需要链接到的页面的名称。每多一级子目录，就在 URL 中添加子目录名以及正斜线字符。所以，如果 Entertainment 是网站根目录中的一个子目录，则从网站根目录中的页面(如网站首页)创建指向相同页面的链接时，应使用如下相对 URL 地址：

```
Entertainment/TV/index.html
Entertainment/Music/index.html
```

(3) 父目录

如果需要从一个目录中创建指向其父目录(即该目录所在目录)的链接，则可以使用“../”符号(又称“上一级”符号)(两个句点加一个正斜线)。例如，要从 Music 目录中的页面指向 Entertainment 目录中的另一个页面，可以使用如下相对路径：

```
../index.html
```

而如果想从 Music 目录指向根目录，则可以重复该符号，例如：

```
../../index.html
```

每多一次“../”符号，就多向上返回一级目录。

(4) 从根目录出发

如果目录层次较多，为了避免使用过多的“上一级”符号，也可以指定文件相对于网站根

目录的位置。所以，如果想从网站内任意位置链接到 Music 版块的索引页面(index.html)，则可以使用以正斜线开头的路径形式，如下所示：

```
/Entertainment/Music/index.html
```

起始位置的正斜线代表根目录，之后的路径是从根目录开始的相对位置。

3. <base>标签

如前所述，当浏览器遇到相对 URL 地址时，会将相对 URL 地址转换为完整的绝对 URL 地址。而<base>标签可以为页面指定基础地址，该标签没有关闭标签。通过这种设置，浏览器会以在相对地址前加上基础地址的方式得到完整的绝对地址。

基础地址的值在<base>标签的 href 属性中指定。例如，如果需要指定 http://www.example.com/ 作为基础地址，可添加如下<base>元素：

```
<base href="http://www.example.com/" />
```

这时，如果页面中有如下相对 URL 地址：

```
Entertainment/TV/index.html
```

则最终浏览器会请求如下绝对 URL 地址：

```
http://www.example.com/Entertainment/TV/index.html
```

4. 链接到电子邮件地址

链接到电子邮件地址的链接是一种特殊的超链接。单击该链接，会启动电子邮件程序并打开一个新邮件，在“收件人”栏中预先填入该电子邮件地址。

创建电子邮件地址链接的方法是在<a>元素中按如下形式设置 href 属性：

```
<a href="mailto:zhaoyanduo@tsinghua.org.cn">联系我们</a>
```

href 属性的值以关键字 mailto 开始，随后是冒号，然后是希望发送到的电子邮件地址。

除了指定电子邮件地址以外，链接到电子邮件时还可以指定邮件消息的一些其他部分，如主题、邮件主体以及需要抄送及秘密抄送的电子邮件地址。方法是在电子邮件地址后附加一个问号并在之后以“名称/值”对的形式指定相应的特性。名称与值之间使用等号分隔。可添加的电子邮件链接特性如表 2-5 所示。

表 2-5　可用的电子邮件链接特性

特　性	描　述
subject	为电子邮件添加主题行
body	向电子邮件主体中添加默认的消息内容。不过，用户可能会更改该消息内容
cc	向该地址抄送电子邮件。特性的值必须是有效的电子邮件地址。如果需要向多个电子邮件地址抄送，只需要简单重复该特性，特性间使用“&”符号分隔
bcc	指定秘密抄送的电子邮件地址，秘密抄送的收件人之间互不可见。特性的值必须是有效的电子邮件地址。如果需要向多个电子邮件地址秘密抄送，只需要简单重复该属性，特性间使用“&”符号分隔

如果需要指定的特性有多个，则多个特性之间用“与符号”(&)分隔。例如，对于下面的电子邮件链接创建的新邮件，默认主题为“读者来信”，抄送给t_mse@163.com和zhaozx@163.com：

```
<a href="mailto:zhaoyanduo@tsinghua.org.cn?subject=读者来信&cc=t_mse@163.com&cc=
zhaozx@163.com"></a>
```

如果需要在主题中使用空格，则应该使用转义字符%20而不能直接使用空格。同样，如果需要在邮件主体中使用换行，则应该使用%0D%0A(注意此处0是阿拉伯数字，而不是大写字母)。

5. 创建锚点链接

锚点链接(也叫书签链接)常用于那些内容庞大、烦琐的网页，通过单击命名锚点，可以跳转到页面中的特定段落。

网络中比较常见的锚点链接如下：

- 位于长页面底部的返回顶端链接
- 将用户跳转至页面中各相关章节的目录列表
- 行文中的脚注以及定义链接等

锚点链接的创建过程分为两步：创建命名锚点和链接到命名锚点。

(1) 创建命名锚点

锚点是指网页中的某个具有名称的位置。创建锚点同样要使用<a>标签，但需要使用的是id或name属性，而非href属性。

name和id属性是两个通用属性，绝大多数元素都可以包含二者。因为id属性是直到HTML 4.0才被引入的，所以在之前版本中主要是使用name属性。HTML5被推出以后，更建议使用id属性来创建锚点。

```
<a id="top"> </a>
```

上面的代码将创建一个id为top的锚点，前面在介绍核心属性时介绍过id属性。在同一个网页中，id属性的值必须唯一。

(2) 链接到命名锚点

定义了锚点后，就可以链接到命名锚点了。链接到锚点的<a>元素的href属性需要与对应的锚点的id属性值相同，并且在值的前面加上井号(#)。下面的链接将跳转到上面创建的锚点处：

```
<a href="#top">返回顶部</a>
```

除了可以链接到当前网页中的锚点，还可以链接到其他页面中的锚点。此时，href属性值的格式为：文件路径+文件名#锚点名称。锚点名称区分大小写。

【例2-13】 创建锚点链接。

新建一个名为2-13.html的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="GB2312" />
    <title>锚点链接</title>
  </head>
```

```
    <body>
      <p> <a id="top"></a><a href="#C4">查看  第 4 回  </a> </p>
      <h2>第 1 回</h2>
      <p>宴桃园豪杰三结义    斩黄巾英雄首立功</p>
      <h2>第 2 回</h2>
      <p>张益德怒鞭督邮    何国舅谋诛宦竖</p>
      <h2>第 3 回</h2>
      <p>议温明董卓叱丁原    馈金珠李肃说吕布</p>
      <a id="C4"></a><h2 >第 4 回</h2>
      <p >废汉帝陈留践位    谋董贼孟德献刀</p>
      <h2>第 5 回</h2>
      <p>发矫诏诸镇应曹公    破关兵三英战吕布</p>
      <h2>第 6 回</h2>
      <p>焚金阙董卓行凶    匿玉玺孙坚背约</p>
      <h2>第 7 回</h2>
      <p>袁绍磐河战公孙    孙坚跨江击刘表</p>
      <h2>第 8 回</h2>
      <p>王司徒巧使连环计    董太师大闹凤仪亭</p>
      <a href="#top">返回顶部</a>
    </body>
</html>
```

在浏览器中的显示效果如图 2-19 所示。单击顶部的“查看 第 4 回”链接，将跳转到本页面下方“第 4 回”的位置，如图 2-20 所示。在页面底部，还有“返回顶部”锚点链接，单击将返回页面顶部。

图 2-19　锚点链接

图 2-20　页内跳转

其实，像本例中这样有多个小节的长页面，也可以直接为各小节的标题添加 id 属性作为锚点，然后通过锚点链接跳转至相应位置。例如，对于例 2-13 中锚点 C4 的定义，可以修改为如下代码：

```
    <h2 id="C4">第 4 回</h2>
```

2.3　使用表格

表格以行和列的形式显示信息，常用于显示网格结构的数据，如列车时刻表、电视节目表、

财务报告以及体育赛事等。本节将学习在 HTML 中如何创建和使用表格。

2.3.1 创建表格

表格由行、列、单元格 3 部分组成。使用表格可以排列页面中的文本、图像等各类数据。表格的行贯穿表格的左右，从上到下为列，行和列交汇的部分为单元格。

1. 创建基本表格

HTML 的表格通过 4 个标签来创建，分别是表格标签<table>、行标签<tr>、表头标签<th>和表格数据标签<td>。

整个表格是一个<table>元素。在<table>元素内，表格是以行挨着行的形式书写的。每一行包含在一个<tr>元素内，“tr”代表“表格行”(table row)。第一行的单元格是表头，每个单元格使用<th>元素写在行元素内，“th”代表“表格头”(table header)；从第二行开始的每个单元格是表格数据，每个单元格使用<td>元素写在行元素内，“td”代表“表格数据”(table data)。

也可以没有表头，这样就没有<th>元素；也可以将第一列作为表头，这样，每行中的第一个单元格是<th>元素。<th>元素和<td>元素差不多，默认情况下，大多数浏览器会以粗体渲染<th>元素的内容。实际使用中，有的甚至直接使用<td>元素代替<th>元素。

下面来看一个简单的表格，例 2-14 中的两个表格都有表头，一个以第一行作为表头，另一个以第一列作为表头。

【例 2-14】 创建简单的表格。

新建一个名为 2-14.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="GB2312" />
    <title>基本表格</title>
  </head>
  <body>
    <h4>表头</h4>
    <table>
    <tr>
      <th>姓名</th>
      <th>性别</th>
      <th>电话</th>
    </tr>
    <tr>
      <td>赵艳铎</td>
      <td>男</td>
      <td>15910806516</td>
    </tr>
    <tr>
```

```
            <td>赵智暄</td>
            <td>女</td>
            <td>18031760170</td>
        </tr>
        </table>

        <h4>垂直的表头</h4>
        <table >
        <tr>
            <th>姓名</th>
            <td>赵艳铎</td>
            <td>赵智暄</td>
        </tr>
        <tr>
            <th>性别</th>
            <td>男</td>
            <td>女</td>
        </tr>
        <tr>
            <th>电话</th>
            <td>15910806516</td>
            <td>18031760170</td>
        </tr>
        </table>
    </body>
</html>
```

在浏览器中的显示效果如图 2-21 所示。

2. 表格边框

上面创建的表格没有边框，虽然是按行和列排列的，但因为每个单元格内数据的长度不一样，所以有的看起来并不是很清晰。为了让每个单元格中的数据更清晰，可以使用 border 属性，设置表格的边框宽度。默认情况下，不指定该属性，表格边框为 0，浏览器不显示表格边框，如图 2-21 所示。

border 属性的值表示边框的宽度，单位为像素，border 属性设置只影响表格四周的边框宽度，而不影响单元格之间的边框尺寸。

为例 2-14 中的两个表格分别设置边框宽度为 1 和 6，相应的代码如下：

```
        <table border="1">
        ...
        <table border="6">
```

此时页面的效果如图 2-22 所示。

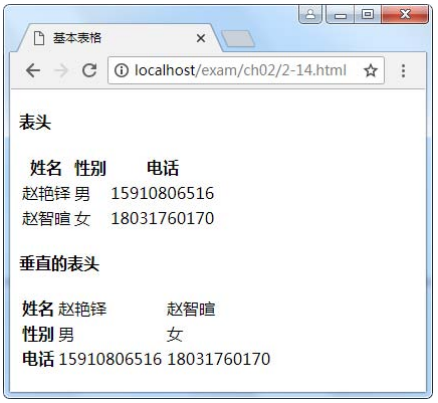


图 2-21　创建简单的表格

图 2-22　为表格添加边框

2.3.2　为表格添加标题

表格常用来显示数据，为了更好地描述表格中数据的来源或用途，通常每个表格都应该拥有标题。这时可以使用<caption>元素，<caption>元素直接出现在开标签<table>之后，并且应该位于第一行之前。默认情况下，多数浏览器会在表格上方的中央位置显示该元素的内容。为例 2-14 中所示表格添加标题的代码如下：

```
<table border="1">
  <caption>通讯录</caption>
  <tr>
```

此时，浏览该页面，效果如图 2-23 所示。

通过使用<caption>元素，相比于仅仅在表格之前或之后的段落中描述其目的，可以将表格内容与该描述相关联，而且这种关联还可以被屏幕阅读器以及其他处理 Web 页面的应用程序(如搜索引擎)所处理。

图 2-23　为表格添加标题

2.3.3　表格的跨行与跨列

上面的表格都比较简单，而现实中往往需要较复杂的表格。比如有时需要把多个单元格合并为一个，这就用到本节要学习的两个属性：colspan 和 rowspan。

1. 使用 colspan 属性实现跨列

跨列是指将同一行中的多个单元格合并为一个，这就用到<td>或<th>标签的 colspan 属性。col 是 column(列)的英文缩写，span 表示宽度，colspan 属性的值为当前单元格跨越的列数。

【例 2-15】 现在很多人都有不止一个手机号码，所以我们修改上面的通讯录表格，增加一列以存放备用电话，在表头中“电话”占两列。

新建一个名为 2-15.html 的页面，输入如下代码：

```
<!DOCTYPE html>
```

```
<html>
  <head>
    <meta charset="GB2312" />
    <title>跨列表格</title>
  </head>
  <body>
    <h4>电话  占两列</h4>
    <table border="1" width="500">
    <caption>  通讯录</caption>
    <tr>
      <th>姓名</th>
      <th>性别</th>
      <th colspan="2">电话</th>
    </tr>
    <tr>
      <td>赵艳铎</td>
      <td>男</td>
      <td>15910806516</td>
      <td>13910002312</td>
    </tr>
    <tr>
      <td>赵智暄</td>
      <td>女</td>
      <td>18031760170</td>
      <td></td>
    </tr>
    </table>
  </body>
</html>
```

在浏览器中的显示效果如图 2-24 所示，表格中的第一行有 3 个单元格，并且第 3 个单元格跨越两列的宽度。

2. 使用 rowspan 属性实现跨行

rowspan 属性的作用与 colspan 类似，只是在相反的方向上工作：rowspan 使单元格可以纵向跨越行。

【例 2-16】在成绩表中，对于相同的成绩，名次是并列的。这时，可以使用跨行表格让并列名次跨多行。

新建一个名为 2-16.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="GB2312" />
    <title>跨行表格</title>
  </head>
```

```
    <body>
      <h4>并列的名次  占多行</h4>
      <table border="1" width="300">
      <caption>  成绩表</caption>
      <tr>
        <th>名次</th>
        <th>姓名</th>
        <th>分数</th>
      </tr>
      <tr>
        <td>1</td>
        <td>赵艳铎</td>
        <td>100</td>
      </tr>
      <tr>
        <td rowspan="2">2</td>
        <td>赵智暄</td>
        <td>99</td>
      </tr>
      <tr>
        <td>贾梓怡</td>
        <td>99</td>
      </tr>
      <tr>
        <td >4</td>
        <td>邢欣蕊</td>
        <td>96</td>
      </tr>
      </table>
    </body>
</html>
```

在浏览器中的显示效果如图 2-25 所示，表格中的第 2 名有两位同学，所以相应的单元格跨越两行。

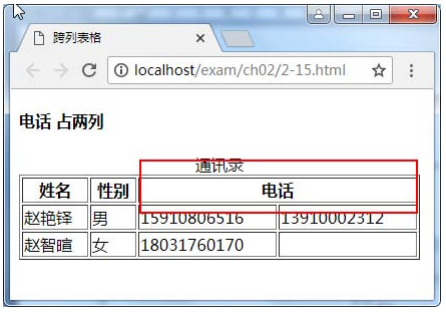


图 2-24　创建跨列表格

图 2-25　创建跨行表格

2.3.4　表格的结构标签

为在源代码中清楚地区分表格结构，HTML 提供了<thead>、<tbody>和<tfoot>三个标签，分别对应表格的表头、表体和表尾。

例如，对于银行的结算系统：可能有这样一个表格，其中表头包含每列的头部信息，表体包含存取交易事务的列表，而表尾则包含账户的结算余额。

如果表格太长以致在屏幕中显示不全，那么表头与表尾可以始终保持可见，而为表体设置滚动条。另外，还可使用 CSS 为<thead>、<tbody>以及<tfoot>中的内容添加不同的样式风格。

这三个标签的用法都比较简单，只需要将相应的<tr>元素放在相应的开闭标签对之间即可。例如，为前面的表格增加表头标签<thead>的代码如下：

```
<thead>
<tr>
  <th>名次</th>
  <th>姓名</th>
  <th>分数</th>
</tr>
</thead>
```

如果表体中的数据还有不同的分组，则可以使用多个<tbody>标签，以指明不同的页面或数据分组。

2.3.5　对表格的列进行分组

如果表格中的两列或更多列是相互关联的，则可以使用<colgroup>元素解释这些列应该被归到同一组中，以便通过 CSS 为不同组中的列应用不同的样式。

<colgroup>元素在使用时，应该直接出现在<table>开标签之后，并包含 span 属性，用来指定该组中包含多少列。例如，下面的表格一共有 6 列。前面的 4 列属于第 1 个列组，而后面的两列则属于第 2 个列组。

```
<table>
  <colgroup span="4" class="mainColumns" />
  <colgroup span="2" class="subColumns" />
  <tr>
    <td>1</td>
    <td>2</td>
    <td>3</td>
    <td>4</td>
    <td>5</td>
    <td>6</td>
  </tr>
</table>
```

在上面的代码中，使用<colgroup>的 class 属性为同一组中的列应用同一 CSS 规则(class 属性的值为 CSS 中定义的规则)，从而告知浏览器该分组中列的宽度以及每个单元格的背景颜色等。有关 CSS 的更多内容，将在后面的第 6 章中介绍。

如果同一分组中的列还需要进一步细分，以应用不同的样式，则可以在<colgroup>元素中使用<col>元素。

<col>元素用来为<colgroup>中的列指定样式规则(如列内单元格的宽度与对齐方式)。<col>元素永远是空元素，它没有任何内容，必须采用单标签的形式<col />，但可以包含属性。

例如，下面的表格有 6 列，前 3 列为一组，后 3 列为一组，在每一组中又使用<col>元素为该组中的列应用不同的样式，从而可以更灵活地设置任意列的样式。

```
<table>
   <colgroup span="3">
      <col span="1" class="mainColumns" />
      <col span="2" class="totalColumn" />
   </colgroup>
   <colgroup span="3">
      <col span="2" class="mainColumns" />
      <col span="1" class="totalColumn" />
   </colgroup>
   <tr>
      <td></td>
      ...
      <td></td>
   </tr>
</table>
```

2.3.6 嵌套表格

在表格的单元格中也可以包含其他 HTML 元素，只要它们全部包含于单元格内即可。当一个表格的单元格内是另一个表格时，就创建了“嵌套表格”。

【例 2-17】 创建一个表格，用前面学过的一些标签填充单元格的内容。

新建一个名为 2-17.html 的页面，输入如下代码：

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="GB2312" />
      <title>嵌套表格</title>
   </head>
   <body>
      <h4>表格内的标签</h4>
      <table border="1">
         <tr>
            <td>   <p>这里包含一个段落</p> </td>
            <td>这里是一个嵌套表格
               <table border="4" >
                  <tr>
                     <td>东邪</td>
                     <td>西毒</td>
```

```
                </tr>
                <tr>
                  <td>南帝</td>
                  <td>北丐</td>
                </tr>
              </table>
            </td>
          </tr>
          <tr>
            <td>这个单元包含一个列表
              <ol>
              <li>苹果</li>
              <li>香蕉</li>
              <li>菠萝</li>
              </ol>
            </td>
            <td>这里包含超链接<br>
              <a href="2-1.html">例 2-1</a><br>
              <a href="2-2.html">例 2-2</a>
            </td>
          </tr>
        </table>
    </body>
</html>
```

在浏览器中的显示效果如图 2-26 所示。

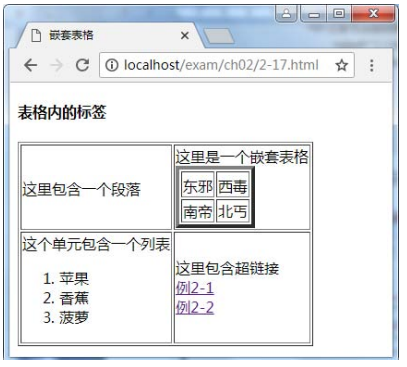


图 2-26　在表格内包含其他 HTML 元素

HTML 的表格功能非常强大，表格本身可以包含大量数据，并能对这些信息提供一种友好的视觉呈现形式。然而，为了使表格对所有人都容易理解，在制作表格时有如下几点建议：

- 使用<caption>元素为表格添加标题。标题能概括表格所描绘的内容，有了醒目的标题，理解表格信息就会容易得多。
- 尽量使用<th>元素指明表头，多数浏览器会默认以粗体渲染<th>元素，让读者一眼看清各列数据的含义。
- 始终将表头放在第一行与第一列。

- 避免使用嵌套表格。
- 少用 rowspan 与 colspan 属性。

2.4 本章小结

本章讲述了 HTML 的基本语法。首先，介绍了 HTML 的历史变迁、XHTML 与 HTML 的关系以及 HTML5 时代的到来。接下来，对 HTML 中的常用标签进行了详细介绍，包括标签和元素的基本概念、HTML 的属性、文本格式化标签、HTML 列表、链接的创建与使用等。最后，学习了 HTML 的表格，HTML 提供了强大的表格功能，合理地利用表格，对于展示页面内容有很大帮助。本章介绍的都是 HTML 中最基础也是最简单的内容，是网页设计的开始，后面会有更多高级技巧等着我们去尝试和挑战。

2.5 思考和练习

1. 在 HTML 4.01 之后，W3C 提出了_____的概念。
2. 在 HTML 的历史变迁中，_____是想作为 HTML 的替代者出现的，但在发展过程中失败了，而 HTML5 取得了成功。
3. XHTML 文档的第一行新增了_____元素，该元素用来定义文档类型。
4. DTD 是一套关于标签的_____。DTD 文件是_____文件，后缀名为_____。
5. 简述 HTML5 的构成。
6. 什么是 HTML 的标签，什么是 HTML 的元素？
7. 在<html>元素内部，存在页面的两个主要部分：_____和_____。
8. 所有 HTML5 页面的第一行代码都是一样的，这行代码是什么？
9. 如何在 HTML 页面中输入多个连续的空格？
10. <pre>标签是干什么用的？
11. 在 HTML 中，如何实现文本上标效果？
12. HTML 支持几种类型的列表？
13. 标签的 type 属性值中的 i 表示什么含义？
14. 什么是绝对地址，什么是相对地址？
15. 如何创建链接到电子邮件的链接？
16. 创建表格要用到哪些标签？
17. 表格的<caption>元素应该被放置于文档内什么位置？默认情况下，它在哪里显示？