

Web前端工程师 修炼之道

Learning Web Design: A Beginner's Guide to HTML, CSS,
JavaScript, and Web Graphics, Fourth Edition

Jennifer Niederst Robbins 著

谢合亮 张晶 靳志伟 译

无论你还是一名初学者还是想进一步提高自己的网页设计技能，本书都能为你提供极具参考性的信息。我力求生动地展现每一个设计细节，配以大量的练习和实例帮助你运用并掌握最新的技术。阅读本书就好像跟随我在课堂学习一样，轻松且愉悦！——Jennifer Niederst Robbins

Web前端工程师修炼之道 原书第4版

你是否也曾想过自己创建网页，但却苦于没有经验？那么从现在开始学习吧！本书由浅入深地讲解了Web设计的一些重要概念、基本原理，以及HTML、CSS和JavaScript的具体使用方法与技巧。当你读完本书后，将会掌握创建适用于移动设备的多列页面的技巧。

本书将帮助你学习如何使用最新技术、最佳实践经验，以及当今的Web标准，其中包括HTML5和CSS3。每章都提供一些练习和小测验，帮助你了解不同的技术，巩固你对重要概念的理解。

作者对本版内容做了全面更新和修订，涵盖Web前端开发所需的一切，无论是初学者，还是想提高已有技能的专业开发人员，本书都是理想的选择。

本书主要内容：

- 使用文本、链接、图像、表格和表单来创建HTML页面
- 使用CSS调整颜色、背景、格式化文本、页面布局，甚至是实现简单的动画效果
- 学习新的HTML5元素、API和CSS3属性——它们改变了Web页面处理方式
- 创建响应式Web设计，使得页面可以在移动设备上得到良好显示
- 学习JavaScript的工作原理及其在Web设计中的重要性
- 创建并优化Web图像，加快图像下载速度

作者简介

Jennifer Niederst Robbins 资深Web设计师、信息架构师，从事Web设计20余年，在Web设计方面有独到的见解与认识。她于1993年设计了全球第一个商业网站——O'Reilly全球网络导航器（O'Reilly Global Network Navigator, GNN）。她曾在马萨诸塞艺术学院和约翰森威尔士大学任教，并且经常在全美许多重要会议上发表演讲，广受欢迎。她现在是O'Reilly Media的数码产品设计师，对信息架构、交互设计、网站制作、应用程序开发和电子书拥有浓厚的兴趣。除了本书，还著有《Web Design in a Nutshell》和《HTML5 Pocket Reference》。



O'Reilly Media, Inc. 授权机械工业出版社出版

此简体中文版仅限于在中华人民共和国境内（但不允许在中国香港、澳门特别行政区和中国台湾地区）销售发行
This Authorized Edition for sale only in the territory of People's Republic of China
(excluding Hong Kong, Macao and Taiwan)

投稿热线：(010) 88379604
客服热线：(010) 88378991 88361066
购书热线：(010) 68326294 88379649 68995259

华章网站：www.hzbook.com
网上购书：www.china-pub.com
数字阅读：www.hzmedia.com.cn

上架指导：计算机/Web设计

ISBN 978-7-111-47168-4



9 787111 471684 >

定价：129.00元

O'REILLY

原书第4版

Web前端工程师修炼之道

Jennifer Niederst Robbins 著

谢合亮 张晶 靳志伟 译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'Reilly Media, Inc. 授权机械工业出版社出版

机械工业出版社

图书在版编目 (CIP) 数据

Web前端工程师修炼之道 (原书第4版) / (美) 罗宾斯 (Robbins, J. N.) 著; 谢合亮, 张晶, 靳志伟译.
—北京: 机械工业出版社, 2014.7
(O'Reilly精品图书系列)

书名原文: Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics, Fourth Edition

ISBN 978-7-111-47168-4

I. W… II. ①罗… ②谢… ③张… ④靳… III. ①超文本标记语言—程序设计 ②网页制作工具 ③JAVA语言—程序设计 IV. ①TP312 ②TP393.092

中国版本图书馆CIP数据核字 (2014) 第139387号

北京市版权局著作权合同登记

图字: 01-2012-8343号

©2012 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and China Machine Press, 2014. Authorized translation of the English edition, 2012 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由O'Reilly Media, Inc. 出版2012。

简体中文版由机械工业出版社出版 2014。英文原版的翻译得到O'Reilly Media, Inc.的授权。此简体中文版的出版和销售得到出版权和销售权的所有者——O'Reilly Media, Inc.的许可。

版权所有, 未得书面许可, 本书的任何部分和全部不得以任何形式重制。

封底无防伪标均为盗版

本书法律顾问

北京大成律师事务所 韩光/邹晓东

书 名/ Web前端工程师修炼之道 (原书第4版)

书 号/ ISBN 978-7-111-47168-4

责任编辑/ 秦健

封面设计/ Karen Montgomery, 张健

出版发行/ 机械工业出版社

地 址/ 北京市西城区百万庄大街22号 (邮政编码 100037)

印 刷/ 藁城市京瑞印刷有限公司

开 本/ 204毫米×255毫米 16开本 38.75印张 (含1印张彩插)

版 次/ 2014年9月第1版 2014年9月第1次印刷

定 价/ 129.00元 (册)

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010)88378991; 88361066

购书热线: (010)68326294; 88379649; 68995259

投稿热线: (010)88379604

读者信箱: hzjsj@hzbook.com



图1-4：一个简单网站的外观和视觉的速写

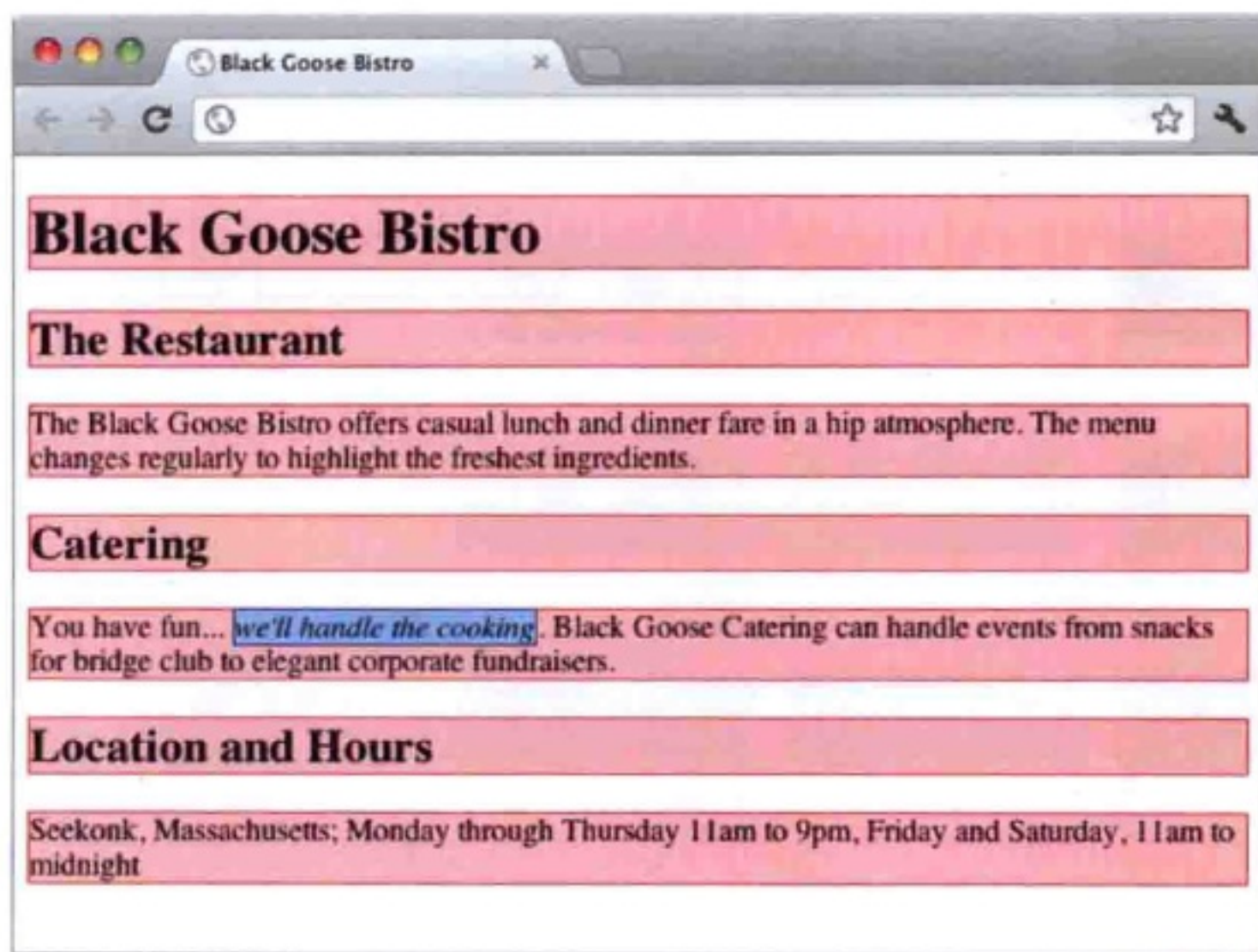


图4-10：突显之处展示了主页中元素的结构



CSS Zen Dragen
by Matthew Buchanan



Shaolin Yokobue
by Javier Cabrera



By the Pier
by Peter OngKelmscott



Organica Creativa
by Eduardo Cesario

图11-1: CSS Zen Garden中的这些网页使用相同的XHTML源码文档, 但使用专门的CSS来改变设计 (CSS Zen Garden和设计师许可使用本图)

			
Black #000000	Gray #808080	Silver #C0C0C0	White #FFFFFF
			
Maroon #800000	Red #FF0000	Purple #800080	Fuchsia #FF00FF
			
Green #008000	Lime #00FF00	Olive #808000	Yellow #FFFF00
			
Navy #000080	Blue #0000FF	Teal #008080	Aqua #00FFFF
			
Orange (CSS 2.1) #FFA500			

图13-1: CSS2.1中的17个标准颜色名

aliceblue 240,248,255 F0F8FF	cornsilk 255,248,220 FFF8DC	darkturquoise 0,206,209 00CED1	hotpink 255,105,180 FF69B4	lightskyblue 135,206,250 87CEFA	midnightblue 25,25,112 191970	peru 205,133,63 CD853F	snow 255,250,250 FFFAFA
antiquewhite 250,235,215 FAEBD7	crimson 220,20,60 DC143C	darkviolet 148,0,211 9400D3	indianred 205,92,92 CD5C5C	lightslategray 119,136,153 778899	mintcream 245,255,250 F5FFFA	pink 255,192,203 FFC0CB	springgreen 0,255,127 00FF7F
aqua 0,255,255 00FFFF	cyan 0,255,255 00FFFF	deeppink 255,20,147 FF1493	indigo 75,0,130 4B0082	lightsteelblue 176,196,222 B0C4DE	mistyrose 255,228,225 FFE4E1	plum 221,160,221 DDA0DD	steelblue 70,130,180 4682B4
aquamarine 127,255,212 7FFFD4	darkblue 0,0,139 00008B	deepskyblue 0,191,255 00BFFF	ivory 255,240,240 FFFFF0	lightyellow 255,255,224 FFFFE0	moccasin 255,228,181 FFE4B5	powderblue 176,224,230 B0E0E6	tan 210,180,140 D2B48C
azure 240,255,255 F0FFFF	darkcyan 0,139,139 008B8B	dimgray 105,105,105 696969	khaki 240,230,140 F0E68C	lime 0,255,0 00FF00	navajowhite 255,222,173 FFDEAD	purple 128,0,128 800080	teal 0,128,128 008080
beige 245,245,220 F5F5DC	darkgoldenrod 184,134,11 8B6914	slateblue 96,144,255 6A5ACD	lavender 230,230,250 E6E6FA	lavender 50,205,50 32CD32	navy 0,0,128 000080	red 255,0,0 FF0000	thistle 216,191,216 D8BFD8
bisque 255,228,196 FFE4C4	darkgray 169,169,169 A9A9A9	firebrick 178,34,34 B22222	lavenderblush 255,240,245 FFF0F5	linen 250,240,230 FAFAD2	oldlace 253,245,230 FDF5E6	rosybrown 188,143,143 DCB78F	tomato 255,99,71 FF6347
black 0,0,0 000000	darkgreen 0,100,0 006400	floralwhite 255,250,240 FFFAF0	lawngreen 124,252,0 7CFC00	magenta 255,0,255 FF00FF	olive 128,128,0 808000	royalblue 65,105,225 4169E1	turquoise 64,224,208 40E0D0
blanchedalmond 255,255,205 FFFACD	darkkhaki 189,183,107 8B762D	forestgreen 34,139,34 228B22	lemonchiffon 255,250,205 FFFACD	maroon 128,0,0 800000	olivedrab 107,142,35 6B8E23	saddlebrown 139,69,19 8B4513	violet 238,133,238 EE82EE
blue 0,0,255 0000FF	darkmagenta 139,0,139 8B008B	fuchsia 255,0,255 FF00FF	lightblue 173,216,230 ADD8E6	mediumaquamarine 102,205,170 66CDAA	orange 255,165,0 FFA500	salmon 250,128,114 FA8072	white 255,255,255 FFFFFF
blueviolet 138,43,226 8A2BE2	darkolivegreen 85,107,47 556B2F	gainsboro 220,220,220 DCDCDC	lightcoral 240,128,128 F08080	mediumslateblue 0,0,205 0000CD	orchid 216,112,214 DA70D6	sandybrown 244,164,96 F4A460	wheat 245,222,179 F5DEB3
brown 165,42,42 A52A2A	darkorange 255,140,0 FF8C00	ghostwhite 248,248,255 F8F8FF	lightgoldenrodyellow 250,250,210 FAFAD2	mediumorchid 186,85,211 BA55D3	orangered 255,68,0 FF4500	seagreen 46,139,87 2E8B57	whitesmoke 245,245,245 F5F5F5
burlywood 222,194,135 DEB887	darkred 139,0,0 8B0000	gold 255,215,0 FFD700	lightcyan 224,255,255 E0FFFF	mediumpurple 147,112,210 9370DB	palegoldenrod 238,232,170 EEE8AA	seashell 255,245,238 FFF5EE	yellow 255,255,0 FFFF00
cadetblue 95,135,160 5F9EA0	darkorchid 153,50,204 9932CC	goldenrod 218,165,32 DAA520	lightgreen 144,238,144 90EE90	mediumseagreen 60,179,113 3CB371	palegreen 152,251,152 98FB98	sienna 160,82,45 A0522D	yellowgreen 154,205,50 9ACD32
chartreuse 127,255,0 7FFF00	darksalmon 233,130,122 E9967A	gray 128,128,128 808080	lightgray 211,211,211 D3D3D3	mediumslateblue 123,104,210 7968D8	paleturquoise 175,238,238 AFEEEE	silver 192,192,192 C0C0C0	
chocolate 210,105,30 CD853F	darkseagreen 143,188,143 8FBC8F	green 0,128,0 008000	lightpink 255,182,193 FFB6C1	mediumspringgreen 0,250,154 90FA8A	palevioletred 219,112,147 DB7093	skyblue 135,206,235 87CEEB	
coral 253,127,80 FF7F50	darkslateblue 72,61,139 483D8B	greenyellow 173,255,47 ADFF2F	lightsalmon 255,160,122 FFA07A	mediumturquoise 72,209,204 48D1CC	papayawhip 255,239,213 FFEFD5	slateblue 106,90,205 6A5ACD	
cornflowerblue 100,149,237 6495ED	darkslategray 47,79,79 2F4F4F	honeydew 240,255,240 F0FFD0	lightseagreen 32,178,170 20B2AA	mediumvioletred 199,21,133 C71585	peachpuff 255,239,213 FFEFD5	slategray 112,128,144 708090	

图13-2: CSS3中的140个扩展颜色名。你需要明白在屏幕上这些色彩看起来可能不太一样

RGB颜色系统

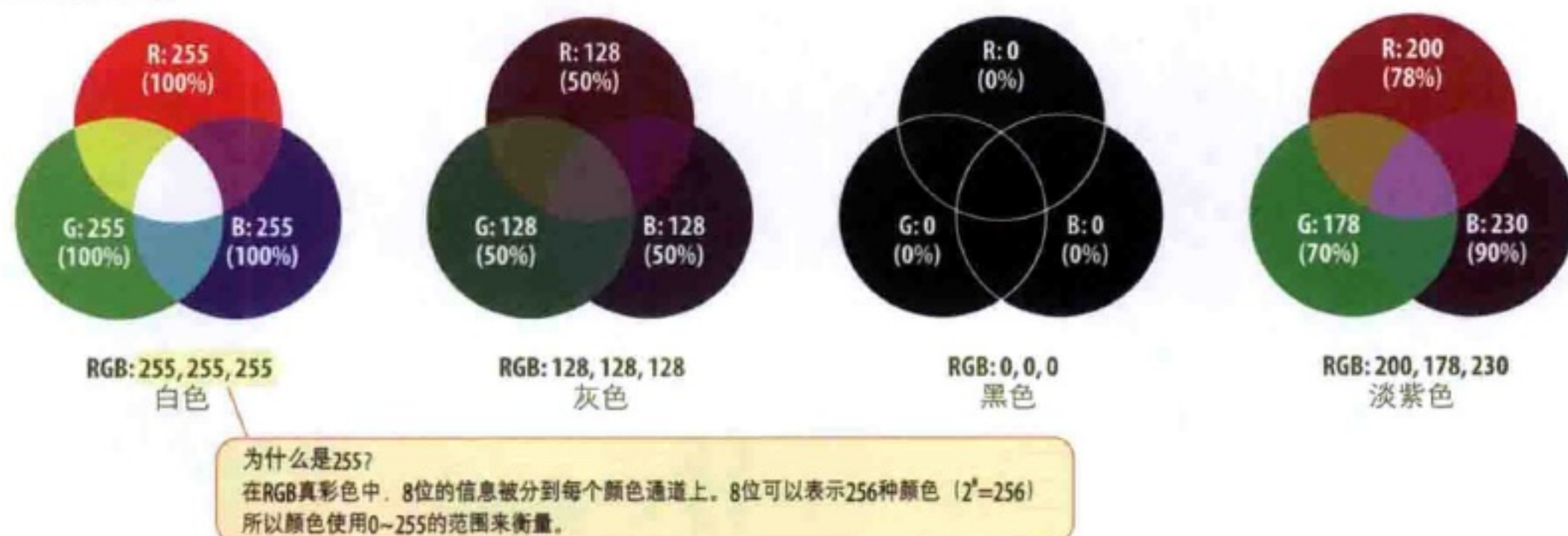


图13-3: 计算机显示的颜色是由不同数量的红、绿、蓝光(即RGB)混合而成的。每个图中间的颜色就是三个颜色通道混合出来的。每个通道里的光越多, 混合色就越接近白色

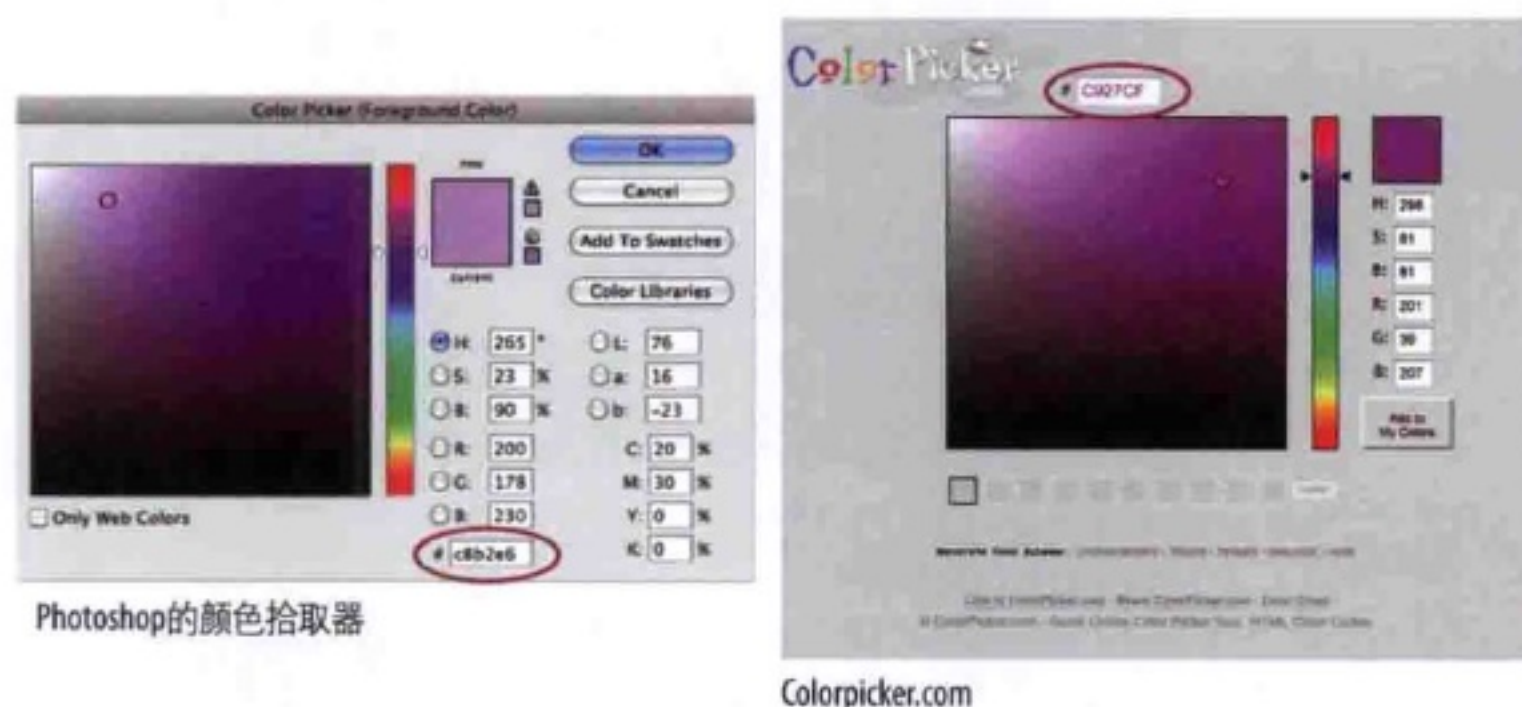


图13-4: Photoshop(左边)和Colorpicker.com(右边)的颜色拾取器, 可以为选中的颜色像素提供RGB值

十六进制的RGB值必须以#符号开始

#RRGGBB

hex RED value hex GREEN value hex BLUE value

图13-5: 十六进制数由三个两位数组成, 分别表示红、绿和蓝

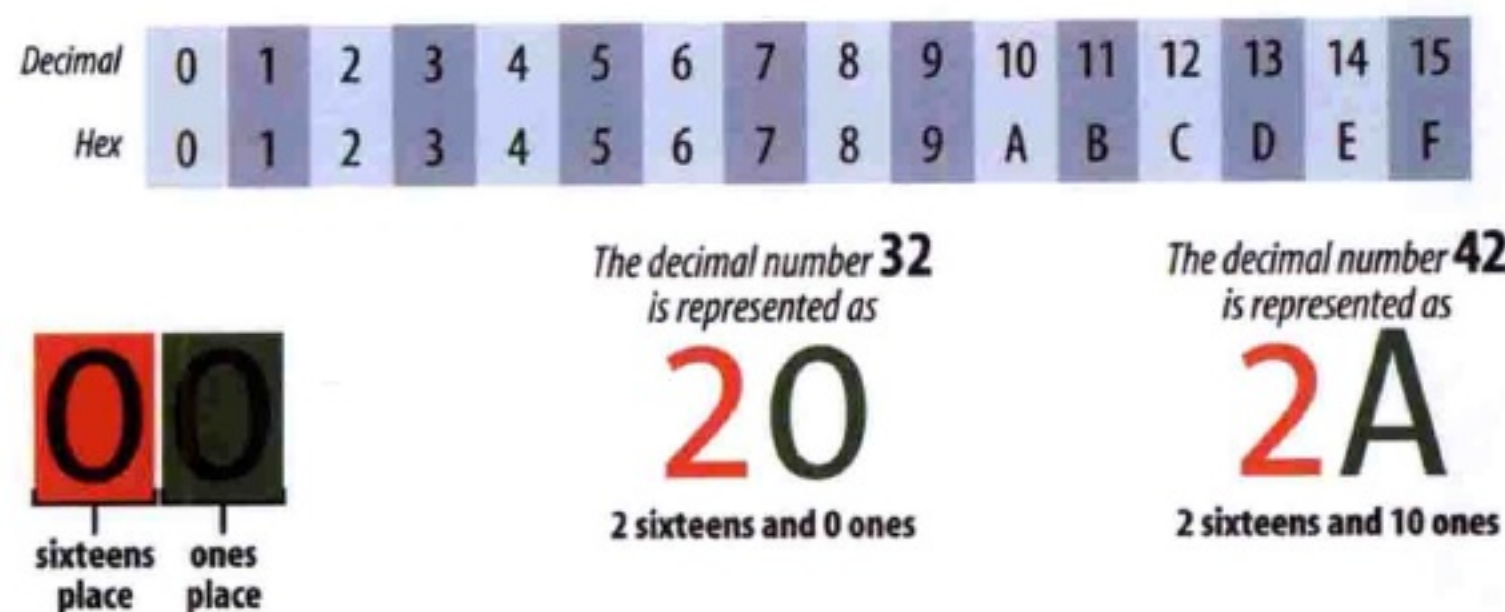


图13-6：十六进制数字系统以16为基数

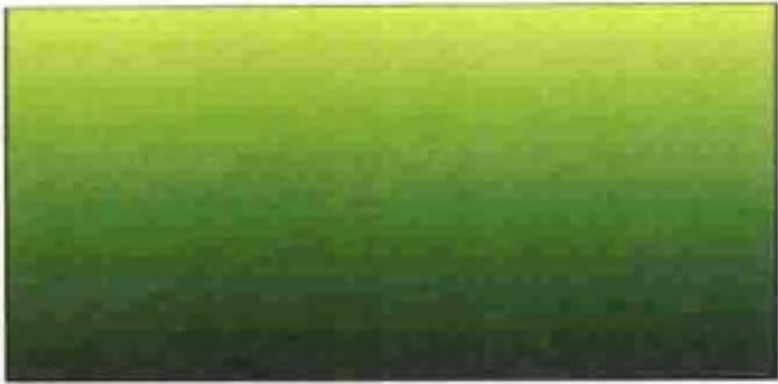


应用之前

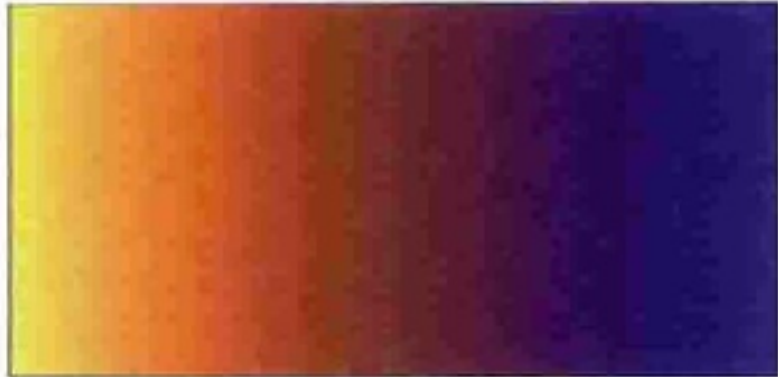


应用之后

图13-15：应用色彩之后的Black Goose Bistro菜单页面



```
linear-gradient(to bottom, yellow, green);
```



```
linear-gradient(90deg, yellow, orange 25%, blue);
```



```
linear-gradient(to bottom, #e2e2e2 0%, #dbdbdb 50%, #d1d1d1 51%, #fefefe 100%);
```

图13-25: 线性渐变的例子



```
radial-gradient(center, yellow, green);
```

图13-26: 径向渐变的例子

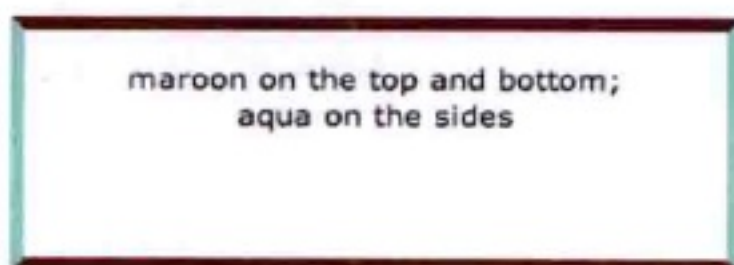


图14-11: 指定边框颜色

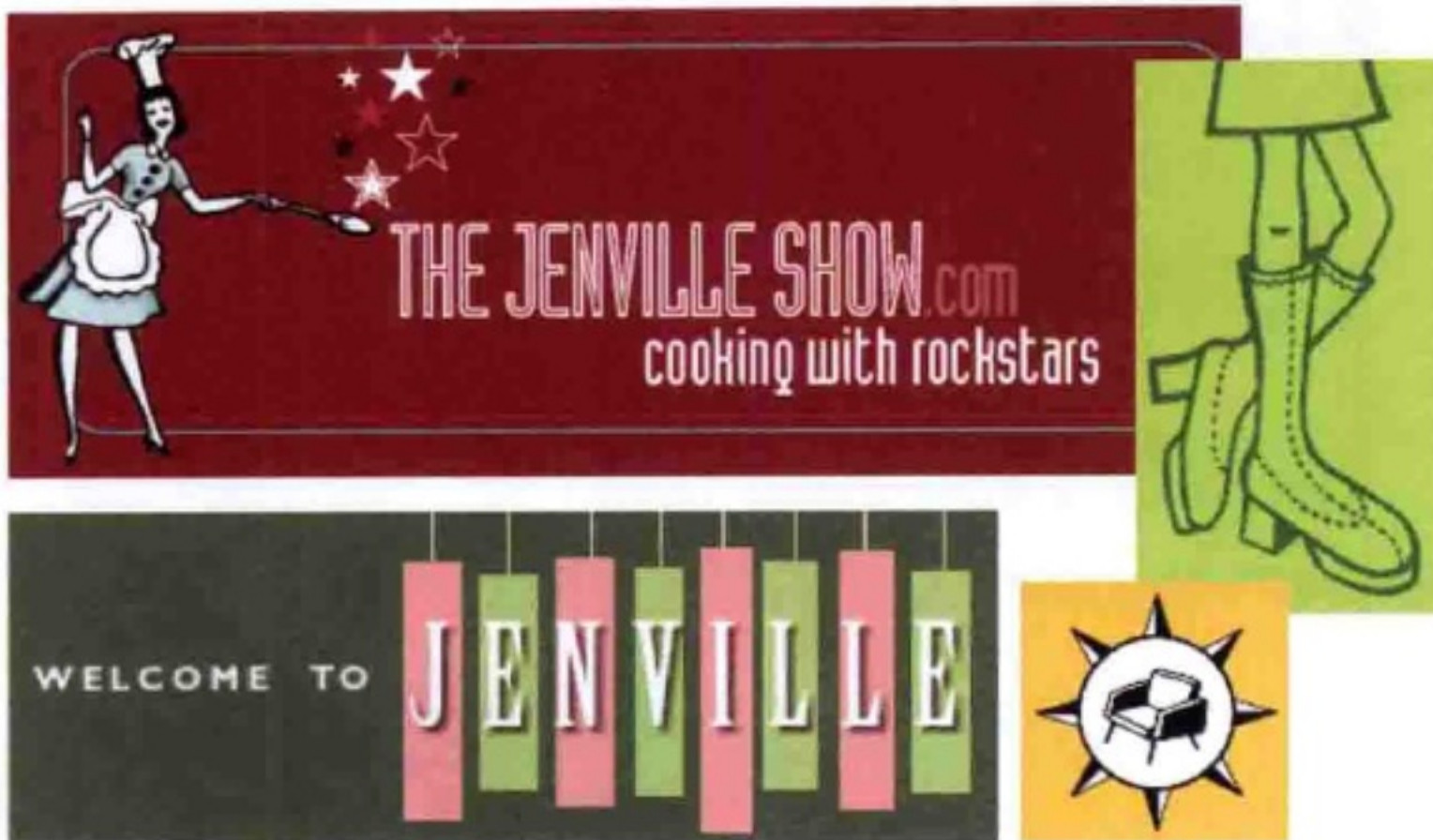


图21-1：对于主要由平色和硬边组成的图像，GIF格式非常合适

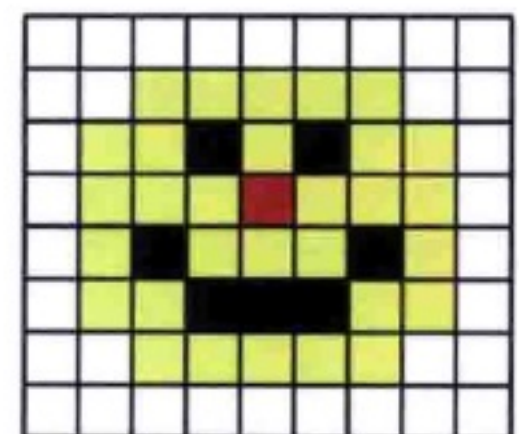
1	1	1	1	1	1	1	1	1	1
1	1	3	3	3	3	3	1	1	
1	3	3	2	3	2	3	3	1	
1	3	3	3	4	3	3	3	1	
1	3	2	3	3	3	2	3	1	
1	3	3	2	2	2	3	3	1	
1	1	3	3	3	3	3	1	1	
1	1	1	1	1	1	1	1	1	

索引颜色图像中的像素包含图像颜色表的数值引用。

1	2	3	4
---	---	---	---

颜色表

颜色表将数值与RGB色彩值匹配。这是2位图像，4色彩的图。



图像在位置上显示色彩。



图21-2：一个2位图像和它的颜色表

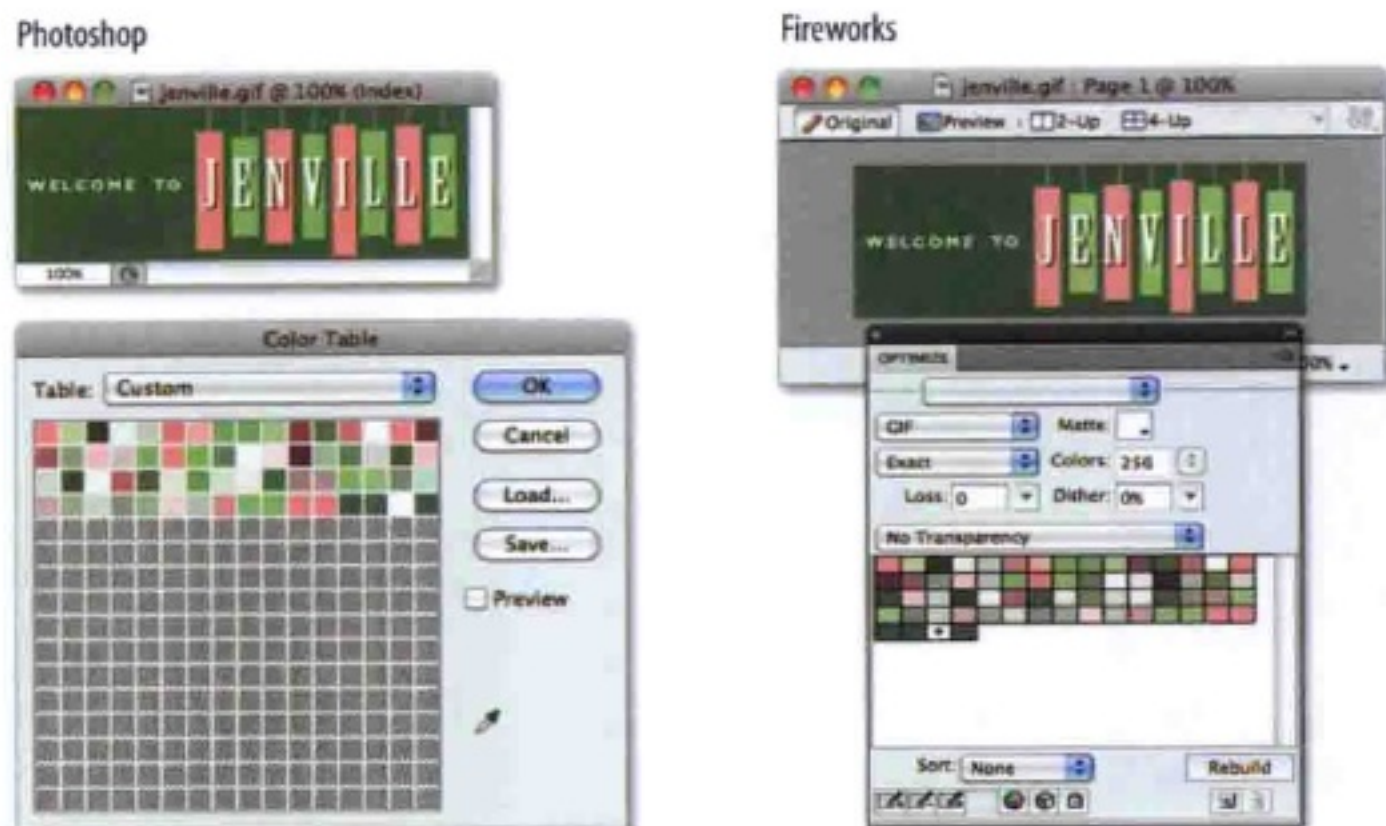


图21-3: Photoshop和Fireworks中的颜色表, 显示图像使用的64像素色彩

GIF压缩会给各个像素都保存为一条描述。



在一个色彩渐变的图像中, 它不得不为每个像素保存信息。信息描述越长, 意味着文件尺寸越大。

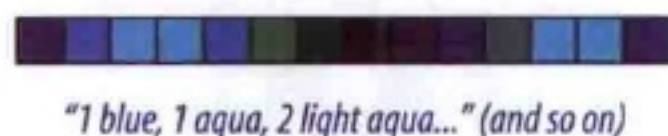


图21-4: GIF图像使用的LZW压缩的简化示例

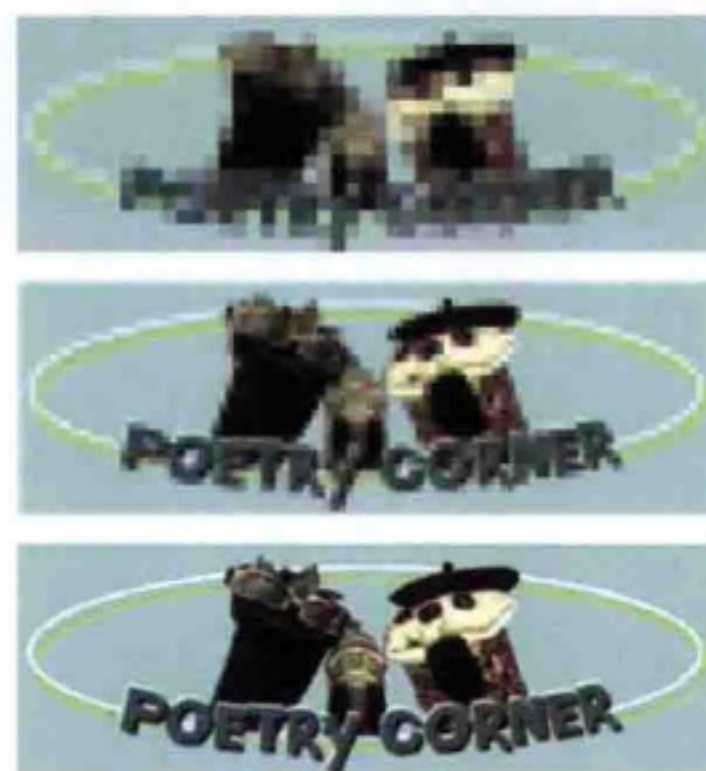
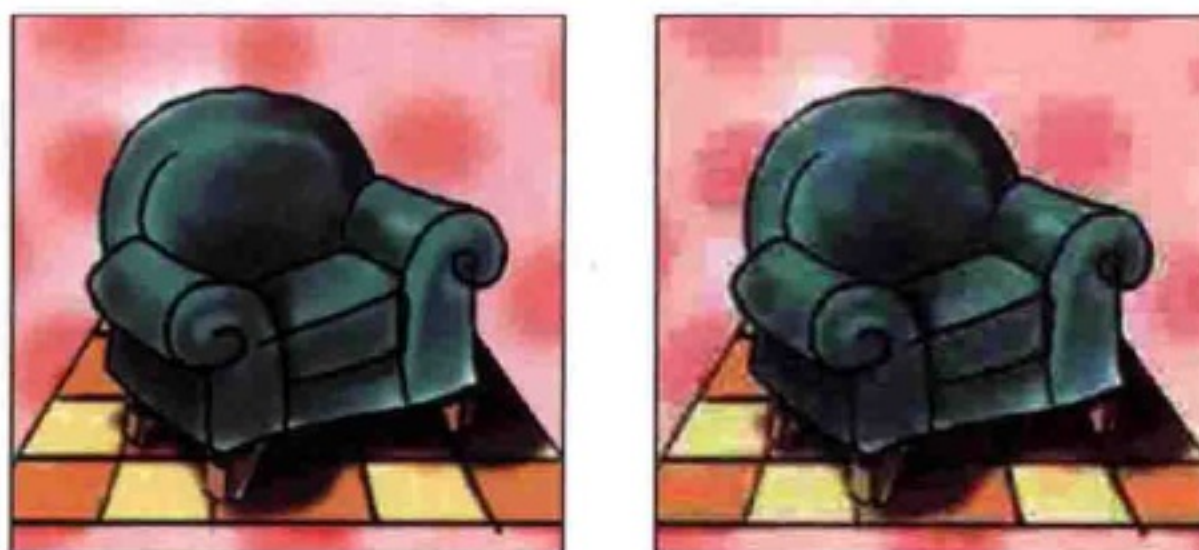


图21-6: 交错GIF以一系列路径显示, 每个路径都比上次清楚



原图

高压缩图

图21-9: JPEG压缩会丢弃图像细节来获得更小文件尺寸。使用高压缩率, 图像质量会受损, 如右图所示



图21-10：渐进式JPEG图像以一系列路径渲染

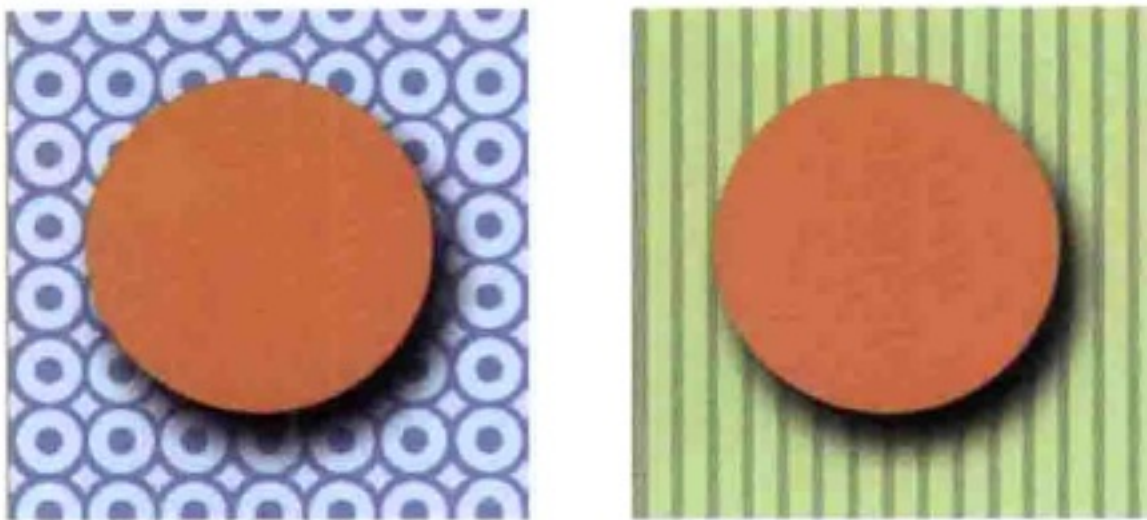


图21-11：Alpha通道透明度允许多级透明度，就如图中橙色圆圈PNG显示的下阴影

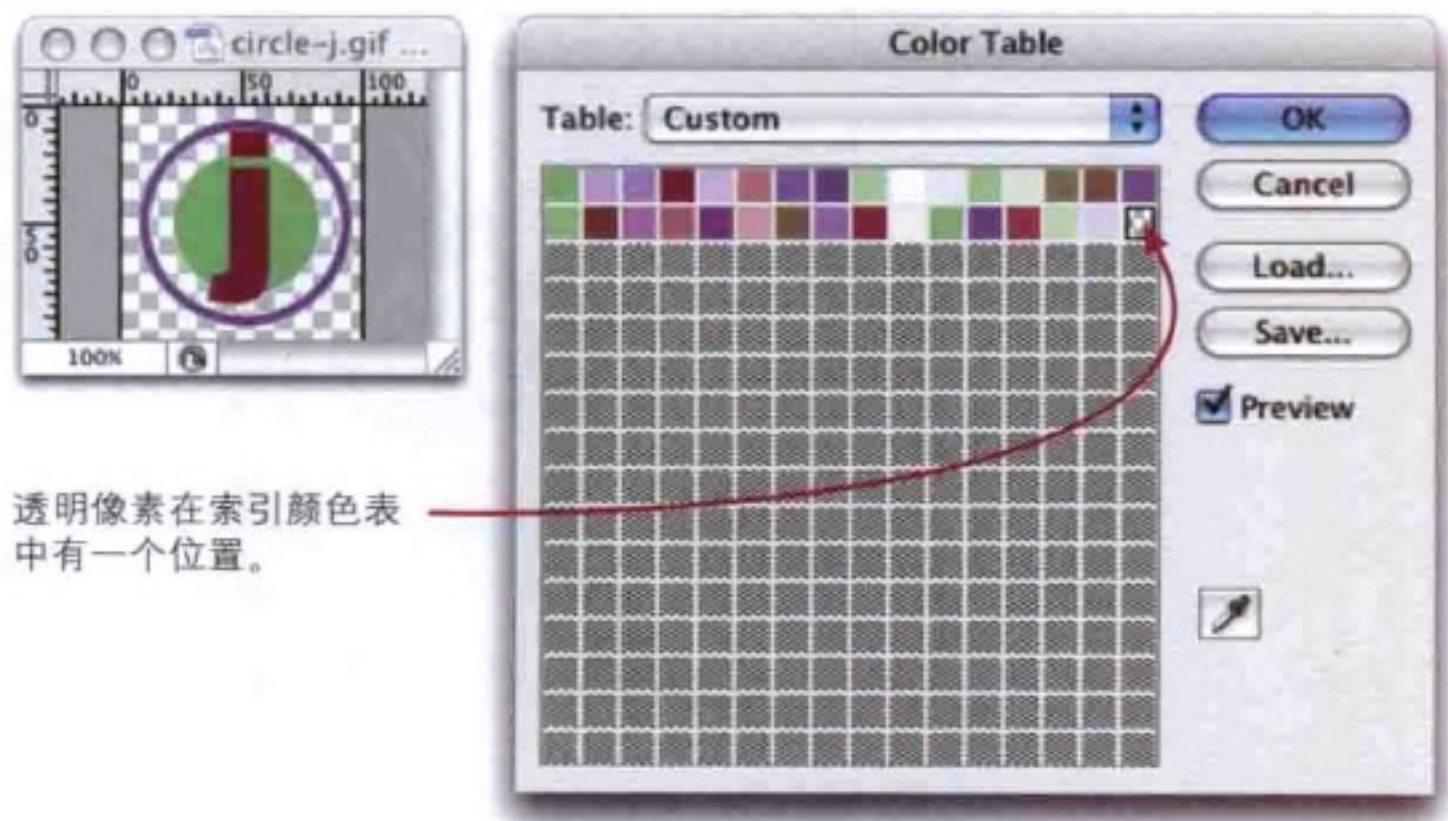


图21-20：在索引颜色表中，透明被当做一种颜色



原先的透明图像

Alpha通道的黑色区域对应透明图像区域；白色区域是不透明的；灰色区域是中间值。

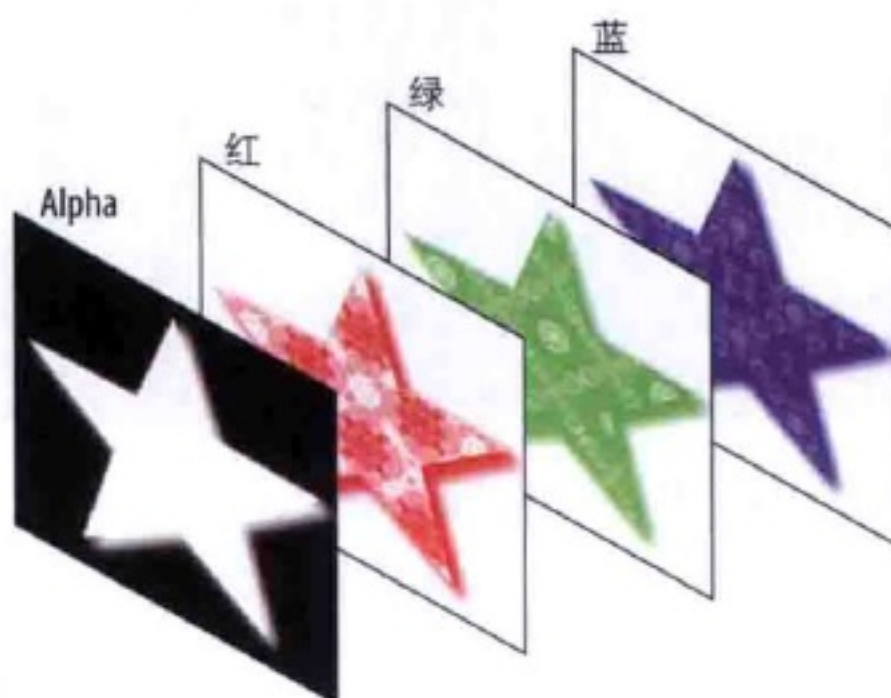


图21-21：透明信息保存在24位PNG的单独通道中

tiger.svg



10x



tiger.gif



10x



图21-30：矢量SVG图像放大时不会损失质量



SVG 4 U!

图21-31：基本的SVG图像，svg4u.svg

ABOUT THIS SITE

A



C



B



D



E

图21-33：为每个图像选择最佳的格式

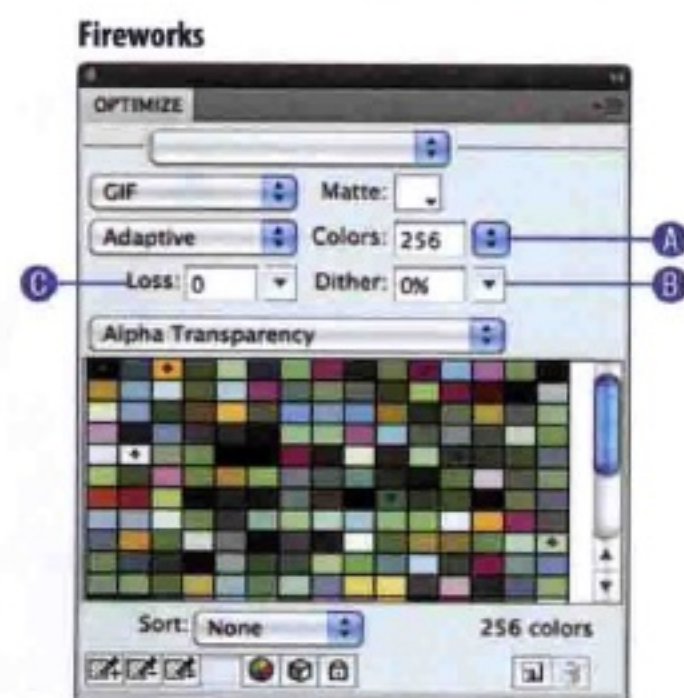
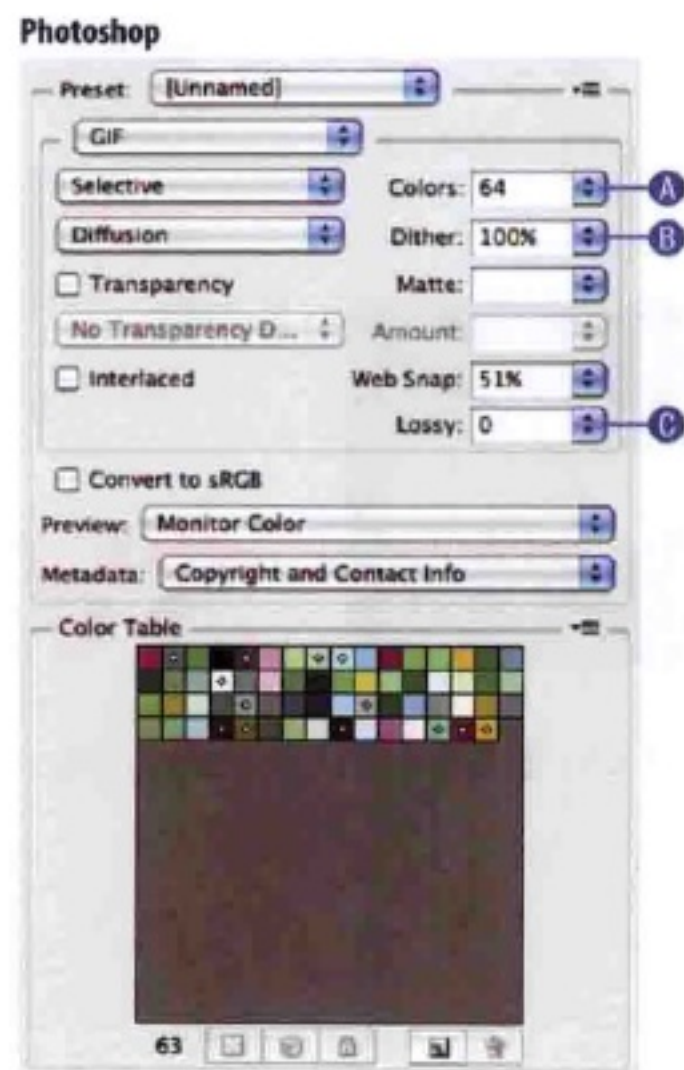


图22-2: Photoshop和Fireworks中的GIF优化选项



256 colors: 21 KB



64 colors: 13 KB



8 colors: 6 KB

图22-3: 减少图像中的颜色数，从而减小文件尺寸

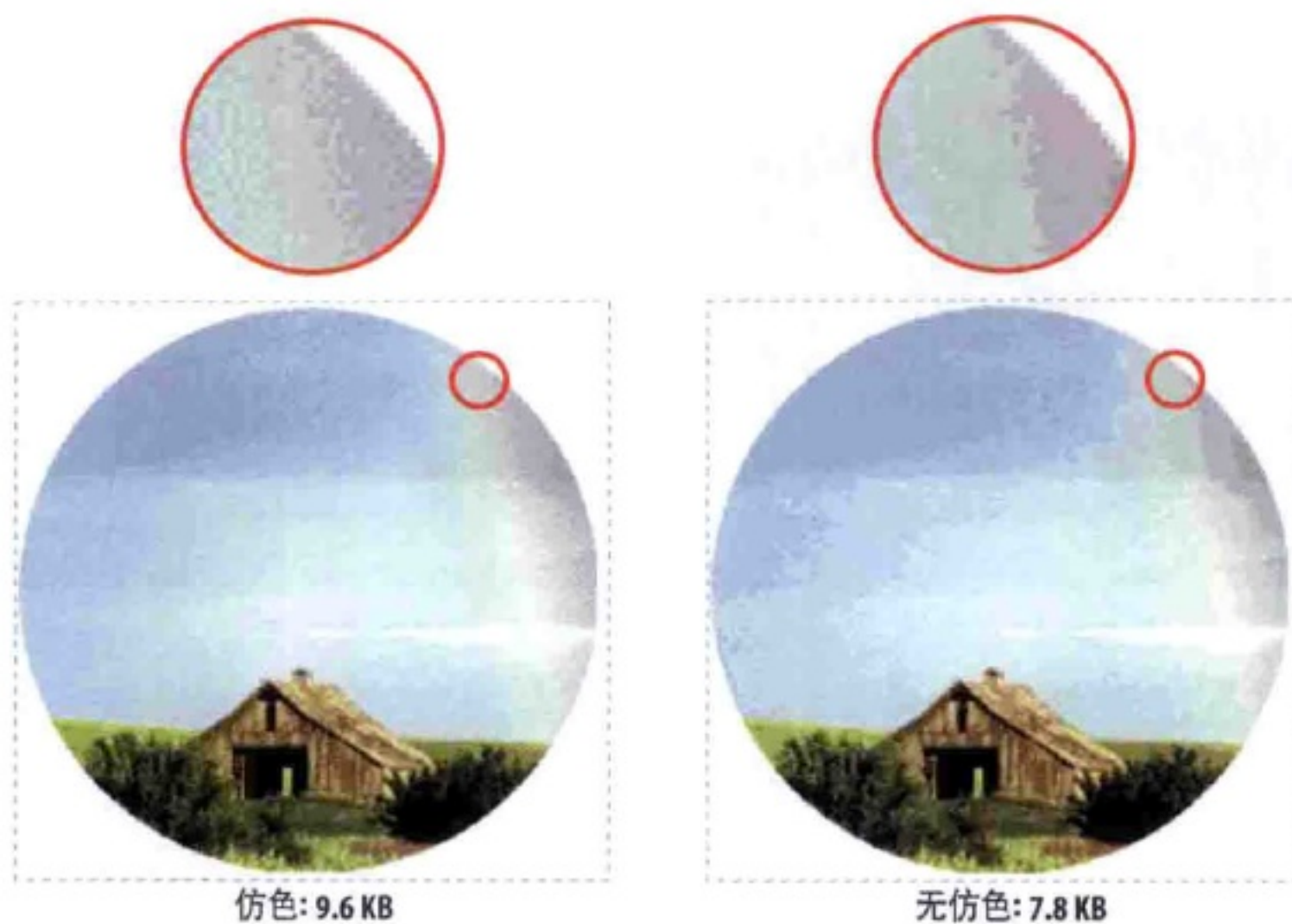


图22-4：关闭或减少仿色可以减少文件尺寸。两个图像都有32像素颜色，并且使用一个适合的色板



图22-5：Photoshop中应用有损设置的文件尺寸和不应用有损设置的图像尺寸



这个GIF有渐变混色，而且为256色。它的文件大小为19KB。

甚至当我把颜色数减少到8时，文件大小还得7.6KB。

当使用平色创建时，文件大小仅为3.2KB。

图22-6：利用GIF的压缩特点来设计，可以减小文件尺寸

gradient.jpg (12 KB)



detail.jpg (49 KB)

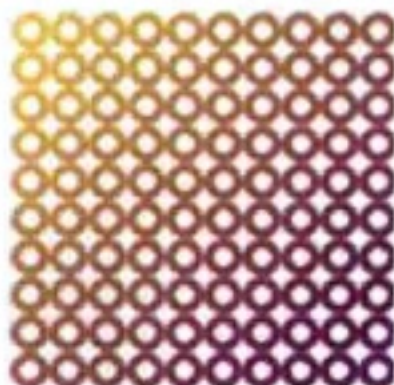


图22-10：相比有硬边和细节的图像，JPEG压缩在平滑、混色上的压缩效率更好

chair.jpg



在JPEG中，平色改变，并且图像斑驳，在JPEG压缩中，细节丢失了。

chair.gif



在GIF中，平色和细节都保留了下来。

图22-11：相同的平色图像保存为JPEG和GIF

Photoshop



100% (42.2 KB)



80% (22.3 KB)



60% (13.6 KB)



40% (8.5 KB)



20% (6.3 KB)



1% (3.7 KB)

Fireworks



100% (51.5 KB)



80% (12.3 KB)



60% (7.7 KB)



40% (5 KB)



20% (1.8 KB)



1% (1.2 KB)

图22-12: Photoshop和Fireworks中不同压缩级别的对比

Photoshop



Fireworks



图22-16: Photoshop 和 Fireworks中的 PNG-24和PNG-8设置

O'Reilly Media, Inc.介绍

O'Reilly Media通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了Make杂志，从而成为DIY革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar博客有口皆碑。”

——Wired

“O'Reilly凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference是聚集关键思想领袖的绝对典范。”

——CRN

“一本O'Reilly的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照Yogi Berra的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去Tim似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

译者序

翻译本书第3版时是2008年，到现在已经有五年时间了。在这五年中，全球的通信行业、互联网行业都发生了巨大的变化，而中国也紧跟时代的步伐。五年里，苹果的iPhone、iPad风靡全球，三星、HTC在安卓智能手机上也不断推陈出新；五年里，随着智能手机普及，WCDMA、CDMA2000、TD-SCDMA等3G技术也得到了很大发展和普及，国内的3G手机用户在2013年超过了3亿，并且还在不断增长；五年里，国外的Twitter、Facebook，国内的新浪微博、微信都成为移动设备上必备的应用。在这浪潮之巅，埋首故纸堆无益于融入这个时代。

原以为本书已经过时的时候，机械工业出版社的编辑告诉我们第4版已经推出。当看到第4版时，我们感到Jennifer Niederst Robbins不愧是在Web设计领域有20年经验的资深专家，她一直紧跟时代的特点，不断地学习、使用最新的技术。在第4版中，Jennifer Niederst Robbins讲述了很多移动设备网页设计的经验和技巧，并且根据最新的HTML 5、CSS3标准，讲解了很多最前沿的元素和属性；针对台式机、平板电脑、智能手机以及上网辅助设备等情况，讲解了最前沿的自适应网页设计和网页图像优化技术；而且与以往版本不同，本书没有避开JavaScript，而专门用两章来讲述了JavaScript的基础和应用，毕竟现在，没有JavaScript的网页，就太过平庸了。

通过翻译本书，我们深感本书覆盖范围之广，实践经验之多，前沿技术之精，是其他类似的一般书籍无法比拟的。也难怪，作者自己都说“本书无论对学生，还是对有一定背景和技术水平的专业人士，都是理想的选择”。

能够完成本书，首先得感谢机械工业出版社的编辑，是他们提供了这个机会，使我们得以首先分享到这一最新实践和技术的佳肴；其次要感谢王炼、张宓，他们对上一版作出了很大的贡献；感谢张榕祯、陈彦先、丁宗

云、刘星辉、孙涛，他们在本书的初译、样例代码测试、验证、中文软件使用等方面都提供了很大的帮助和支持。

最后，希望本书能够满足你的需要，切实解决你的疑惑，并提高你的实践经验和技能。在此提示，作者为本书提供了一个网页，提供了刊物、样例等信息，阅读时可以查阅，以获取最新的信息：http://oreil.ly/learn_web_design_4e。

目录

前言	1
----------	---

第一部分 开始起步

第1章 我从哪里开始.....	9
-----------------	---

我该从哪里开始呢.....	10
Web设计师是做什么的	10
我需要学习哪些语言	16
我需要购买些什么.....	19
你学会了什么	24
自我测验	25

第2章 Web是如何工作的.....	27
--------------------	----

互联网与Web.....	27
提供你的信息	27
关于浏览器.....	29
网页地址（URL）	30
网页结构	32
小结	36
自我测验	38

第3章 Web设计基本概念.....	39
--------------------	----

眼花缭乱的设备	40
与标准一致.....	41

逐步提高42

自适应Web设计43

可访问性——所有用户，一个网站46

连接速度的要求（站点性能）48

自我测验50

第二部分 HTML结构化标记

第4章 创建简单网页（HTML概述）53

 一个网页，一步一步来53

 开始之前，启动文本编辑器54

 第1步：从内容开始57

 第2步：文档结构化59

 第3步：确定文本元素62

 第4步：添加图像65

 第5步：使用样式表改变外观68

 当网页出错时69

 验证你的文档70

 自我测验71

 元素回顾：文档结构72

第5章 标记文本73

 段落74

 标题74

 列表77

 更多内容元素80

 组织网页内容83

 内联元素综述88

 泛型元素（div和span）99

 特殊字符103

 小结104

 自我测验106

 元素回顾：文本107

第6章 添加链接	109
href特性	110
链接到Web上的网页	111
站内链接	112
以新浏览器窗口为目标	122
邮件链接	123
电话链接	124
自我测验	125
元素回顾：链接	126
 第7章 添加图片	 127
首先，关于图片格式的信息	127
img元素	128
窗口中的窗口	134
自我测验	135
元素回顾：图片	135
 第8章 表格标记	 137
如何使用表格	137
最小表结构	139
表头	142
合并单元格	143
表格可达性	146
小结	148
自我测验	150
元素回顾：表格	150
 第9章 表单	 151
表单如何工作	151
表单元素	153
变量和内容	155
重要表单控件综述	156
表格的可访问性	175
表单布局和设计	177

自我测验	179
元素回顾：表单	180
第10章 HTML 5	185
在通往XHTML 2的路上发生的有趣的事	186
标记部分	188
遇到API	193
视频与音频	195
canvas	202
小结	205
自我检测	206
 第三部分 表现层的CSS	
第11章 CSS入门	211
CSS的益处	211
样式表如何工作	213
大概念	218
继续使用CSS	224
自我测验	226
 第12章 格式化文本（使用更多选择器）	229
字体属性	229
改变文本颜色	246
更多选择器类型	247
字行设置	253
下划线和其他的“修饰”	256
改变字母大写	256
空格	257
文本阴影	258
改变列表的数字编号	263
自我测验	265
CSS回顾：字体和文本属性	267

第13章 颜色和背景（附加更多选择器和外部样式表）	269
指定颜色值	269
前景色	276
背景色	277
使用不透明度	278
伪类选择器简介	279
伪元素选择器	282
属性选择器	284
背景图像	287
快捷背景属性	296
像彩虹一样（渐变）	299
最后，外部样式表	303
自我测验	306
CSS回顾：色彩和背景属性	307
 第14章 盒子思想（填充、边框和空白边）	309
元素盒子	309
指定盒子尺寸	310
填充	316
边框	320
空白边	332
分配显示角色	337
给盒子添加阴影	338
自我测验	340
CSS回顾：基本盒子属性	342
 第15章 浮动与定位	345
普通流	345
浮动	346
定位基础	360
相对定位	361
绝对定位	362
固定定位	372

自我测验	373
CSS回顾：浮动和定位属性	374
第16章 使用CSS进行网页布局	375
网页布局策略	375
网页布局技术	382
使用浮动元素的多栏布局	382
定位布局	394
自上而下的栏目背景	397
自我测验	400
第17章 过渡、变换和动画	401
很容易做到（CSS过渡）	401
CSS变换	412
关键帧动画	422
自我测验	425
CSS回顾：过渡、变换和动画	427
第18章 CSS技术	429
一个干净的石板（CSS重置）	429
图像代替技术	431
CSS sprite	432
样式化表单	436
样式化表格	442
简单的自适应Web设计	445
小结	456
自我测验	456
CSS回顾：表格属性	458
 第四部分 JavaScript行为	
第19章 JavaScript简介	461
什么是JavaScript	461
给网页添加JavaScript	464
脚本剖析	465

浏览器对象	478
事件	478
综合应用	481
自我测验	483
第20章 使用JavaScript	485
遇见DOM	485
polyfills	493
JavaScript库	497
小结	501
自我测验	502
 第五部分 创建Web图像	
第21章 Web图像基础	505
图像来源	505
格式简介	508
图像尺寸和分辨率	520
使用透明度	524
SVG简介	531
小结	536
自我测验	536
第22章 精简Web图像	539
通用图像优化策略	539
优化GIF图像	541
优化JPEG图像	544
优化PNG图像	550
优化到指定大小	551
小结	552
自我测验	552
附录A 答案	555
附录B CSS3选择器	583

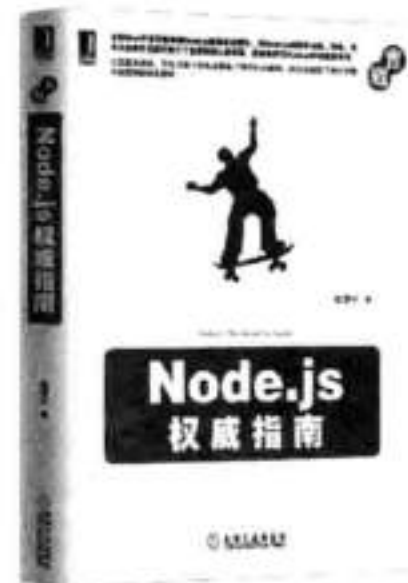
Web前端开发&设计经典

HTML5&CSS3篇

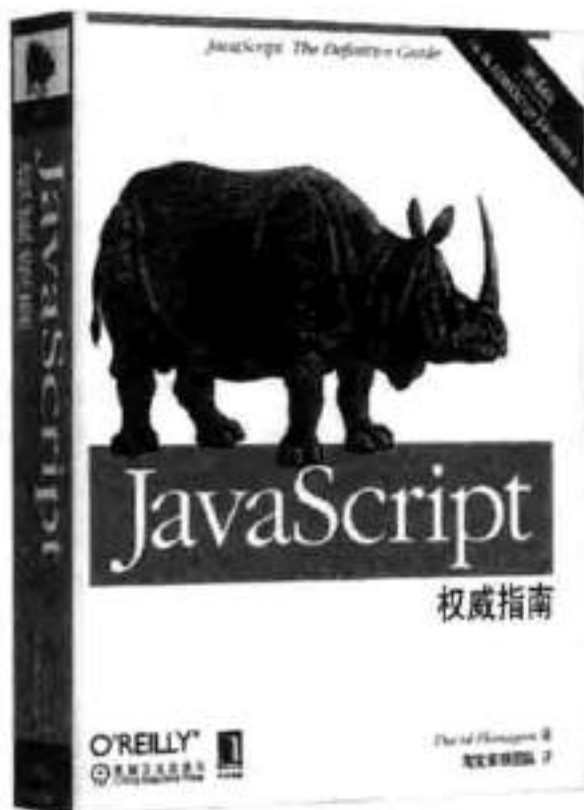


Web前端开发&设计经典

框架篇



推荐阅读



JavaScript权威指南（原书第6版）

从1996年以来，本书已经成为 JavaScript 程序员的《圣经》。

第6版涵盖HTML5和ECMAScript 5。很多章节完全重写，以便跟得上当今的最佳Web开发实践。该版本的新增章节描述了jQuery和服务端JavaScript。

对于那些希望学习Web编程语言的有经验的程序员和希望精通JavaScript的当前JavaScript程序员，本书最适合不过了。

深入理解PHP：高级技巧、面向对象与核心技术（原书第3版）

国际知名Web开发专家和技术畅销书作家最新力作，PHP领域经典著作。

从编程技巧、面向对象和扩展三个角度系统讲解和总结了成为中高级PHP程序员应该具备的技术和技能，包含大量实用案例，极具实践指导意义。

如果你已经具备一定的PHP编程基础，想使开发效率更高，想把应用做得更好，那么这本书应该是你需要阅读的。本书旨在为想修炼成为高级PHP程序员的中初级PHP程序员提供实用的方法和建议。

前言

非常欢迎阅读第4版。

从上一版到现在发生太多的事情了！浏览器厂商和开发社区确定了Web标准，一切看起来刚刚尘埃落定时，移动互联网时代又带来了巨大的变化。随着智能手机和平板电脑的出现，Web在小屏幕和便携式设备上终于有了自己的位置。当我们想方设法使网站体验更好时，形势已经为Web设计者和程序员带来了新的挑战。

就像我所写的那样，很多这样的挑战依然处于争论中，例如，如何把适当的图片发送到适当的设备上。这是Web设计难以置信且充满活力的时代，到处都在试验和合作。这使我回想起了1993年，一个属于Web的西部蛮荒岁月，当时我刚开始从事Web设计生涯。太多的问题需要解决！无限的可能！坦率地说，在这个时代，要确定不断变化的科技，并把技术编写成书是相当棘手的事情。因此我也尽我所能来指出还不确定的论题，并且提供一些在线的资源链接，来帮助你与时俱进。

现在有两个新标准——HTML 5（超文本标记语言的第5次重大修订）和CSS3（层叠样式表，第3级）——提供给我们，当我写本书的上一版时，这些还都只是传说。现在本书的HTML部分使用了当前的HTML 5标准，而且涵盖了CSS3标准的发展，新增了一章（第17章）来讲解动画、过渡、变换的交互。我们的工具使我们能够比几年前做得更多、更有效。

最后，由于JavaScript已经成为Web开发的重要部分，所以本书新增了两章（第19章和第20章）来介绍JavaScript的语法和应用。我并不是JavaScript专家，但是我有幸找到了一位专家。第19章是由Mat “Wilto” Marquis编写的，他是Filament Group的设计者和开发者，也是jQuery Mobile团队的成员，同时还是《A List Apart》的技术编辑。

本书的同步网站

本书的同步网站learningwebdesign.com提供了本书练习的材料、可以下载的文章、链接列表以及本书的引用和其他一些精彩内容。

与本书前三版一样，无论初学者的背景是什么，可以是经验丰富的平面设计师，或者是寻找更好出路的程序员、办公室助理、刚毕业的大学生、家庭主妇，只要想学习如何设计Web，本书都可以满足你的需求，解决你的疑问。我已尽我所能将初学Web设计的课程经验编辑成书，还提供了练习和测试，这样你就可以得到实践经验，从而进步。

无论你是专门阅读这本书，还是把它作为一个Web设计课程的辅助读物，我都希望它能给你一个良好的开端，并预祝你学得开心。

本书组织结构

本书分为五部分，每一个都是Web开发的一个重要部分。

本书的排版约定

本书使用如下排版约定

斜体 (*Italic*)

用于URL、电子邮件地址、文件名与目录名以及用于强调。

等宽字体 (`constant width`)

用于代码示例和键盘输入的指令。

等宽斜体 (*`Constant width italic`*)

用于表示属性的位置标示符和样式表中的属性值。

第一部分：开始起步

第一部分为本书后面的学习奠定了基础。我从关于Web设计的一些重要信息开始讲，包括你可能会扮演的各种角色，你可能会学到的技术和工具。你会学到HTML和CSS，并学习Web和网页一般如何工作。我也会向你介绍一些重要的概念，以便让你领会现代Web设计师的思想。

第二部分：HTML结构化标记

第二部分的章节涵盖每个元素和属性的本质以及语义结构，其中包括HTML 5中新引入的元素。我们将讨论标记文字、链接、图片、表格和表单。第二部分结束时会深入讨论HTML 5，并且说明了它与以前标准的不同之处。

第三部分：表现层的CSS

在第三部分，你将会学习使用CSS来改变文本的外观，为页面创建多列布局，甚至添加基于时间的动画、交互等。这部分也会介绍常用的CSS技术，包括如何创建一个使用自适应Web设计的页面。

第四部分：JavaScript行为

在第四部分，Mat Marquis从JavaScript的语法纲要开始讲解，帮助你逐步学习变量、函数。你也会学习到JavaScript的使用方式，包括DOM脚本，以及已有的JavaScript工具，如polyfills和库，即便你还没有完全准备好编写代码，这些工具都可以使你更快地使用JavaScript。

第五部分：创建Web图像

第五部分介绍了Web适用的各种文件格式，而且描述了如何进行优化，才能使文件尺寸尽可能小。

致谢

我要感谢编辑Simon St. Laurent，与他的合作非常愉快，我期待着以后有更多的合作机会。还要感谢我的合作者Mat Marquis (*matmarquis.com*)，他使JavaScript有趣，他虽有控制欲，却也有着很好的合作精神。

我得到了许多聪明可爱的人的支持。我要感谢我的主技术审核者Aaron Gustafson (*easy-designs.net*)、Joel Marsh (*thehipperelement.com*) 和 Matt Menzer，他们从百忙中抽出了很多时间，以确保章节的细节准确。还要感谢提出了宝贵意见的评论者：Anthony Calzadilla、Danny Chapman、Matt Haughey、Gerald Lewis、Jason Pamental和Stephanie Rieger。

我有幸认识了这个领域的很多领导者，他们的书籍、文章、演讲、幻灯片以及同我的交往，使我能够一直走下去。没有这些天才的帮助，我也无法完成本书，他们是（按英文字母顺序排列）：Dan Cederholm、Josh Clark、Andy Clarke、Chris Coyier、Brad Frost、Lyza Gardner、Jason Grigsby、Stephen Hay、Scott Jehl、Scott Jenson、Tim Kadlec、Jeremy Keith、Sanders Kleinfeld、Peter-Paul Koch、Bruce Lawson、Ethan Marcotte、Eric Meyer、Karen McGrane、Shelley Powers、Bryan Rieger、Stephanie Rieger、Remy Sharp、Luke Wroblewski和Jeffrey Zeldman。

完成一本书需要一个团队，我想衷心感谢这些伙伴的贡献：Melanie Yarbrough（生产编辑和校对员）、Genevieve d'Entremont（文字编辑）、Rebecca Demarest（图像制作）、Newgen（页面布局）、Ellen Troutmen Zeig（索引制作）、Randy Comer（封面设计）和Ron Bilodeau（装帧设计）。

最后，我想感谢Edie Freedman（有史以来最好的老板），她显示了充分的耐心。而我也终于能对我最亲爱的宝贝Jeff和Arlo说：“我回来了！”

关于作者

Jennifer Robbins于1993年开始Web设计，当时她是全球网络导航器（Global Network Navigator）的平面设计师，这也是第一个商业网站。除了这本书之外，她还写了《Web Design in a Nutshell》和《HTML 5 Pocket Reference》——这本书也是一个iOS应用——都是由O'Reilly出版的。过去，Jennifer已经在许多会议（包括Seybold 和South By Southwest）上发言，并且她已经在罗德岛普罗维登斯的约翰逊和威尔士大学开始教授Web设计。她现在是O'Reilly媒体的数码产品设计师。她对信息架构、交互设

计、网站制作、应用程序和电子书表现了浓厚的兴趣。在空闲的时候，Jennifer喜欢独立摇滚、烹饪和教育子女。

使用代码示例

本书提供代码的目的是帮你快速完成工作。一般情况下，你可以在你的程序或文档中使用本书中的代码，而不必得到我们的许可，除非你想复制书中很大一部分代码。例如，你在编写程序时，用到了本书中的几个代码片段，这不必得到我们的许可。但若将O'Reilly图书中的代码制作成光盘并进行出售或传播，则需获得我们的许可。引用示例代码或书中内容来解答问题无须许可。将书中很大一部分的示例代码用于你个人的产品文档，这需要我们的许可。

如果你引用了本书的内容并标明版权归属声明，我们对此表示感谢，但这不是必需的。版权归属声明通常包括：标题、作者、出版社和ISBN，例如：“*Learning WebDesign, Fourth Edition* by Jennifer Robbins. Copyright 2012 Littlechair, Inc., 978-1-449-31927-4”。

如果你认为你对示例代码的使用已经超出上述范围，或者你对是否需要获得示例代码的授权还不清楚，请随时联系我们：permissions@oreilly.com。

联系我们

我们已尽力检验本书所提供的信息，尽管如此，仍不能保证本书完全没有瑕疵，而且网络世界的变化之快，也使得本书永不过时的保证显得不太可能。请让我们了解你找到的错误，以及你对后续版本的建议，我们都将虚心接受读者的指教。

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街2号成铭大厦C座807室（100035）
奥莱利技术咨询（北京）有限公司

申请会员或订阅图书目录，请发送邮件至：

info@oreilly.com

技术问题或评论本书，请发送邮件至：

bookquestions@oreilly.com

本书网页列出了勘误表、实例、其他附加信息以及后续版本计划。

<http://www.oreilly.com/catalog/0636920023494>。

后记

我们会关注读者的意见、我们自己的试验和销售渠道反馈的结果。独特的页面可以配合我们独特的方法技术主题，从而为这些枯燥的主题增加一些生机和活力。

开始起步

第一部分

本部分内容

第1章

我从哪里开始

第2章

Web是如何工作的

第3章

Web设计基本概念

第1章 我从哪里开始

Web的存在已经有20多年了，先后经历了令人兴奋的早期扩张、网络泡沫的破裂、创新的重生。有一件事情是肯定的，那就是网络作为沟通和商业的工具正在成为现实。而且不仅如此，网络也在影响着智能手机、平板电脑、电视等。懂得如何设计网页的程序员迎来了前所未有的机会。

通过教授Web设计课程和工作的经历，我可以满足所有对Web设计感兴趣的人的需求，无论这些人拥有什么样的背景。在此，请允许我介绍一下：

“我有17年的印刷设计师从业经验，现在我所有的客户都需要Web设计服务。”

“我曾在一个小办公室里做过秘书。我的老板曾要求我制作一个很小的内部网站以方便员工分享公司信息。”

“我做程序员很多年了，但是我希望在视觉设计上一展身手。我感觉Web是一个很好的学习新技术的机会。”

“我是一个艺术家，我想知道如何在线获取我的画和雕塑的样品。”

“我曾高中简单接触过Web设计，我想它可能成为我今后的生活依赖。”

无论动机如何，都会遇到一个相同的问题：“我该从哪里开始呢？”看起来有太多的东西需要学习，所以往往不知道该如何起步。但是你必须从某个地方开始。

本章试图从学习曲线的角度来回答准备学习的人最常遇见的问题。本章将介绍与Web设计相关的规则、技术和工具。

本章内容

我该从哪里开始

Web设计师是做什么的

我需要学习哪些语言

我需要购买什么软件和设备

我该从哪里开始呢

根据你的背景和目标，你的起点也应该有所不同。然而，对于每个人来说，第一步应当是理解Web和网页运作的基本原理。现在你看的这本书将教给你正确的学习方法。一旦了解了基本知识，Web上和书店图书馆有很多资源可以方便你进一步学习特定的领域。

同样，学习Web设计也有许多不同目标，有的只是为了创建一个自己的站点，有的却要把它当做职业。你可以是能够提供全套服务的一个网站开发者，也可以只把它当做一项技术。你完全可以自由选择。

如果你学习Web设计只是出于爱好，或者你需要发布一个或两个Web项目，你会发现要完成手头的任务，你可能需要做的仅是个人研究（就像阅读本书），利用现有的模板，或者购买一个可视化的Web设计工具（如Adobe Dreamweaver），可以帮你完成手头的任务。还有许多继续教育项目提供关于Web设计和制作的一些入门级课程。

如果你想把Web设计作为职业，那么你就需要把技术锤炼到专业水平。雇主可能并不需要你具有Web设计学位，但是他们期望看到一些样本站点，以便证明你的技术和经验。这些站点可能只是课堂作业、个人项目，也可能只是为了做些小生意而制作的简单站点。重要的是，这些站点看起来要专业，在后台使用清晰的HTML、样式表和可能的脚本。获得一个基本的工作，并且和一个团队一起工作，可以了解大型站点是如何构建的，同时可以帮助你确定Web设计的哪些方面是你真正喜爱的。

Web设计师是做什么的

多年来，术语“Web设计”已经成为包罗万象、包含许多不同的学科的一个学科，包括用户体验设计、文档标记和编程等。本节介绍最基本的一些东西。

如果你正在设计自己的一个小网站，你会需要多个头衔。当然你可能没有注意到。想一下，日复一日的家居生活，需要你既是兼职厨师、清洁工、会计师、外交家、园丁，还得是建筑工人，但对你来说，都只是你的生活而已。同样，作为一个Web设计师，你需要是一名兼职平面设计师、作家、HTML作者和信息架构师，但对你来说，它都只是“网页制作”而已。除此之外没什么可担心的。

如果你不具备某些技能，你可以聘请一些专家。例如，我自1993年以来就创建网站，当我的客户需要交互功能时，我仍然聘请程序员和多媒体

我只是需要一个博客

并不是一定要成为一个Web设计师才能在Web上发布你的文章和图片。你可以使用免费的或者非常便宜的博客托管服务来建立一个自己的“博客”或者个人旅行发布站点。这些服务会提供一些模板，你不必学习HTML就可以使用（当然学学HTML也没有坏处）。下面是三个最受欢迎的系统：

- WordPress (www.wordpress.com)
- Blogger (www.blogger.com)
- Tumblr (www.tumblr.com)

你可以在Squarespace (www.squarespace.com) 上了解其他的简单Web设计方法和托管服务。

看一看

“Web设计”综合了各种学科，包括：

- 视觉（图像）设计
- 用户界面和体验设计
- Web文档和样式表设计
- 脚本和编程
- 内容策略
- 多媒体

开发者。这可以让我专注于我做的那一部分（对我来说，主要工作是内容组织、界面和视觉设计）。

大型网站几乎都是由一个团队创建的，从几人到数百人。在这种情况下，每个团队成员专注于网站建设过程的一个方面。如果你的情况确实如此，你可能只会用到你目前的一套技能（写作、Photoshop和编程等）或者可以把你的兴趣投入新的方面。

我把“Web设计”分成了多种角色和责任，共有四个大类：设计、开发、内容策略和多媒体。

设计

啊，设计！这听起来很简单，但即使这样简单的要求，在建立站点的过程中也已经分成多种专业。虽然这里有一些有关设计一个站点的新工作的介绍，但请记住学科往往是重叠的，自称“设计师”的人往往需要承担多种职责（如果不是全部）。

用户体验、交互和用户界面设计

通常情况下，我们认为设计是应该思考东西看起来怎么样。在Web上，第一件事是设计的站点是如何工作的。在选取颜色和字体前，重要的是要确定站点的目标、站点将如何使用以及访客如何在网站内浏览。这些任务属于学科的交互设计（Interaction Design, IxD）、用户界面（User Interface, UI）设计和用户体验（User Experience, UX）设计。这些设计中有很多重叠的关系，由一个人或一个小组来处理这三种设计屡见不鲜。

交互设计师的目标是尽可能使站点方便、高效、愉快地使用。用户界面设计与交互设计密切相关，用户界面设计往往是更狭隘地专注于页面的功能组织以及具体工具（按钮、链接、菜单等），用户通过这些来导航内容或完成任务。

在Web设计领域的新生的职位是用户体验设计师。UX设计师需要更全面的看法，以确保整个站点的体验是良好的。UX设计基于观察和访谈理解用户的需求。Donald Norman（他创造了UX这个词）认为，用户体验设计包括“用户与产品交互的各个方面：它是如何感知、学习和使用的”。对于一个网站或应用程序来说，这包括视觉设计、用户界面、内容的质量和信
息，甚至是站点的整体性能。用户体验必须符合该组织的品牌和业务目标，才能取得成功。

IxD、UI或UX设计师可能提供的文档包括：

如果你没有兴趣成为一个无所不能的个人Web设计师，你可以选择专业化和加入团队来工作，也可以做一个自由职业承包商。

用户研究和测试报告

了解用户的需求、欲望和局限对于站点或Web应用程序设计成功至关重要。这种设计方法围绕用户的需求，是以用户为中心的设计（User Centered Design, UCD），它是当代设计的核心。站点设计通常从研究用户开始，包括访谈和观察，以更好地理解站点如何解决问题，或者将如何使用。对于设计师来说，在设计过程的每个阶段进行用户测验以确保其设计的可用性是非常普遍的。如果用户很难弄清楚在哪里可以找到的内容或如何移动到一个过程的下一个步骤，就需要回到绘图板重新设计了。

线框图

线框图对每个内容类型和窗口都使用大纲来显示网页结构（见图1-1）。线框图的目的是表明屏幕如何划分，并指出导航、搜索框、表单元素等功能和内容都放在哪里，因此线框图没有任何装饰或图形设计。它们通常只是注释，仅仅说明事情应该如何工作，以便于开发团队知道如何去做。

站点图

站点图表示站点作为一个整体与单独的网页之间如何链接的结构。图1-2显示了一个非常简单的站点图。某些站点图填满整个墙壁！

故事板 and 用户流程图

故事板是从一个典型用户（UX的术语是一个角色）的角度来看站点或应用程序的路径。它通常包含一个脚本和“场景”，场景包括屏幕的观点或用户与屏幕的交互。其实，故事板旨在说明完成任务所需要的步骤、可能的选项，并且介绍了一些标准的页面类型。图1-3显示了一个简单的故事板。用户流程图是显示如何连接一个站点或



图1-1：线框图

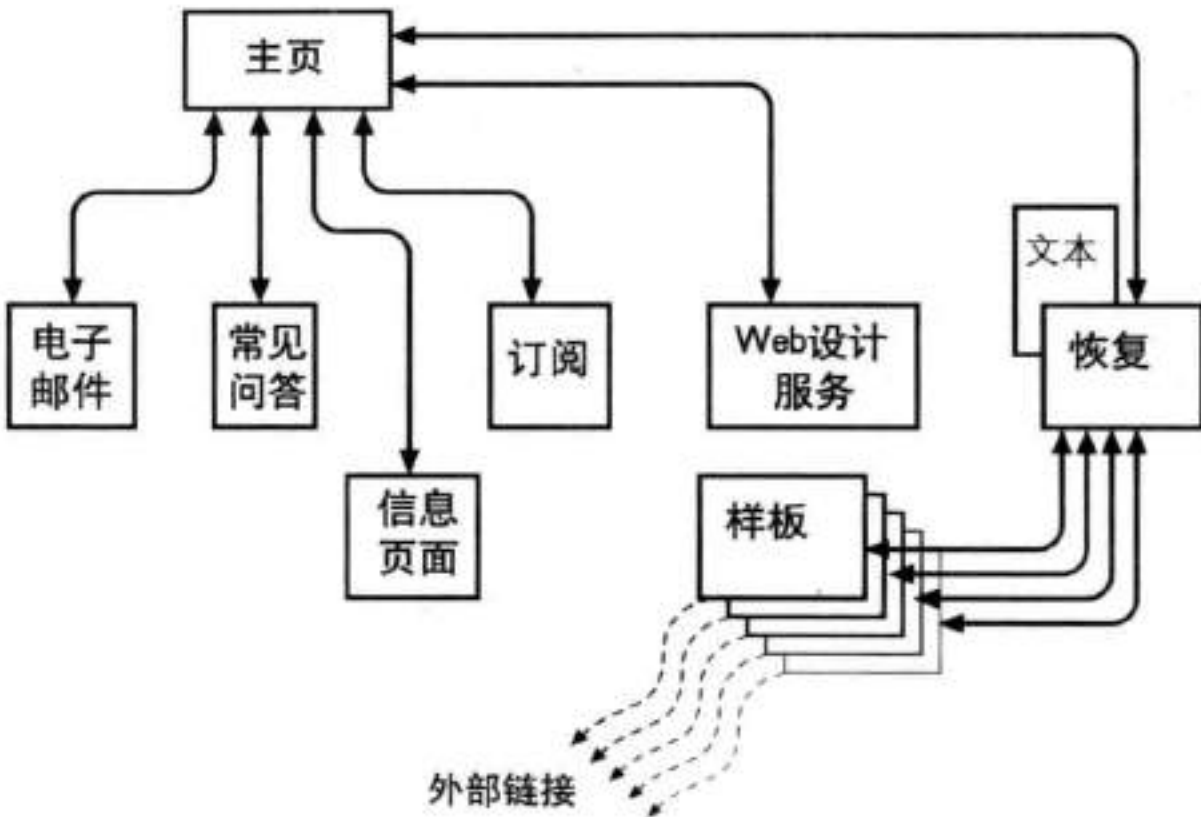


图1-2：一个简单的站点图

应用程序的各个部分，它往往专注于技术细节，而不是讲故事。例如，当用户这样做时，它会触发服务器上的某个功能。对设计师来说，使用用户流程图来描述某个过程的步骤是很常见的，如描述用户注册或网上支付过程中的步骤。

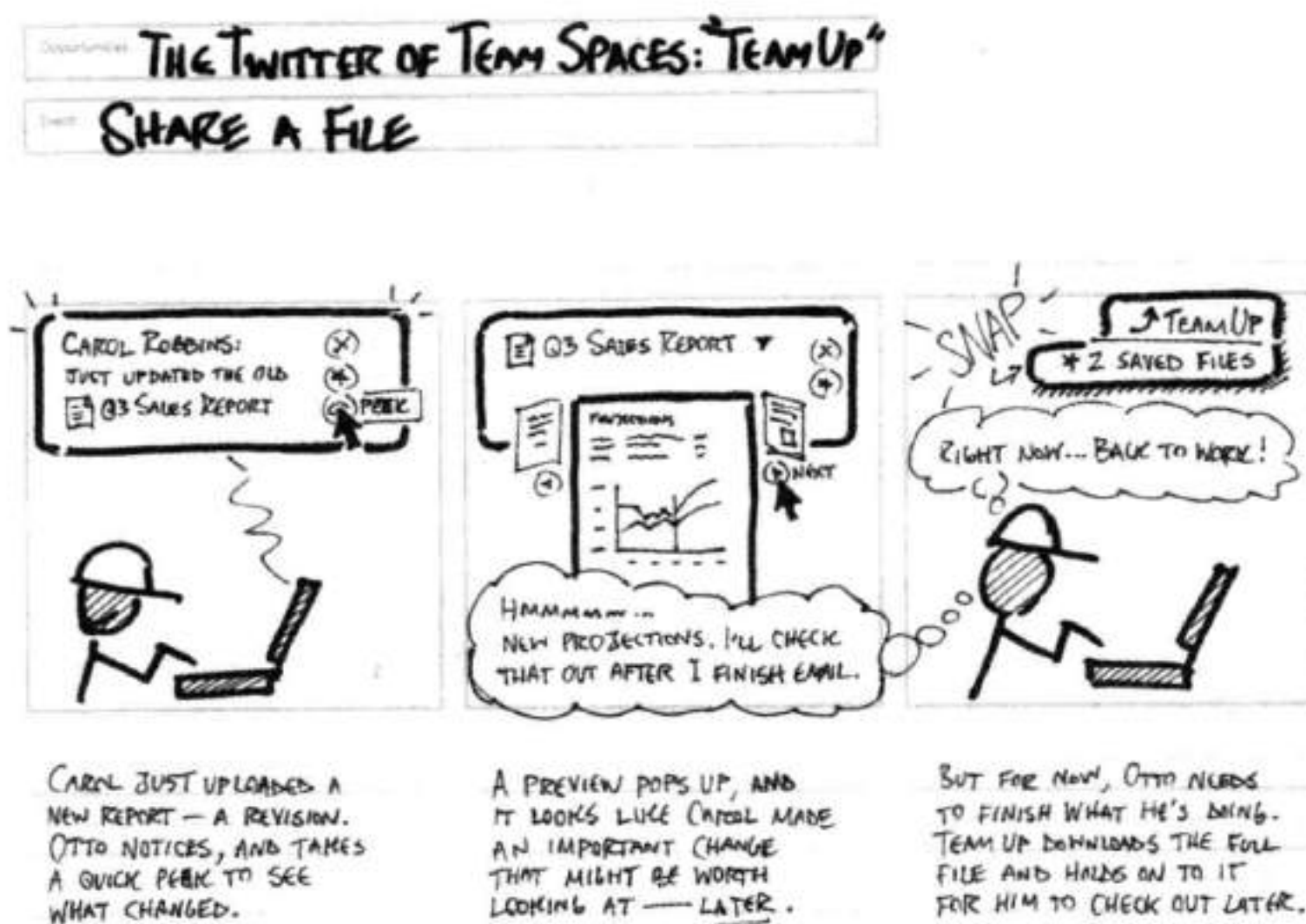


图1-3：一个典型的情节故事板，Brandon Schauer画

视觉图像设计

由于Web是一个视觉媒体，所以网页需要注意展现和设计。平面设计师决定了站点的视觉和感觉：标志、图像、类型、颜色、布局等，以确保网页有良好的第一印象，并且与品牌和内容相匹配。视觉设计师通常会完成网页外观的草图，如图1-4所示。他们也会负责制作图像文件，并且优化，以便于在网上传播图像（第22章将介绍图像优化技术）。

如果你有兴趣或专业做商业站点的视觉设计，我强烈建议参加平面设计培训和精通Adobe Photoshop（这是行业标准）或Adobe Fireworks。如果你已经是一名平面设计师，你可以很容易地将你的技术适用于Web。但仍然需要扎实理解HTML、CSS和其他Web技术。因为大多数站点至少有几张图片，甚至仅仅作为业余爱好者的Web设计师也至少需要知道如何创建和编辑图像。

需要再次指出的是，所有这些职责可能落在一名设计师的肩上，包括创建站点的外观和功能。但对于有较多预算的较大的站点来说，可以在设计过程中找到自己的特殊位置。



图1-4：一个简单网站的外观和视觉的速写

样式模板

控制网页外观和感觉的另一个方法是使用样式模板。样式模板提供了配色方案、品牌元素、内容和UI设计的样例，而不必把它们应用在特定的页面布局上。基本的想法是要为页面提供一致的视觉语言。想了解更多内容，可以阅读Samantha Warren的文章“Style Tiles and How They Work” (www.alistapart.com/articles/style-tiles-and-how-they-work/)，并且可以访问他的网站 styletiles.com 来下载一个模板。

注意：许多视觉设计师将自己的设计转化为HTML和CSS文档。实际上，普遍的争论是，为什么唤醒自己是一名Web设计师，几乎所有的人都认为如果你的编程技能与设计技能一样好，你的工作前景将会更美好。

开发

Web设计过程中有相当一部分工作涉及创建文件、样式表、脚本和图像，以创建和故障排除。在Web设计公司，负责创建网站（或动态组装的模板）页面文件的是开发或生产部门。

Web开发人员可能不设计站点的外观或结构，但他们确实需要与设计师良好沟通以了解目标站点的目标，然后他们会提出解决方案以满足这些目标。

创作、造型和脚本/编程等都属于开发工作。

创作/标记

创作是指准备内容发送在Web上的过程，或者更具体地说，是指使用HTML标记来标记内容，标记HTML描述了的内容和功能。如果你想做一个Web开发人员，你需要仔细了解HTML及其如何在各种浏览器和设备上应用。随着HTML规范不断发展，这意味着你将需要紧跟最新和最佳的实践和机会，以及最新的错误和限制。好消息是这并不是很难上手，而且你可以逐步增加自己的技能。我们将在第2章介绍HTML，然后在本书的第二部分详细讨论。

造型

在Web设计中，浏览器中页面的外观样式规则由CSS（层叠样式表）控制。在本书的第三部分，我们会深入讲解CSS（包括“层叠”的意思），但现在只须知道，在现代Web设计中，页面的外观由HTML标记之外的东西来处理。同样，如果你对web开发工作感兴趣，也知道CSS及其是如何通过浏览器支持（或不支持）以确保成为你的工作描述的一部分的。

脚本和编程

当Web已经发展成为应用程序的一个平台，编程变得前所未有的重要。JavaScript是可以让页面元素完成工作的一种语言。它给页面的元素乃至浏览器窗口本身增加了行为和功能。

其他Web相关的编程语言还包括PHP、Ruby、Python和ASP.NET，它们一般运行在服务器上，在发送到用户的浏览器之前将处理数据和信息。参阅侧栏“前端与后台”了解更多信息。

Web脚本和编程需要高超的电脑编程技术。虽然很多Web程序员具有计算机科学学位，但是依然有很多开发者是自学成才的。开发者往往从复制并修改已有的脚本开始学习，然后在工作中逐渐提高他们的编程能力。

如果你从没编程的经验，学习起来可能会有些难度。

教授Web编程超出了本书的范围。JavaScript将在第19章中讲述（如果全部讲解JavaScript的内容，整本书都写不完）。不用编程也完全可以创建出令人满意的、内容丰富的、设计良好的站点，所以Web设计爱好者没有必要灰心丧气。然而，如果你需要通过表格收集信息或者根据要求发送信息，那么你的团队就需要有一个程序员了。你也可以看看你的托管公司是否提供简易的、封装好的一些功能。

内容策略和创作

内容策略和创作在这里列为第三项，但在实际的网站创建过程中应该是第一位的，站点的内容是关键的问题。任何拥有“Web设计师”头衔的人必须知道支持用户获取内容、信息或功能的一切过程。此外，良好的写作可以帮助我们创建更有效的用户界面。

当然，有人需要创建内容和维护——不要低估成功做到这一点所需的资源。此外，我想请你注意现代Web开发团队中与内容相关的两项工作：内容策划师和信息架构师（Information Architect, IA）。

当内容写得不正确时，站点就无法充分发挥作用。一个内容策划师要确保站点上的每一个文字位，从长长的解释文本到按钮上的标签说明文字，都支持公司的品牌形象和营销目标。内容策略也可能在更大和持续时间上延伸到数据建模和内容管理上，如规划内容重用和更新时间表。

前端与后台

你可能听设计师和开发者说过他们关注网页设计的前端或者后台。

前端设计

“前端”设计指与浏览器显示直接相关的设计。本书就主要讲述网页前端设计。

下面是前端设计的内容：

- 图像设计和制作；
- 界面设计；
- 与网页用户体验相关的信息设计；
- HTML文档和样式表开发；
- JavaScript。

后台设计

“后台”设计指运行在服务器端的程序，这些程序可以生成动态和互动的页面。一般来说，后台设计由有经验的程序员完成，但是作为网页设计师了解后台功能是非常有益的。

下面是后台设计的内容：

- 服务器中信息组织方式的设计；
- 表单处理；
- 数据库编程；
- 内容管理系统；
- 其他使用PHP、JSP、Ruby、ASP.NET、Java等编程语言的服务器端应用程序。

万维网联盟

万维网联盟（简称W3C）是负责监督Web技术发展的组织。这个组织在1994年成立于麻省理工学院（MIT），创始人为网络的发明者Berners-Lee。

最初，W3C主要关注于HTTP协议和HTML的发展。现在，W3C通过开发能够集成于同一个基础平台上的很多技术和协议，来为Web将来的发展奠定基础。

如果需要准确解答任何Web技术方面的问题，可以看看W3C的网站：

www.w3.org

如果要了解更多关于W3C的信息，请参见：

www.w3.org/Consortium/

信息架构师（也称为信息设计师）负责有逻辑地组织内容，以便于检索。她可能负责搜索功能、站点图，以及如何在服务器上组织内容和数据。UX和UI设计的信息架构不可避免地交织在一起，一个人或团队来执行所有角色的情况并不鲜见。

多媒体

Web很酷的一个特点是可以在站点上添加多媒体元素，包括声音、影像、动画和互动游戏。你可能想要在你的Web设计工具栏上添加多媒体技术，如音频、视频编辑或Flash开发（参见关于Flash的一些知识）或者成为这方面的专家。如果你并不想成为一个多媒体开发者，你完全可以雇用一个。Web开发公司经常寻找精通多媒体工具、有良好的视觉感觉和灵感，并且能够设计具有创意的多媒体的人。

我需要学习哪些语言

如果你是一个把时间花在Photoshop和Illustrator上的视觉设计师，你需要暂时放下学习如何用文本设计的念头，但我向你保证，这是很简单的一项工作。很多创作工具可以加快创作过程，我们将在本章的后面讨论。

下面是与网页开发相关的技术列表。至于学习哪种语言或者技术，取决于你对自己的定位。但是我建议每一个人都了解HTML 5和层叠样式表，

关于Flash的一些知识

Adobe Flash（以前叫Macromedia Flash、FutureSplash）是特别为Web创建的一种多媒体格式。Flash使你能够创建全屏的动画片、互动图片、集成音频和视频，甚至创建脚本游戏和应用程序。然而近年来随着技术发展，Flash的使用量开始减少，比如：

- 苹果决定在其iPhone和iPad上不支持Flash，而支持HTML 5；
- Adobe决定不再对手机浏览器支持Flash；
- HTML 5中新的可编程的canvas元素提供了与Flash相同的功能；
- 批评者认为Flash有时候无法满足用户的要求。比如，用户只想知道餐馆周日是否营业，却只能看着餐馆网页上的电影和配乐发呆；
- 播放Flash媒体需要插件也让开发者厌恶。

事实上，网页专家经常会说“Flash已死”，但是尽管Flash有些被削弱了，但是如果使用恰当，Flash还是有一些优势的：

- 由于Flash使用矢量图形，所以它的文件体积小，可以在不丢失细节的情况下调整大小；
- 它是一个流媒体格式，所以电影播放速度迅速，并且可以边下载边播放；
- 你可以使用ActionScript来添加功能和互动，在前端使用Flash来动态生成内容或提供电子商务功能；
- Flash插件遍布个人电脑，所以Flash在桌面浏览器的支持是可靠的；
- 虽然HTML 5有前途并且发展迅速，但是截至写此书时，它还无法比拟Flash的性能。

Flash不可能在一夜之间消失，但Adobe也确实在考虑使用HTML 5替代Flash。

而且如果你想做前端设计，JavaScript也是必备的技能。网络技术更多地倾向于服务端配置、数据库、站点性能，但是一般来说这些都不是前端的开发工作（虽然了解基本的后台技术没有什么坏处）。

看一看

Web相关技术如下：

- HTML
- CSS
- JavaScript和 DOM 脚本
- 服务器端脚本和数据库管理

HTML

HTML（超文本标记语言）是用来创建网页文件的语言。当前业界使用的HTML版本有多个：包括有广泛基础的HTML4.01，新的健壮的HTML5也获得更多的关注和浏览器支持。两个版本都有一个严谨的实现，即XHTML（扩展HTML）。XHTML本质上采用同样的语法规则。我们将在第10章中讲二者的区别。

HTML并不是一种编程语言，它是一种标记语言，也就是说，它是用来识别和描述一个文件中各个组件的系统，比如标题、段落和列表。标记表示文档的底层结构（你可以把它当做详细的、机器可读的纲要）。要编写HTML，你只需要耐心和常识，并不需要编程技巧。

学习HTML的最好方法是动手编写一些页面，我们将在练习中这么做。如果你最终从事Web制作，那么你将天天与HTML打交道。即便只是一个爱好者，了解底层是如何工作的，对你也是很有帮助的。令人鼓舞的是，学习基本知识非常简单。

层叠样式表（CSS）

HTML用来描述网页的内容，而层叠样式表（CSS）用来描述你希望这些内容如何显示。在Web设计界，页面的外观称为“展示”。也就是说，字体、色彩、背景图片、行间距、页面布局等都是由CSS来控制的。使用最新版的CSS3可以给你的页面添加一些特殊效果和简单动画。

CSS不仅提供页面在传统的浏览器和屏幕上展示的方法，还提供控制文件在其他媒介上展示的方法，如在打印机和掌上设备中展示。对于明确说明非视觉展示的文件，它也提供了相应的规则，如要用屏幕阅读器阅读时，该如何发音（虽然这些并不被良好支持）。

样式表也是自动制作的很棒的工具，因为你可以通过修改单独的样式表

我们经常会看到HTML和XHTML统称为（X）HTML。

注意： 当本书讲到“样式表”时，往往指的是层叠样式表，也就是万维网的标准样式表语言。

文件改变站点所有的页面。现在几乎所有的浏览器都不同程度地支持样式表。

虽然单独使用HTML就已经可以发布网页，但如果使用样式表，你就可以不必拘泥于浏览器的默认样式。如果你想专业地设计网站，精通样式表是必须的。

我们将在第三部分更详细地讨论样式表。

JavaScript/DOM脚本

JavaScript是一种脚本语言，在网页中，它可以用来添加互动和行为，包括（只是其中一部分）：

- 检查表单输入，以确保输入有效；
- 更换一个元素或整个站点的风格；
- 使浏览器记住有关用户的资料，方便他们下次访问；
- 构建界面窗口，如扩展菜单。

JavaScript是常用于操纵网页元素或者某些浏览器窗口功能的一种语言。当然还有其他的Web脚本语言，但是JavaScript（也称为ECMAScript）是标准和最普遍应用的脚本语言。

你可能也听过术语DOM脚本，它的使用与JavaScript相关。DOM是对象模型（Document Object Model）的简称，它指的是用JavaScript（或其他脚本语言）访问并控制的网页元素的标准列表。DOM脚本是一个更新的术语，它原来是指DHTML（动态HTML），现在这种说法已经过时了。

编写JavaScript就是编程，所以如果你先前没有编程经验，学起来可能会很费力。很多人通过阅读书籍，跟随并修改现有的程序来自学JavaScript。大多数Web创作工具都带有标准的脚本，你可以用这些脚本来实现常用的功能。

专业的Web开发者需要知道JavaScript，然而很多设计师都是依赖开发者来在他们的设计中添加JavaScript行为的。所以，虽然JavaScript很有用，但并不是所有的Web设计师都需要会编写。教授JavaScript超出了本书的范围；但是如果你想学习，我推荐你看看《Learning JavaScript》，Shelley Powers著（O'Reilly出版社，2006）。

服务端编程

有一些网站的页面都是静态HTML文件和图片，但是大多数商业站点都有

Web设计层划分

当代的Web设计是可视化的，它由三个独立的“层”组成。

用（X）HTML标记的文件内容组成了结构层（Structure Layer）。它是其他层赖以存在的基础。

一旦文件的结构准备就绪，就可以添加样式表信息来控制内容显示的方式。这一层就是展示层（Presentation Layer）。

最后是行为层（Behavior Layer），它包括完成页面互动性的脚本。

非常高级的功能，如表单处理、动态生成页面、购物车、内容管理系统和数据库等。这些功能都是由运行在服务器上的特殊Web程序来实现的。有很多编程语言和架构（括号中列出的）可以用来创建Web程序，包括：

- PHP (CakePHP、CodeIgniter、Drupal)
- Python (Django、TurboGears)
- Ruby (Ruby on Rails、Sinatra)
- JavaScript (Node.js、Rhino、SpiderMonkey)
- Java (Grails、Google Web Toolkit、JavaServer Faces)
- ASP.Net (DotNetNuk、ASP.Net MVC)

开发Web程序是程序员的任务，并不需要所有的Web设计师都编程。这就意味着你也可以在客户端提供这些功能。你不必自己编写所有的程序就可以实现购物车、内容管理系统、邮件列表和留言簿等功能。

我需要购买些什么

毫无疑问，专业的Web设计师需要很多工具，包括硬件和软件。我的学生最常问的一个问题是：“我需要购买些什么呢？”我不能告诉你去买什么，但是我可以介绍一下这个行业的典型工具。

虽然在这里我列出的是最受欢迎的商业软件，但是它们同时都有免费软件和共享软件，你可以下载这些软件（试试看CNET的网站*Download.com*）。稍做一些努力，不必花很多钱，你就可以完全创建一个网站。

XML简介

如果你经常接触Web设计，你肯定听过XML (eXtensible Markup Language, 扩展标记语言)。XML并不是一种特定语言，而是创造其他标记语言的一套强有力的规则。

举一个简单的例子，如果你要出版一套食谱，你可能就需要使用XML来创建一套自定义的标记语言，包括元素<ingredient>、<instructions>和<servings>，这些元素是用来描述食谱文件中信息的类型。正确标注后，这些信息就可以当做数据处理。事实上，XML已经成为应用程序之间分享数据的强有力的工具。尽管XML最初设计是用于Web中，但是由于它的数据处理功能，它在Web之外的领域也产生了巨大影响。越来越多的软件，如Microsoft Office、Adobe Flash和Apple iTunes都开始

幕后使用XML文件。

但是依然有很多种XML语言应用在Web中。最流行的是XHTML，它是根据XML（在第10章会更多地介绍XHTML）的规则来重写HTML得到的。还有一种语言是RSS (Really Simple Syndication或RDF Site Summary, 简易信息集合)，它可以把你的内容当做数据来共享，并且可以使用RSS种子阅读器阅读；SVG (Scalable Vector Graphics, 可伸缩矢量图形) 用标签来描述几何形状，MathML是用来描述数学符号的。

作为一个Web设计师，你最先接触的XML可能就是用来创作文件的XHTML，或者是添加在网站上的RSS或SVG图像。开发新的XML语言是程序员或XML专家的职责。

在Mac上运行Windows

如果你的苹果电脑使用的是运行DS X (Leopard或更高版本)的Intel芯片,你就不需要使用一台单独的电脑来测试Windows环境了。苹果在Leopard OS X发布中提供了免费的启动程序,允许在重启时进入Windows系统。

Mac OS上也提供了几个虚拟机(VM),你可以使用它们在Windows和Mac OS之间切换。这些虚拟机有:

- VMFusion (www.vmware.com/fusion) 是一个商业软件,你可以下载使用免费版。
- Parallels Desktop也是Mac OS中的商业软件,它也提供了免费版。
- Oracle VirtualBox (virtualbox.org) 是一个免费软件,它支持运行多个寄生操作系统,包括Windows和多版Unix。

所有的虚拟机都需要你购买Microsoft Windows操作系统,当然只是软件,不需要购买一整台机器。

设备

为了有一个舒适的Web开发环境,我建议使用如下装备:

一台性能稳定的最新的计算机。操作系统可以是Macintosh、Windows或者Linux。在专业的Web开发公司,创意部门往往使用苹果机。虽然最好使用性能非常好的电脑,但是组成网页的文件是非常小的,所以在电脑的配置上不必太奢侈。除非你需要编辑音频和视频,否则你完全没必要担心你的计算机配置不是最新和最强的。

额外的内存。由于你需要同时使用多个应用程序,所以你的计算机上最好有足够的内存,这样才能同时运行几个对内存要求很高的程序。

大一些的显示器。这个要求虽然不是必需的,但是高清显示器看起来会舒服得多,尤其对于视觉设计师而言。(我见到过写代码的开发者在11寸的MacBookAir上干得也不错。)显示器越多,你可以同时打开的窗口和控制面板数就越多。在进行设计决策时,你也可以同时看到多个页面。

如果你使用的是高清显示器,那么务必记住你是为用小的显示器和低清晰度的用户做设计的。

一台扫描仪和/或一台数码相机。如果你要制作自己的图像和文本,你就需要一些工具来创建它们。我知道有一个设计师有两台扫描仪:一台是性能好的扫描仪,另一台一般的扫描仪则只用来扫描一些像死鱼或者生锈的平底锅一样的东西。

辅助计算机。很多Web设计师发现,有一台使用不同平台的计算机做测试是非常有用的(例如,如果你使用Mac设计,那么就在PC机上测试)。由于Mac操作系统的浏览器和Windows操作系统上的浏览器工作原理不同,所以在尽可能多的环境测试是非常重要的,尤其要在Windows操作系统上测试。如果你只是居家工作的Web设计爱好者,可以在你朋友的计算机上测试。Mac用户应该看看侧栏的“在Mac上运行Windows”。

移动设备。Web已经进入移动时代了!这意味着在智能手机或平板设备的移动浏览器上测试站点的外观和性能绝对至关重要。你可能已经有了自己的智能手机。如果你的预算不足以购买多个平台的设备,可以问问你的朋友看看是否可以花几分钟,用他们的设备来查看你的站点。我有一个Web开发者朋友,他在当地的移动运营商商店来查看他设计的网站(虽然你可能很快就会不受欢迎)

软件

创建网页的软件应有尽有。在早期，只有适用于打印机的软件工具。但是今天，已经有很多工具是专门用来进行Web设计的，从而使得Web设计的效率大幅度提高。虽然无法列出每一个可用的软件版本，但是我可以介绍最常用和行之有效的Web设计工具。注意，你可以在公司主页下载这些程序的试用版本，将在本章侧栏“看一看：流行的Web设计软件”中列出这些公司网站。

Web创作

Web创作工具类似于桌面出版工具，只是它最终的产品是网页（一个HTML文件及其相关的文件）。这些工具提供一种“所见即所得”的界面和捷径，为你节省编写HTML和CSS的大量时间。这些工具可以使你避免陷入HTML学习中。虽然即便最好的工具也无法生成与手动编写一样的干净、专业的HTML，但是如果你清楚要做什么，使用它们可以加快进度。

下面是一些流行的Web创作程序：

Adobe Dreamweaver。由于这个软件的简洁和高级性能，它已经成为行业标准。

Microsoft Expression Web（只限Windows）。这是微软专业设计工具套装软件的一部分，它的强项是标准兼容代码和基于CSS的布局。

Nvu（Linux、Windows和Mac OS X）。不想购买一个所见即所得的编辑器么？那么你可以选择Nvu（叫做N-view，表示“new view”），这是一个开源工具，它实现了Dreamweaver中的很多功能，可以免费从nvu.com下载。

注意： 本书的练习只需要使用操作系统自带的文本编辑器即可。并不需要什么特殊的程序。

HTML编辑器

HTML编辑器（不同于所见即所得的创作工具）可以加快手写HTML的速度。它不允许可视化地编辑页面，所以你需要在浏览器中查看你的工作。实际上很多专业的Web设计师更喜欢手写HTML文件，他们强烈推荐以下工具：

TextPad（只限Windows）。TextPad是Windows上的一种简单又便宜的文本编辑器。

Sublime Text（Windows, Mac, Linux）。这个软件不贵，看起来像毫无装饰的文本编辑器一样，但是由于有很多功能（如色彩编码和全码查看）受到开发者的喜爱而日益变得重要。

Panic软件制作的Coda（只限Macintosh）。代码工作者喜欢它的工作

流、文件管理工具和内置的终端访问工具。

TextMate by MacroMates（只限Macintosh）。这个高级的文本编辑器具有项目管理工具和一个接口，这个接口是与Mac操作系统集成的。由于它很容易使用、功能丰富且价格低廉，所以越来越受欢迎。

Bare Bones 软件制作的BBEdit（只限Macintosh）。大量的快捷方式使其成为基于Mac的Web开发者的主要编辑器。

图像编辑和绘制软件

你很可能需要在你的页面上添加图片，这样就需要一个图片编辑程序。将在第四部分详细了解一些备受欢迎的程序。现在，你可能想要看一些很受欢迎的网页图片创建工具：

Adobe Photoshop。毋庸置疑，在印刷和网站方面，Photoshop都是图片创建的行业标准。

Adobe Photoshop Elements。这个轻量级的Photoshop是用来进行照片编辑和管理的，但是一些爱好者会发现，它具有在网页上发布图片时所需的所有工具。

Adobe Illustrator。由于设计师需要创建各种大小和分辨率的商标、图标和矢量图，很多设计师以在Illustrator中的矢量图开始寻求最大的灵活度。你可以使用Illustrator直接输出Web图像，或者在Photoshop中稍微调整。

Adobe Fireworks。这个Web画图程序由一个图片编辑器和创建插图的矢量工具集成。此外，在输出Web图形上，它也是先进的工具。

Corel Paint Shop Pro Photo（只限Windows）。这是一个全功能的图片编辑器，对于使用Windows的用户来说，这个软件非常受欢迎，首要原因是它的价格低廉。

GIMP，“GNU图像控制程序”（Unix、Windows、Mac）。这个免费的图像编辑器类似于Photoshop。

互联网工具

由于你将要与互联网打交道，所以你需要一些专用的工具，以便在互联网上查看和移动文件：

各种浏览器。由于各种浏览器表现页面的方式不同，所以你需要在尽可能多的浏览器上测试你的页面，无论是台式机还是移动设备。市场上有数百种浏览器，下面这些是Windows和Macintosh操作系统上最常应用的最好的浏览器：

Windows:	Macintosh OS X:
Internet Explorer (当前版本和至少前两个版本)	Safari
Chrome	Chrome
Firefox	Firefox
Safari	Opera
Opera	

而且你不该忽视手机浏览器！下面的列表是写本书时最常见的一些手机浏览器（没人知道当你读到本书的时候，什么样的手机浏览器会变得重要）：

- 移动版Safari（iOS）
- 安卓浏览器（Android）
- 黑莓浏览器（RIM）
- 诺基亚40系列和塞班系统的诺基亚浏览器
- 移动和迷你设备的Opera（可以安装在任何设备上）
- 移动Internet Explorer（Windows Phone）
- Silk（Kindle Fire）

文件传输程序（FTP）。FTP程序使你可以在电脑和Web服务器之间上传和下载文件。前面列出的Web创作工具都有内置的FTP程序。当然一些专用的FTP程序，如下：

Windows:	Macintosh OS X:
WS_FTP	Transmit
CuteFTP	Cyberduck
AceFTP	Fetch
Filezilla	

终端应用程序。如果你使用过Unix操作系统，你会发现终端（命令行）应用程序是非常有用的，它可以使你在服务器上输入Unix命令。这对设置文件权限、移动或复制文件和目录以及管理服务端软件是非常有用的。

Windows用户可以安装一个Linux模拟器，叫做Cygwin，以便进行命令行访问。还有一个程序是PuTTY，这是一个免费的Telnet/SSH客户端。Mac OS X包含一个名叫Terminal的程序，这是一个完全成熟的终端应用程序，它可以让你访问底层的Unix系统，并且可以使用SSH来访问互联网上的其他命令行系统。

流行的Web设计软件

网页创作

Adobe (Macromedia) Dreamweaver www.adobe.comMicrosoft Expression Web www.microsoft.com/products/expressionNvu (开源网页编辑器) www.nvu.com

HTML编辑

MacroMates制作的TextMate by (适用于Mac OS) www.macromates.comSublime Text www.sublimetext.comTextPad (适用于Windows) www.textpad.comPanic Software制作的Coda www.panic.com/coda/Bare Bones Software制作的BBEdit www.barebones.com

图像编辑和绘制

Adobe Photoshop www.adobe.comAdobe Photoshop Elements www.adobe.comAdobe Illustrator www.adobe.comAdobe Fireworks www.adobe.comCorel Paint Shop Pro Photo www.corel.comGIMP gimp.org

浏览器

Microsoft Internet Explorer (只限Windows) www.microsoft.com/windows/internet-explorer/Firefox www.firefox.comGoogle Chrome www.google.com/chromeOpera www.opera.comSafari www.apple.com/safari

网络

适用于Windows WS_FTP、CuteFTP、AceFTP以及其他软件: www.download.comTransmit (适用于Macintosh OS X) www.panic.comCyberduck (支持Macintosh OS X) cyberduck.chFetch (支持Macintosh OS X) fetchsoftworks.comCygwin (适用于Windows的Linux 模拟器) www.cygwin.comPuTTY (telnet/SSH terminal模拟器) www.chiark.greenend.org.uk/~sgtatham/putty/

你学会了什么

在本章中，你学会的是“没有必要学习一切知识”。即便你最终想了解一切，你也没必要一次学完。所以放松，不要着急。另一个好消息是，虽然有很多专业工具，但是你可以使用你已有的电脑设置和免费的或者便宜的工具来创建一个简单的网站，并且发布这个网站，不是非要花很多钱才能完成这个目标。

你将会看到，制作一个网页非常简单——在阅读本书的时候，你就可以创建简单的页面。然后，你可以不断地把你新学到的技术添加到你的Web设计中。

自我测验

每章的最后都有一系列的问题，你可以通过回答这些问题来检查是否已经掌握了重要的信息。答案在附录A中。

- 把下面这些Web专业人士和他们负责制作的产品匹配起来。

A. 图像设计师	_____ HTML 和 CSS文件
B. 产品部署者	_____ PHP脚本
C. 用户体验设计师	_____ Photoshop草图
D. Web程序员	_____ 故事板
- W3C负责什么工作?
- 把下面的Web技术和相应的工作匹配起来。

A. HTML	_____ 检查表单输入，以确保输入有效
B. CSS	_____ 创建一个自定义的服务端Web程序
C. JavaScript	_____ 把文本标记为二级标题
D. PHP	_____ 定义一个新的标记语言，以便分享财经信息
E. XML	_____ 使所有二级标题成为蓝色
- 前台和后台Web开发有什么不同?
- Web创作程序和HTML编辑工具有什么不同?

练习1-1 评估学习效果

你已经开始了Web设计的第一步，现在总结一下你了解了多少，以及你的目标。将本章的列表作为向导，尽量回答下面的问题：

- 你的Web设计目标是什么？成为一个专业的Web设计师？还是只需要做一个专业的网站？
- 你对Web设计哪一方面最感兴趣？
- 你已掌握的哪些技能对网页设计是有用的？
- 你需要继续学习哪些技能？
- 要进行Web设计，你有哪些硬件和软件工具？
- 你需要购买哪些工具？你最想购买哪些工具？

第2章 Web是如何工作的

我早在1993年就开始了Web设计——那时候Web也刚刚起步。在Web时代，我算是个老人了，我甚至已经记不清楚第一次看到网页是什么时候了。很难说明信息是从哪里来的，以及这些信息是如何工作的。

本章讲述的就是这些问题，并且会介绍一些你可能遇到的基本术语。首先我们会概述，然后具体讲述每个细节。

互联网与Web

请不要误会，在这里并非要详细讨论这两个概念，只是借此机会指出二者的区别，因为这两个概念现在越来越多地被交换使用。

互联网是一个电脑连接的网络。互联网不属于任何一家公司，它是在规范的标准和规则管理下协同努力的结果。把电脑连接起来的目的是分享信息。在电脑之间传递信息有很多种方式，包括 Email、文件传输（FTP）和其他基于互联网构建的专有模型等。这些在网络中传递数据和文件的标准化的方式称为协议（protocol）。

Web（通常称为万维网，也就是站点地址中的“WWW”）只是互联网上分享信息的方式之一。它的特点是，它允许使用超文本链接把一个文件链接到另一个文件——以这样的方式形成一个巨大的信息连接的“Web”。Web使用HTTP（超文本传输协议）协议。你应该对缩写HTTP非常熟悉，因为它几乎是所有的站点地址的前四个字母，在下面的章节中将会对它进行讨论。

提供你的信息

我们先详细地讲述组成互联网的电脑。因为它们可以根据请求来“提供”

本章内容

对Web进行解释，因为它
与互联网相关
服务器的作用
浏览器的作用
介绍URL及其组成
分析一个网页

Web是互联网的子集，它只是信息在网络计算机中传播的方式之一。

Web简史

1989年Web诞生于瑞士日内瓦的一个量子物理实验室（CERN）。在那里，一位名叫Tim Berners-Lee的计算机专家首先提出了信息管理的系统，这套系统使用“超文本”来链接网络上相关的文件。他和他的合作伙伴Robert Cailliau，创建了一个原型，并且把这个原型发布出来以便检测。在最初的几年，网页只是文本。令人难以置信的是，直到1992年，这个世界总共仅有50台Web服务器。

Web的真正爆发是在1992年，当第一个图形化浏览器（NCSA Mosaic）出现后，Web从科研机构普及到普通大众。Web的发展是由万维网联盟（W3C）来监督的。

如果你想深入了解Web的历史，可以查看下面的站点：

万维网联盟的历史文档：

www.w3.org/History.html

文件，所以这些电脑被称为服务器。更准确地说，服务器是一些可以让电脑彼此通信的软件（而不是电脑本身）；然而，通常用“服务器”这个词来代替电脑。服务器软件的作用是等待请求信息，然后尽快取出并发送这些信息。

电脑本身并没有什么特殊之处……高性能的Unix机器和普通的个人电脑都一样。真正使电脑成为服务器的是服务器软件。要想使一台电脑成为Web的一部分，就必须在电脑上运行特殊的Web服务器软件来处理超文本传输协议。Web服务器也称为“HTTP服务器”。

有很多服务器软件可供选择，但是最流行的是Apache（开源软件）和微软公司的互联网信息服务（Internet Information Services, IIS）。Apache在基于Unix的电脑上免费可用的，而且可以安装在苹果机的Mac OS X操作系统上，当然也有一个Windows版本。微软公司的IIS是微软家庭服务器解决方案的一部分。

每台连接在互联网上的电脑和终端（调制解调器、路由器、智能手机和电缆继电器等）都被分配到唯一的一个数字IP地址（IP是Internet Protocol的缩写）。例如，oreilly.com主机的IP地址是208.201.239.100。这些数字很容易使人眼花缭乱，幸运的是，域名系统（DNS）可以让域名指向服务器，oreilly.com就是一个域名。数字IP地址对计算机软件是非常有用的，同时域名对人类来说更容易接受。把文字式的域名和数字IP地址分别对应起来就是独立的DNS服务器的工作。

当然，你完全可以配置你的Web服务器，使很多域名与单独的IP地址映射起来，从而让多个站点共享一台服务器。

术语

开源

开源（open source）软件是以协作的方式来开发的，并且可以让其他程序员看到源代码，以供他们使用和修改。开源程序通常是免费可用的。

没有足够的IP地址了

IANA是分配IP地址的组织，在2011年2月3日拿出了最后一批IP地址，也就是说，再也没有像###.###.###.###格式的IP地址了。这种格式的IP地址（称为IPv4）可以提供43亿单独的地址。在1977年互联网“试验”的时候，这些地址看起来是足够的。当时的创造者们无法预料有一天每个手机、电视、超市货架上的每个物品都需要一个IP地址。

解决的办法是使用新的IP格式（IPv6，已经在使用中），可以提供数万亿的IP地址，但是IPv6与当前基于IPv4的网络是不兼容的，所以IPv6使用在与当前的网络平行的网络上。IPv4终会平稳过渡到IPv6，但是这一天的到来可能需要数十年。

关于浏览器

现在我们知道服务器完成服务的工作，但是另一半的工作由谁来完成呢？发送请求的软件叫做客户端（client）。人们用桌面浏览器、手机浏览器和其他自助技术（如屏幕阅读器）作为客户端来获取Web上的文件和信息。服务器会把相应的文件返回给浏览器（在技术领域通常是指用户代理）来显示。

请求和响应是通过我们之前提到过的HTTP协议来处理的。尽管我们谈过HTTP可以处理的“文件”，它还可以用来传输图片、电影、音频文件、数据、脚本和其他通常构成网站或应用的Web资源。

当我们想到浏览器时，通常的印象总是一台计算机显示器中的一个窗口，其中显示一个网页。这在相当长一段时间都是图形化浏览器或桌面浏览器，它们是当时仅有的Web查看策略。最受欢迎的这种形式的桌面浏览器有Windows上的Internet Explorer、Chrome、Firefox、Safari和Opera。而且现在越来越多的人开始使用手机或平板电脑上的浏览器来上网。然而，还有一点需要记住的是用户体验。视觉障碍的用户可能使用一个屏幕阅读器来听网页的内容；行动不便的用户可以使用辅助设备来访问链接和输入。我们创建的网站必须适用于所有这些用户。

谨记，即便在之前在Web的发展中介绍过的桌面浏览器上，你的网页的外观和工作方式也可能随着不同浏览器而变化。这是因为对Web技术的支持不同，以及用户设置他们自己的浏览器属性的能力不同。

术语

服务端（server-side）和客户端（client-side）

在Web设计中，你常常会听到“客户端”或“服务端”应用程序。这些术语只是为了表示哪台计算机在处理事务。客户端应用程序运行在用户的电脑上，而服务端应用程序和功能利用的是服务器的性能。

内部网和外部网

通常认为，一个网站可以被Web上的任何一个用户访问。然而，很多公司利用网站的信息分享和聚集能力在公司内部交流信息。这种特殊的网络称为内部网（intranet）。这些网站的创建方式和功能与普通网站相似，只是这些网站被放在配有特殊安全设备（叫做防火墙）的电脑上，这种安全设备可以阻止外部用户访问。内部网有很多作用，例如，分享人力资源信息，或者提供访问数据库的权限。

外部网（extranet）与内部网相似，不同之处是，外部网允许公司以外的用户访问。例如，一家制造公司可以为其用户提供一个密码，使他们能够在公司的订单数据库中查询他们自己的订单状态。当然，密码确定了用户可以查看公司的哪些信息。

嗨，那个URL怎么没有 http://

由于所有网页都使用超文本传输协议，所以http://部分常常被省略。在打印机和电视上，网站的名字往往省略http://，这样URL看起来会简短一些，更好记。

另外，浏览器可以自动添加http://，省得用户输入。看起来你可能遗漏了http://，但实际上浏览器已经自动添加并发送到服务器上。

在第6章中，我们使用URL在HTML文件中创建超链接时，当把一个链接指向另一台服务器上的网页时，必须包含所用协议。

注意：有时你会看到一些URL以https://开头。这表示这是一个安全服务事务。安全服务有特殊的加密设备，可以在传给浏览器，或者从浏览器传出的时候，隐藏机密内容。下次网上购物时可以寻找一下。

网页地址 (URL)

Web上的每个页面和资源都有特殊的地址，那就是URL（统一资源定位器）。在现在的公交车侧面、公司名片或者电视的商业化广播上，你每天都可能看到URL（读作“U-R-L”，而不是“erl”）。网站地址深入了我们生活的方方面面。

一些URL很简短，还有一些URL是用点和斜线隔开的混乱的字符串，虽然看起来很繁杂，但是其中每一部分都有特定的含义。我们将详细讲述。

URL的组成

完整的URL通常由三部分组成：协议、站点名字和文档或资源的绝对路径，如图2-1所示。



图2-1: URL的组成

① http://

URL所做的第一件事是指定处理事务所用的协议。字母 HTTP告诉服务器使用超文本传输协议，或者进入“Web模式”。

② www.example.com

URL的第二部分用域名标识了网站。在这个例子中，域名是example.com。开头的www.是特殊的主机名字。主机“www”已经成为一个惯例，但并不是规定。事实上，有时候主机名可以省略。每个域名都可以有多个网站（也就是子域名）。例如，可能有development.example.com和clients.example.com等。

③ /2012/samples/first.html

这是在服务器上被请求的HTML文件first.html的绝对路径。用斜线分隔的单词表示目录名称，从主机的根目录开始。由于互联网最初由运行Unix操作系统的计算机组成，所以到现在依然遵循很多Unix的规则和约定，因此使用/分离目录名称。

总之，图2-1例子中的URL说明它使用的是HTTP协议来连接互联网上名叫www.example.com的Web服务器，并且请求first.html文件（这个文件在Samples文件夹中，再上一层目录是2012）。

默认文件

显然，并非每一个URL都是这么长。很多地址没有包含文件名字，只简单地指向一个目录，就像下面这样：

```
http://oreilly.com
http://www.jendesign.com/resume/
```

当服务器收到对一个目录的请求而不是对特定文件的请求时，它就会在目录中寻找默认文件，这个文件通常称为`index.html`，所以当在浏览器中输入上面这些URL时，实际会看到的是：

```
http://oreilly.com/index.html
http://jendesign.com/resume/index.html
```

默认文件（一般指的是index文件）的名字可以改变，这取决于服务器是如何配置的。在这个例子中，默认文件是`index.html`，但还有一些服务器会使用`default.htm`作为默认文件。如果你的站点是使用服务端程序来生成页面的，index文件可能就会命名为`index.php`或者`index.asp`。与你的服务器管理员或技术支持部门一同检测一下，以确保默认文件的名称正确。

在第一个例子中还有一个要注意的地方，原先的URL并没有末尾的斜线，以表示这是一个目录。当省略这个斜线时，如果服务器发现存在这个名称的目录，它会自动添加一个斜线。

Index文件对安全也是有用的。如果没有找到默认文件，一些服务器（取决于它们的配置）会显示目录的内容。如图2-2所示，由于没有找到默认文件，目录`housepics`中的所有文件都暴露出来了。阻止人们看到你的这些文件的一个方法是，确保每个目录都有一个index文件。你的系统管理员也可能会添加一些保护措施，以免浏览器显示目录内容。

图2-2：一些服务器在没有找到index文件时，会显示目录的内容



如果没有找到默认的文件，根据配置，一些服务器会返回目录下文件的列表。



网页结构

我们对浏览器窗口中的网页外观都很熟悉，但是背后究竟发生了什么呢？

在图2-3的上部，你可以看到显示在图形化浏览器中的一个简单的网页。虽然你可以把它当做一个整体页面，但是实际上这个页面由四个分离的文件组成：一个HTML文件（*index.html*）一个样式表（*kitchen.css*）和两个图像文件（*foods.gif*和*spoon.gif*）。HTML文件控制了整个展示。

HTML文件

当知道我们在Web看到的图像丰富并且互动性强的页面是由简单的纯文本文件生成的时，你一定会感到惊讶。这种文本文件指的是源文件。

看一看*index.html*，它是Jen's Kitchen网页的源文件。你可以看到它包含的只是文本内容，再加一些描述了页面上的每个文本元素的特殊标记（用<>表示）。

在文本文件上加入描述性标签称为“标记”文件。网页使用的是超文本标记语言，简称HTML，HTML是专门为使用超文本链接的文件创建的。HTML定义了很多文本元素，这些元素组成了文件，如标题、段落、强调文本以及链接。还有一些HTML元素用来添加关于文件的信息（如标题），还可以在页面上添加多媒体信息，如图片、视频以及用于表格输入的小部件等。

值得注意的是，实际上使用的HTML有多种。其中应用最为广泛的是HTML 4.01版和XHTML 1.0版。你可能听说过Web是如何随着设计的HTML 5而广泛应用的。我会在第10章给你揭示所有的HTML版本和使它们变得独特的原因。同时我们还要介绍关于HTML的一些基础性的知识。

HTML标记简介

由于你在第二部分会详细学习HTML，所以我们现在不会讲述太多细节，这里我先告诉你HTML是如何工作的，以及浏览器是如何处理HTML的。

通读图2-3中的HTML文件，并且与浏览器中的结果对比。很容易就可以看出源文件中用HTML标记的元素是如何与浏览器窗口中的显示对应的。

练习2-1

查看源代码

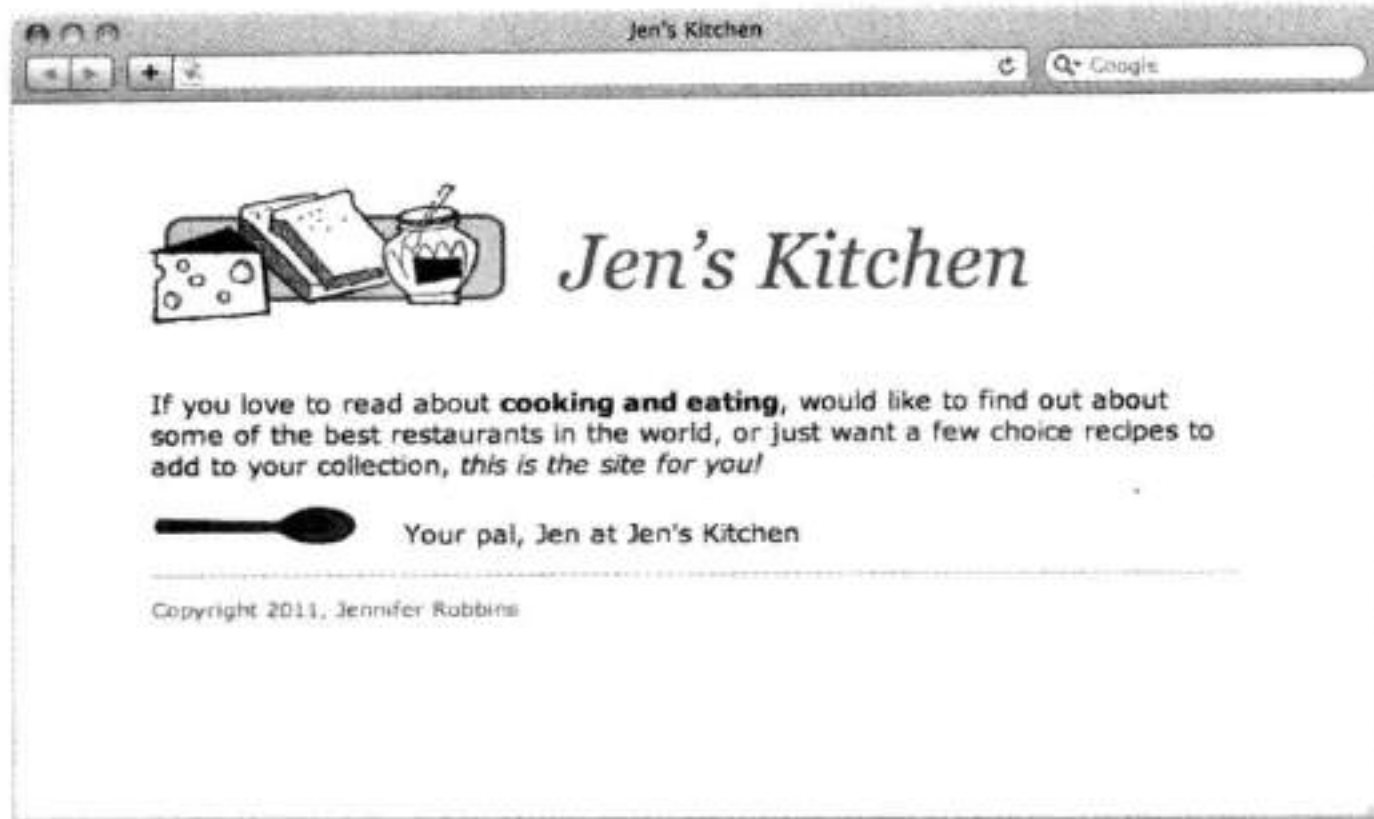
你可以查看任意网页的HTML文件，方法是在浏览器菜单中选择查看一页面源代码（或者查看一源代码）。你的浏览器会在一个单独的窗口上打开源文件。让我们来看看网页的背后到底是什么。

1. 在你的浏览器中输入URL：
www.learningwebdesign.com/4e/materials/chapter02/kitchen.html

你会从图2-3中看到Jen's Kitchen网页。

2. 从浏览器菜单中选择查看一页面源代码（或者查看一源代码）。在Chrome和Opera浏览器中，查看源代码是在开发菜单中。会出现一个新的窗口，显示的就是页面的源文件。
3. 大多数站点的源文件是比较复杂的。请查看*oreilly.com*的源文件，或者任意你选择的站点的源文件。如果你不明白这些源文件的意思，也不用着急。在你读完本书后，很多东西你都会明白的。

警告：请记住，从别人的作品中学习也是很有用的，但是完全照抄别人的代码是愚蠢的（也是非法的）。如果你想使用你所看到的代码，最好还是请求许可，并且为别人的劳动付费。



在这个浏览器窗口中展示的网页实际上是由四个独立的文件组成的：一个HTML文本文件、一个样式表和两个图片。HTML源文档中的标签告诉浏览器如何处理文本，以及如何放置图片。

index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Jen's Kitchen</title>
<link rel="stylesheet" href="kitchen.css" type="text/css" >
</head>

<body>
<h1> Jen's Kitchen</h1>

<p>If you love to read about <strong>cooking and eating</strong>, would like to find out about
of some of the best restaurants in the world, or just want a few choice recipes to add to your
collection, <em>this is the site for you!</em></p>

<p> Your pal, Jen at Jen's Kitchen</p>
<hr>
<p><small>Copyright 2011, Jennifer Robbins</small></p>
</body>
</html>
```

kitchen.css

```
body { font: normal 1em Verdana; margin: 1em 10%;}
h1 { font: italic 3em Georgia; color: rgb(23, 109, 109); margin: 1em 0 1em;}
img { margin: 0 20px 0 0; }
h1 img { margin-bottom: -20px; }
small { color: #666666; }
```

foods.gif



spoon.gif



图2-3：源文件和组成这个简单页面的图片

首先，你会注意到尖括号中的文本（如<body>）并没有显示在最终页面上。浏览器只显示标记之间的元素的内容，而标记反而隐藏了。标记提供了HTML元素的名字——通常是一个缩写，如“h1”表示“一级标题（heading level 1）”、“em”表示“强调文本（emphasized text）”。

其次，你会看到大多数HTML标记是成对出现的，它们之间就是元素的内容。在HTML文件中，<h1>表示接下来的文本是一级标题；</h1>表示标题结束。有些元素称为空元素，它没有任何内容。在这个例子中，<hr>标记表示一个空元素，它告诉浏览器“在这里画一条水平线”（大多数浏览器显示主题划分时使用一条水平线，这也是hr命名的缘由）。

因为我不熟悉计算机编程，当我第一次开始写HTML时，我把标签和文本当做“一串字符”，而由浏览器按顺序来逐字对它们进行解释。例如，当浏览器遇到一个左括号（<）时，它假定接下来所有的字符都是标记的一部分，直到找到右括号（>）。同样，它假定所有<h1>开头的内容都是一个标题，直到遇到</ h1>标记。这就是浏览器解析HTML文档的方式。了解浏览器的解析方式有助于解决不规范的HTML文档中存在的问题。

图片在哪里呢？

显然，HTML文件里并没有图片，那么最终页面中的图片来自哪里呢？

你可以看看图2-3，每个图片都是一个单独的图像文件。图像文件放置在HTML图片元素（img）中，图片元素可以告诉浏览器到哪里去找图片（它的URL）。当浏览器看到img元素时，它就会给服务器发送另一个请求，以获取图片文件，然后把这个图片放在img元素的内容中。浏览器软件把这些零碎的文件整合到最终的页面中。视频和其他嵌入式媒体文件的处理也是如此。

由于页面的整合发生在一瞬间，所以看起来整个页面是一次性载入的。如果网速太慢或者网页包括太大的图片或者多媒体文件，整合的过程就会很明显：文字的显示要快于图片。如果有新的图片到达，页面甚至需要刷新（当然，你可以用某种方式构造你的页面，以阻止发生这样的行为）。

添加一点样式

我希望你注意到我们的最小页面上的最后一个部分。在HTML文档的顶部附近有一个链接元素指向样式表文件`kitchen.css`。该样式表包含了几行指令，说明页面应该如何在浏览器中显示。这些样式指令遵循CSS的规则。CSS允许设计师将视觉样式指令（称为文档外观）添加到标记文本（在网页设计术语里就是指文档的结构）中。在第三部分中，你就会看到CSS的作用。

图2-4显示了Jen's Kitchen页面，一个有样式，另一个没有样式。浏览器为每一个HTML文档都提供了默认样式，所以如果页面没有自己的自定义样式，浏览器就会使用默认样式（也就是右图中所看到的那样）。即便很少的样式规则也可以使页面的外观大为改善。

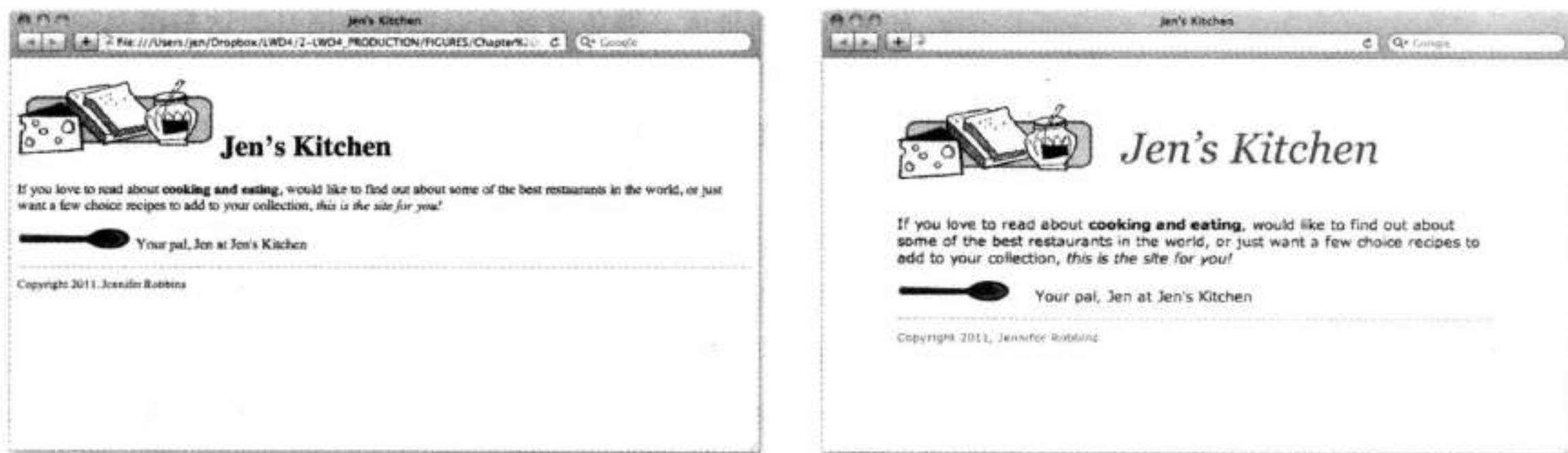


图2-4：Jen's Kitchen页面：使用样式规则之前（左）和使用样式规则之后（右）

使用JavaScript添加行为

除了文档结构和外观，还有一个用来定义如何运行的行为部分。页面使用脚本语言JavaScript来控制行为。我们将在第四部分简单介绍，但是如果想深入学习，则需要了解更多。很多设计师（包括我）都要靠有脚本经验的帮手来给网站添加功能。不管怎样，了解如何编写JavaScript对于网页设计师来说变得越来越重要了。

小结

为了总结Web是如何工作的，让我们追踪出现在屏幕上的每个网页所发生的事件（图2-5）。

- ① 你可以直接在浏览器上输入一个URL（如`http://jenskitchensite.com`），也可以单击页面上的某个链接来获取一个网页。URL包含获取互联网上目标Web服务器上的目标文件所需的一切信息。
- ② 你的浏览器向以URL命名的服务器发送一个HTTP请求，以获取某个特定的文件。如果URL指定的是一个目录（而不是一个文件），就是在请求获取目录中的默认文件。
- ③ 服务器寻找被请求的文件，并发送一个HTTP的反馈。
 - a. 如果找不到页面，服务器就会返回错误消息。这些消息通常是“404 Not Found”，当然还有很多其他的错误消息。
 - b. 如果找到文件，服务器就会取得请求的文件，并且把它返回给浏览器。
- ④ 浏览器解析HTML文件。如果页面包含图片（使用HTML的元素）或者其他外部的资源如脚本，浏览器就会与服务器再次通信，以获取标记指定的图片。
- ⑤ 浏览器在每个元素指定位置插入图片。然后整个网页就整合好了。

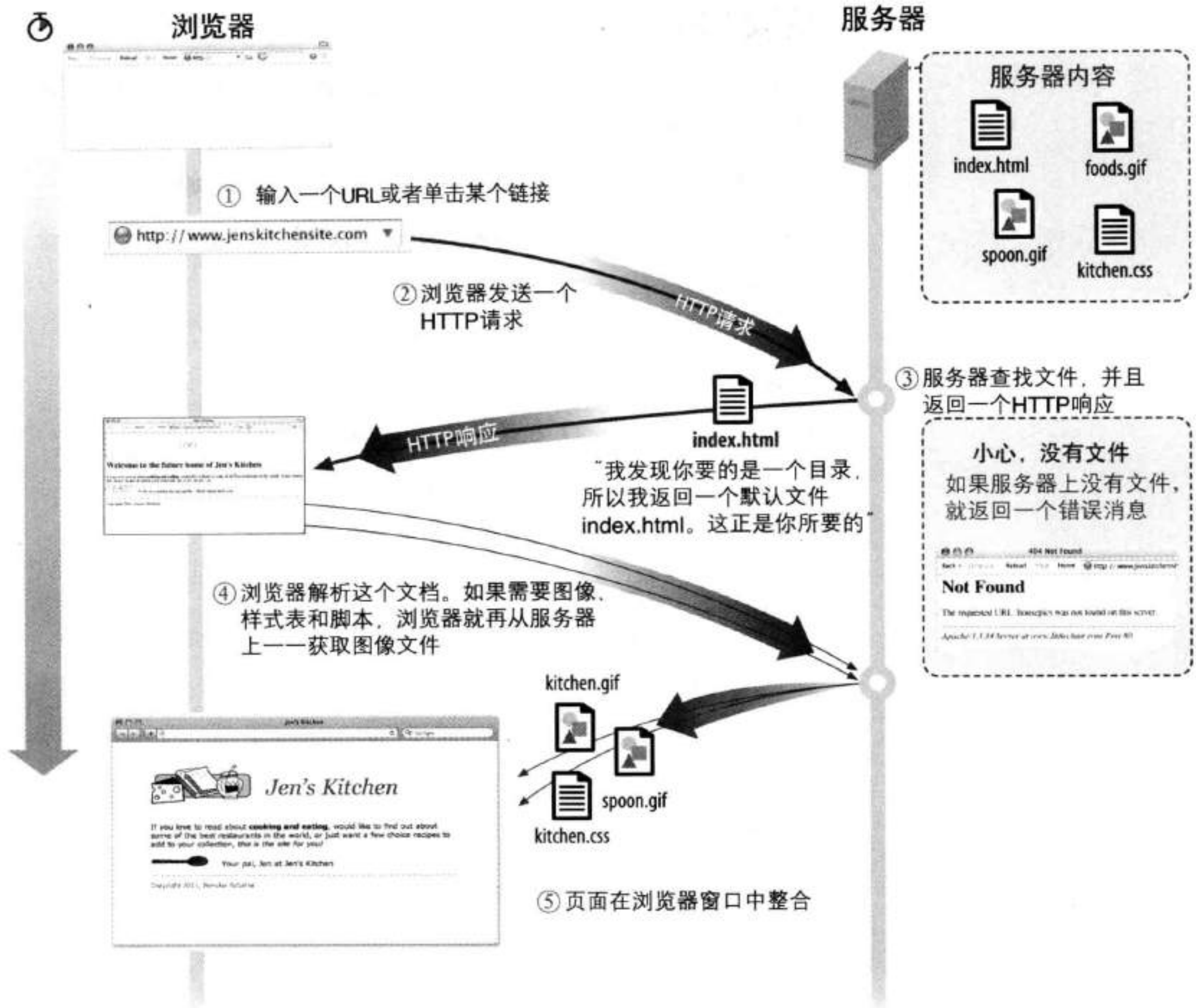


图2-5: 浏览器如何显示网页

自我测验

让我们做一个“识别简称”的游戏，下面是本章提到的一些基本的Web术语。答案在附录A中。

- | | |
|---------|------------------------------|
| 1. HTML | _____ a) Mosaic的主页，第一个图形化浏览器 |
| 2. W3C | _____ b) Web文件或资源的位置 |
| 3. CERN | _____ c) 所有Web文件的标记语言 |
| 4. CSS | _____ d) 匹配域名和数字IP地址 |
| 5. HTTP | _____ e) 传输文件的一种协议 |
| 6. IP | _____ f) 在互联网上传输Web文件的协议 |
| 7. URL | _____ g) 用来指示Web内容如何显示的语言 |
| 8. NCSA | _____ h) Web诞生的量子物理实验室 |
| 9. DNS | _____ i) 互联网协议 |
| 10. FTP | _____ j) 监视Web技术的组织 |

第3章 Web设计基本概念

随着Web的成熟和设备访问数量的成倍增加，作为Web设计师和开发人员，我们的工作变得更加复杂。坦率地说，我无法在一本书中涵盖所有的内容。在本章接下来的部分，我将专注于介绍Web设计的基本部件：HTML元素、CSS样式、小试一下JavaScript和Web图像制作，这会为你的进一步发展奠定坚实的基础。

但在我们实质学习之前，我想介绍每一个Web设计师都需要知道的一些重要概念。我们将看到促进当代Web设计环境优化的一些思想和观念。本书后续的部分中，我会经常提到本章介绍的术语。

Web设计师核心的问题是，我们从来不知道我们创建的页面是如何被查看的。在数以百计的浏览器中，我们不知道用户使用的到底是哪个，用户使用的到底是桌面计算机还是便携设备，更不知道浏览器窗口有多大，安装了哪些字体，JavaScript等功能是否启用，互联网连接的速度如何，用户是否通过屏幕阅读器读取等问题。我想你已经大概有一个框架了。本章中的重要概念主要是为了应对这些无法避免的未知问题。这些概念包括：

- 各种设备
- Web标准
- 逐步提高
- 自适应Web设计
- 可访问性
- 站点性能

在即将开始之前，我将做尽可能简练和非技术的描述。之所以如此，是为了让你在遇到后面的练习时，能基本理解我所说的“逐步提高”的意思。在相关的主题和技术上，已经有很多优秀的文章和书籍，为了方便进一步

本章内容

移动设备上的Web
Web标准的好处
逐步提高
自适应Web设计
可访问性
站点性能

学习，我也会提供指向它们的链接。

眼花缭乱的设备

在2007年前，我们可以确定地说用户访问我们的站点时，往往是坐在桌子旁用着一个速度非常快的浏览器，我们也基本认为960像素对网页来说是一个合适的宽度。而再往前追溯，我们最大的担心是如何兼容多种浏览器、如何支持过时的IE。我们认为那是非常痛苦的！

虽然在2007年前，你也可以通过移动手机来访问网页以获取网页内容，但是直到iPhone、安卓智能手机，覆盖广泛的3G网络的到来，才迎来了一个巨变：何时、何地、怎样上网都发生了变化（尤其是在美国，虽然在移动技术上美国已经落后于亚洲和欧洲）。自那以后，我们就开始看到各种不同尺寸的平板电脑，以及在各种设备甚至电视上的浏览器。而且多样化仍在增加。我想移动教父Brad Frost总结的是非常不错的，如图3-1所示。



图3-1：Brad Frost很好地总结了设备多样性的现状（bradfrostweb.com）

面向所有这些设备的设计挑战不只是解决不同的屏幕尺寸问题。通过宽带连接访问远远不同于通过3G或EDGE网络访问网站，需要考虑很多不同的因素。用户可能坐在办公桌前，在家中悠闲地浏览，也可能在旅途中快速地获取信息。设计师不应该假定网络速度和屏幕尺寸。用户往往可能坐在家里的沙发上，用智能手机连接上WiFi来悠闲地浏览网页，也有可能使用最新的高清晰iPad通过缓慢的3G连接来浏览。总而言之，情景很复杂！

很快，越来越多的人通过移动设备或者其他替代设备而不是桌面电脑来访问Web。当前已经有相当多的美国人将自己的手机作为其唯一接入互联网的途径。这意味着网站设计良好是至关重要的。但说实话，在写本书时，我们还没有完全想通如何使习惯了通过桌面电脑来访问内容的用户在手持设备上依然可以获得愉悦的体验。我们看到我们正在取得长足的进步，而且我们有很好的协作精神，但事实是，我们的工具和技术还不太能满足任务的需求，这将需要一段时间才能赶上。

我希望你能在这里学到的是，你在桌面机上所设计的体验并不是每个人所能体验到的。这其实应该是所有网页设计专业人员应该牢记的。

移动Web

你可能听设计师说过移动Web这个术语，但事实是（就像Stephen Hay 2011年在Twitter上所说的那样，见图3-2）没有什么移动Web、桌面Web或平板Web或者诸如此类的Web。Web就是Web，它可以通过不同设备访问。截至写本书时，“移动Web”是一个统称，它囊括了所有适应于不同使用场景的设计的技术。而且我们发现有多种方式来解释它。

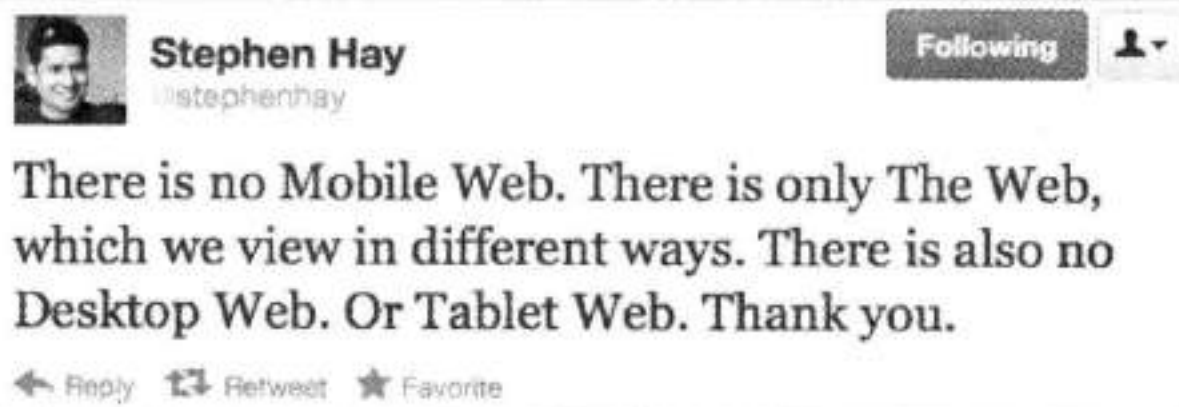


图3-2: Stephen Hay于2011年1月的tweet。阅读他的后续文章可访问www.the-haystack.com/2011/01/07/there-is-no-mobile-web/

进一步阅读

- Scott Jensen在他的文章“*The Coming Zombie Apocalypse*”中思考了廉价网络设备的冲击（designmind.frogdesign.com/blog/the-coming-zombie-apocalypse-small-cheap-devices-will-disrupt-our-old-school-ux-assumptions.htm）。本文非常值得一读。
- Luke Wroblewski所写的Mobile First（书的一部分）。Luke是在变革时首先坚持站点要良好地适应移动设备的人，而且在本书中，他分享了他许多富有创见的观点。
- 未来友好站点（futurefriendly.ly）汇聚了当今许多最具才华的手机设计师。他们认为，环境的变化如此之快，我们不能确保设计在未来也是良好的，但我们可以让它们“对未来友好”。为了这样做，他们收集了一些提示和资源。

与标准一致

那么我们该如何处理这种多样性？首先要做的是遵循由万维网联盟（W3C）提供的HTML、CSS和JavaScript标准。坚持与Web标准一致是确保你的站点在所有符合标准的浏览器显示一致的主要工具（当前使用中的几乎99%的浏览器都符合标准）。它也有助于使你的内容与Web技术和浏览器的发展前向兼容。另一个好处是，你可以告诉用户你的站点符合标准，这样他们会更喜欢你的站点。

坚持与Web标准一致是使你的网站一致的主要工具。

符合标准的概念可能乍看起来毫无理由，但是每个人，包括浏览器厂商，都在快捷和轻松地使用HTML和脚本。我们付出的代价只是需要把站点再创建一次来适应不兼容的浏览器。我在这本书会多次谈到Web标准，所以在这里我不会讲太多的细节。我只想说，Web标准是你的朋友。你在本书学到的一切，将带你走向正确的方向。

进一步阅读

Jeffrey Zeldman的《Designing with Web Standards》是一本关于遵循标准和如何使站点更有商业感觉的“圣经”。去看看吧（当然你得先看完本书）。

逐步提高

随着时间流逝大量浏览器面世，它们都为Web标准提供了不同程度的支持。事实上，没有浏览器能够完全符合标准，同时还有新的技术在不断地引起关注。此外，用户可以设置自己的浏览器偏好，所以他们可能有一个支持JavaScript的浏览器，但选择将其关闭。这里的要点是，我们面对的浏览器提供的功能很多，从基本的HTML支持到其他的一些附加的功能。

面对浏览器的未知功能，逐步提高是一种策略。

面对浏览器的未知功能，逐步提高是一种策略。当使用逐步提高的方法来设计时，你开始时只有一些使内容和功能在基本的浏览器和设备上可用的基本经验。然后，你会利用浏览器能处理的更先进的特性。你可能会使用一些“好”的效果，如动画或圆角框，来提升使用最先进浏览器的用户体验，虽然这些对于品牌和内容来说并不是最重要的。

逐步提高的方法是与页面设计和创建的各个方面都有关的方法，包括HTML、CSS和JavaScript。

创建策略

当按照逻辑顺序编写HTML文档，并且使它的元素用有意义的方式标记时，创建策略对于所有的浏览器环境都是有用的，包括最古老的浏览器、未来的浏览器与移动设备和辅助设备。它可能不会长得一模一样，但最重要的事情是内容是可用的。这也保证了像Google这样的搜索引擎可以正确地采集内容。准确、完整地编写一个HTML文件，是确保站点可访问的重要基础。

样式策略

你只需利用浏览器解析样式表规则的方式来创建体验分层。无须太多的技术细节，你就可以编写一个样式规则使元素背景为红色，只要浏览器支持，也可以用一个样式，使背景色彩渐变。或者也可以

注意：逐步提高是从一个应对浏览器多样性的古老方式（称为优雅降级）而来。在优雅降级的方法中，你首先设计完全的用户体验，然后再对不支持的浏览器做一些改进。

使用尖端的CSS选择器为浏览器提供某些样式。了解浏览器会简单地忽略它无法解释的属性和规则，你就可以大胆地创新，而无须担心浏览器会出现什么问题。你只需要牢记：优先使用基本的样式，然后一点一点地改善。

脚本策略

JavaScript是可以使创造互动网页和动态网页（在使用中更新）的脚本语言。如果没有它，Web就只能是静态页面。与其他Web技术一样，浏览器处理JavaScript的方式也存在差异，而且有的用户倾向于不使用JavaScript。逐步提高的第一规则是即便关闭JavaScript，也要保证基本的功能（如在页面间相互链接，通过表单完成数据的提交等重要功能）可以使用。通过这种方式，你可以确保基本的用户体验，而且在可以使用JavaScript时，进一步增强效果。

进一步阅读

《Adaptive Web Design: Crafting Rich Experiences with Progressive Enhancement》是介绍逐步提高方法最好的书，作者是Aaron Gustafson（Easy Readers出版）。Aaron是本书的技术评审者，但是即便不是，我也会优先推荐他。可以在easy-readers.net/books/adaptive-web-design/看到更多详细信息。

如果你有了更多的开发经验，Todd Parker、Patty Toland、Scott Jehl和Maggie Costello Wachs所著（New Riders出版）的《Designing with Progressive Enhancement》是一本深入技术和最佳实践的优秀作品。可以在filamentgroup.com/dwpe/读到更多信息。

自适应Web设计

默认情况下，大多数的小型设备（如智能手机和平板电脑）上的浏览器可以缩小网页，使其适应屏幕大小，并提供放大页面、拖动浏览的机制。虽然如此，严格来讲，它不是一个良好的体验。在这些设备上，文字太小、阅读不便，链接太小、不好单击，而缩放和左右拖动浏览又增加了操作。

自适应Web设计的策略是根据设备上视窗的大小（浏览器窗口）来提供自定义布局。自适应Web设计的技巧是为所有的设备提供同一个HTML文档，但根据设备不同而应用不同的样式表，从而为不同设备提供最优化的布局。例如，当用智能手机查看页面时，内容只出现一列，并且将链接放大，以便于单击。但是当用桌面浏览器来浏览同样的页面时，内容

自适应Web设计是处理未知屏幕尺寸的策略。

重新排列为多列，并提供传统的导航元素。听起来就像个魔术！（但它实际上只是CSS。）

自从Ethan Marcotte在他的文章“自适应Web设计”中第一次创造了这个词汇后，Web设计社区就一直在热烈地讨论（Ethan Marcotte的这篇文章在2010年的A List Apart上www.alistapart.com/articles/responsive-web-design/。它已经成为应对未知视窗大小的主要工具之一。

图3-3显示了一些例子，表现了自适应网页在典型的桌面显示器、平板电脑、智能手机上的显示。在媒体查询库网站（mediaqueri.es），你可以看到许多更鼓舞人心的例子。你可以尝试打开浏览器，然后将窗口调整到很窄、很宽，可以看看随着窗口大小变化，网页布局的变化。



图3-3：随浏览器窗口大小自适应网站的布局变化

自适应Web设计有助于布局，但它并不能一劳永逸地应对所有移动Web设计的挑战。事实上，为用户和他们所选择的设备提供最好的体验需要

的优化可能不只是调整外观和感觉。通过使用服务器来检测设备及其功能，然后作出决定发送什么可以更好地解决一些问题。通过逐步提高，你可以为大多数的浏览器和设备提供一个基本的体验，对于更好的设备，可以发送增强的选项。

对于一些站点和服务，为移动设备单独建立一个站点（见侧栏“移动专用网站”）是一个更好的选择，单独建立的站点可以有专门的界面和功能，并可以充分利用手机的功能，如地理位置等。这就是说，虽然自适应设计不会解决一切问题，但是如果要为大多数浏览器提供令人满意的体验，它是一个不可或缺的重要组成部分。

移动专用网站

自适应网站的替代方案，是建立一个完全独立的网站、一个单独的网址来为移动设备提供服务。移动网站的网址常用的前缀是m.或mobile.。对于某些类型的网站，如果你知道手机用户和台式电脑用户具有不同的行为模式，建立移动专用网站是最好的解决办法。在移动专用网站上，最常用的功能需要在第一个屏幕上突出显示，而台式电脑网站上的一些额外功能（如促销）就被移除了。（这也会使你思考这些额外功能究竟为台式电脑网站提供了什么样的价值。）

图3-4对比了2012年中期Walgreen公司主网站和移动网站。你可以看到移动网站为手机用户提供了一个更精简的选项集。

移动专用网站为智能手机上的用户将复杂的任务简化。Luke Wroblewski在他的文章中“为什么分别建立移动和桌面网页？”（www.lukew.com/ff/entry.asp?1390）中详细地说明了他服务的Bagcheck选择建立一个单独的网站的原因。我建议你读一读。

这里的要点是，自适应Web设计并不是一个普遍的解决方案。对于功能主要是提供文本内容的网站来说，一点点布局的调整就可能为所有设备带来良好的阅读体验。对于其他网站和Web应用来说，可能就要优先考虑提供完全不同的体验。

移动专用网站的缺点是，它需要两倍以上的工作量。它需要额外的内容规划、模板设计、生产时间和日常维护。但是，如果这样做能给用户提供真正需要的功能，当然是非常值得投资的。

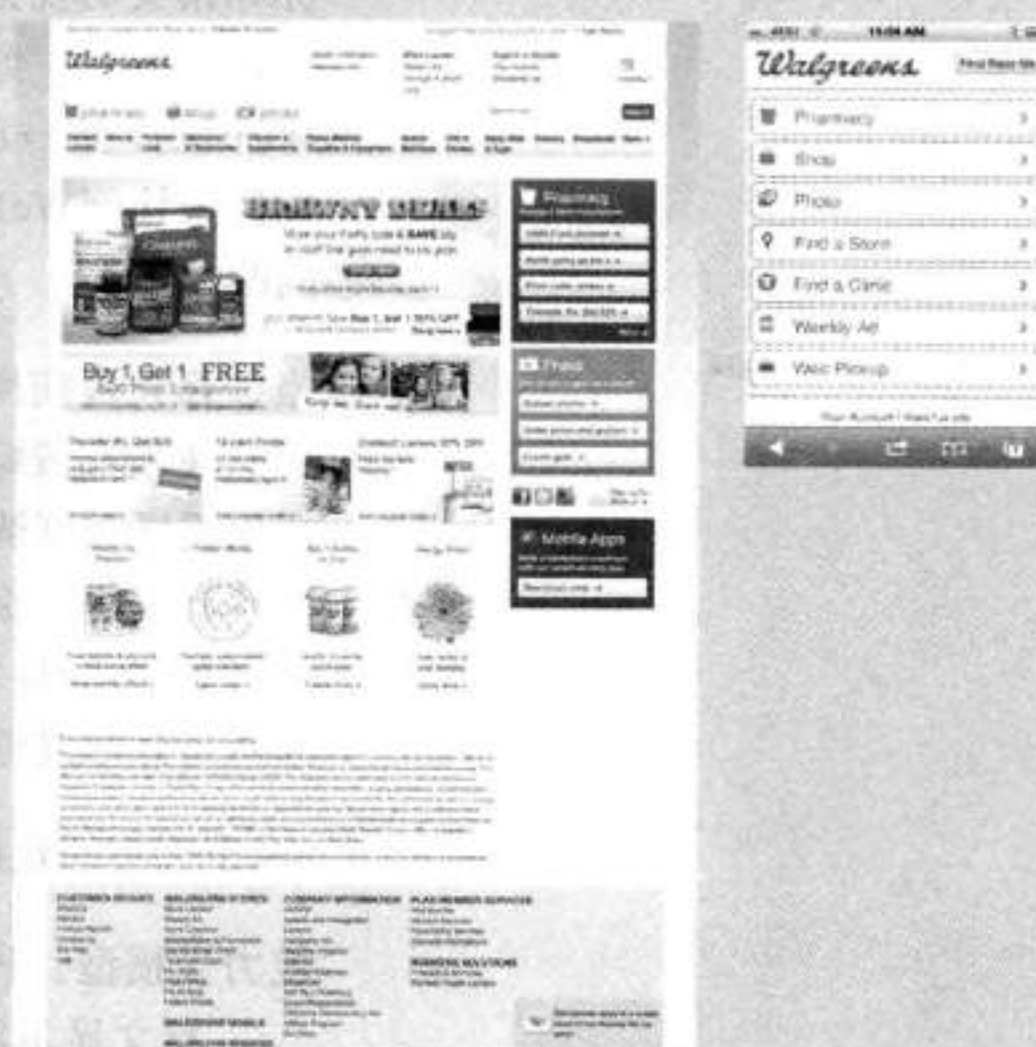


图3-4：移动专用网站与主网站的对比

注意：移动专用网站也能使用自适应技术，来为不同的设备提供良好的体验。移动专用网站和自适应技术并不是要顾此失彼。Stephanie Rieger在她的文章“自适应是一个特性”中很好地总结了这一点，你可以通过stephanierieger.com/responsiveness-is-a-characteristic/来阅读。

进一步阅读

在具有了编码的经验后，在第18章中会详细讨论自适应Web设计。为了进一步对你进行自适应设计教育，我推荐以下书目：

- Ethan Marcott的书《Responsive Web Design》（A Book Apart出版）是初级网页设计的必读书。这是一本小书，是学习自适应Web设计，以及如何自己尝试的完美起点。
- Lyza Danger Gardner和Jason Grigsby所著（O'Reilly Media出版）的《Head First Mobile Web》。这部书详细阐述了自适应Web设计，并讲解了脚本和服务端检测的技术。这本书需要你熟悉一些CSS和JavaScript，而且读来也是非常有趣的。

可访问性——所有用户，一个网站

我们已经讲过当今使用的很多浏览器，但是迄今为止，我们讲的只是鼠标和指尖的视觉控制，但是很重要的是，要记住人们往往用不同的方式来访问Web，如屏幕阅读器、盲文输出、放大器、控制杆、脚踏板等。Web设计师一定要采用某种方式来使信息展示出来，不管用户的能力如何、访问Web的设备如何。换句话说，Web设计师的设计必须是可访问的。

当打算为残疾用户，如视力低下或者移动不便的人设计网页时，会采用一些技术和策略，这些技术和策略对其他用户（如手持设备用户，或者采用缓慢调制解调器连接的，以及禁用图片和JavaScript传统的浏览器的用户）来说，会带来低于最佳效果的浏览体验。可访问的站点被搜索引擎（如Google）收录的效率也更高。多下些功夫，使得你的站点便于访问将是非常值得的。

有4个大类的残疾会影响到人们与自己的电脑和信息互动，它们分别是：

视力受损型。视力低下或失明的人可是使用辅助装置，如屏幕阅读器、盲文显示器或者屏幕放大器来从屏幕上获得内容。他们也可以简单地使用浏览器中的文本变焦功能，以方便阅读文本。

运动不便型。如果用户的手不便移动或者根本无法移动，可以使用特殊的装置，如特制的鼠标和键盘、脚踏板或者操纵杆来浏览网页，并输入信息。

听觉受损型。用户听力微弱或者失聪会错过多媒体的音频信息，因此，有必要提供一些替代的方式，如视频的文字记录或者视频字幕。

认知障碍型。网页设计的简洁明了对于在记忆力、理解力、解决问题的能力上和注意力上有缺陷的用户来说是非常有帮助的，而且对任何浏览你的网站的用户都是有帮助的。

W3C从Web的可访问性开始来探讨如何使Web为每个人所使用。WAI站点 (www.w3.org/WAI) 对学习Web的可访问性来说是一个很好的起点。其中由WAI发布的一个用来帮助开发者创建可访问的站点的文件是网络无障碍指南 (WCAG和WCAG2.0)，你可以通过访问www.w3.org/WAI/intro/wcag.php来阅读。美国政府用WCAG的Priority 1 points来作为可访问指导原则508节的基础 (见侧栏“政府可访问性需求：508节”)。所有的站点都受益于此项指导原则，如果你正在设计一个政府网站，遵循这项原则是必需的要求。

W3C的另一个贡献是WAI-ARIA (可访问的富Internet应用程序) 规约，它提出了Web应用程序 (包括动态内容生成、脚本、适用于辅助设备的界面元素) 的可访问性。ARIA建议书定义了一系列的内容和部件，对这些内容和部件可以使用role属性。role包含菜单、进度条、滑动条、定时器、工具箱等，并在语义上添加了更深层的特性。想进一步了解role，可以访问：www.w3.org/TR/wai-aria/roles#role_definitions。

政府可访问性需求：508节

如果你要为联邦政府创建一个网站，你就必须遵守508节的指导方针，以确保电子信息和技术对于残障人士也是可用的。政府和其他公众机构也需要遵守这些准则。

以下准则是从www.section508.gov的508节中摘出来的，它提供了一个关于基本可访问性的很好的清单。

1. 每一项非文本元素必须提供相应的等效文本内容 (例如，通过使用“alt”、“longdesc”，或在元素内使用)。
2. 等效内容必须在各种表现层上内容一致。
3. 网页设计时，所有通过色彩表达的信息，必须同时以非色彩方法表达，如上下文或置标符号。
4. 文档必须在没有相关样式表时也可正常阅读。
5. 对于行与列的头有两层以上逻辑的数据表格。数据单元格与头单元格应使用置标语言标出。
6. 对于具有两个以上含有行或列标题的逻辑层的数据表单来说，标记可以用于相关的数据单元格和标题单元格。
7. 页面设计中，应避免出现屏幕闪烁频率高于2Hz且低于55Hz的情况。
8. 当页面采用脚本语言显示内容或生成界面元素时，所传递的功能与信息必须可以被辅助技术/工具阅读并识别。
9. 当网页要求应用、插件或其他应用程序在客户端表现页面内容时，该页面必须提供相应链接指向该插件或应用。该插件或应用必须符合 § 1194.21 中(a)到(l)的条款。
10. 当电子表单设计成在线填写时，该表单必须允许用户可使用辅助技术/工具来访问相关信息、字段元素，以及为完成表单的填写与提交而设计的功能，包括各种说明与提示。
11. 必须提供方法，以允许用户跳过重复的导航链接。
12. 当对用户的响应有时间要求时，该用户应该得到事先提醒，并有足够的时间来要求更多的时间宽限。

进一步阅读

下列资源对刚开始学习Web可访问性是很有用的：

- Web可访问性倡议（WAI），www.w3.org/WAI。
- WebAIM：牢记Web可访问性，www.webaim.org。
- Joshue O Connor 所著（Professional Apress，2012年出版）的《Pro HTML 5 Accessibility》。
- Wendy Chisholm 和 Matt May所著（O'Reilly，2008年出版）《Universal Design for Web Applications: Web Applications that Reach Everyone》。

连接速度的要求（站点性能）

注意：看看这篇文章“Effect of Website Speed on Users, Statistics Reveal Slow Loading Times Cost Sites Serious Money”（munchweb.com/effect-of-website-speed），可以学到更多站点性能知识。

虽然使用慢速拨号方式上网的用户不断减少（写作本书时，美国只有5%~10%这样的用户），使用手机访问Web的用户比例明显增加，并将最终超过桌面用户。如果你有一部智能手机，那么你就知道使用蜂窝数据连接访问网页时，漫长的等待是多么令人沮丧。

但是不管用户如何访问你的网站，网站的性能都是至关重要的。Google在2009（注1）年的一项研究表明，搜索时间增加100~400ms时，搜索会减少-0.2%~-0.6%。Amazon.com表明，页面加载时间只需减少100ms，就会带来1%的收入增长（注2）。其他研究表明，用户希望网站在两秒钟内加载，如果网站做不到，有近三分之一的用户就会访问别的网站。而且，这些用户几乎是一去不复返。Google已经优化了其搜索算法的速度，所以如果你的网站很慢，你的网站就不可能出现在Google搜索结果的第一页。因此网站的性能（精确到毫秒）是相当重要的。

有很多办法可以提高网站的性能，这些办法可以分为两大类：限制文件大小或者减少服务器的请求数量。下面的列表仅仅是网站优化的简单技术，但它会给你提供一些总体思路。

- 优化图像，使图像在没有质量损失的条件下最小。你将在第22章中学习这个技术。
- 最小化HTML和CSS文件，去除多余的字符空格和换行回车符。
- JavaScript最小化。

注1：参见“Speed Matters”，googleresearch.blogspot.com/2009/06/speed-matters.html。

注2：统计来源于“Make Data Matter”，由Stanford大学的Greg Linden所做的幻灯片（2006）。

- 加载脚本，使脚本与其他页面资源并行加载，不要影响网页渲染。
- 不要加载不必要的东西（如图片、脚本、JavaScript库）。
- 减少浏览器给服务器发送请求（即HTTP请求）的次数。

服务器每次收到HTTP请求都要耗费几毫秒的时间，而那些毫秒累加起来就不得了。Twitter小工具、Facebook的Like按钮和广告都会产生几十个服务器请求。你可能会惊讶地看到，甚至一个简单的网站都会产生非常多的服务器请求。

如果你想自己看看，你可以使用Chrome浏览器中的开发工具来看每一个发送到服务器的请求，以及这些请求需要的毫秒数。下面告诉你如何使用这个工具：

1. 打开Chrome浏览器，随便打开一个网页。
2. 打开View菜单，选择Developer→Developer Tools。浏览器的下端就会打开一个面板。
3. 选择工具中的Network标签，并重新载入页面。这个图（通常简称为瀑布图）显示了所有的请求和下载的资源。在右边的列显示了每个请求花费的时间（以毫秒为单位）。在底部的图表，你可以看到发出的请求的总数和传输的数据总量。

图3-5显示了我的网站Jenville.com上的一部分性能瀑布图，我的网站只是一个简单的站点（但也不是那么简单）。你可以用这个方法来看任何网站。看了之后，你会收获颇丰。



图3-5：像这样由Chrome网络开发人员创建的瀑布图工具显示页面产生的每个请求，以及每个请求所需耗费的时间

在本书中我不会提及太多的站点性能，但是我确实想让你明白尽可能减小文件大小和省去不必要的服务器请求的重要性。

更多的站点性能工具

可以用下面的一些工具来测试站点性能：

- WebPagetest (webpagetest.org) 最初是为AOL开发的工具，但是现在已经是一个可以自由下载使用的开源软件。只需要输入URL，WebPagetest就会返回一个瀑布图、截屏和其他统计。
- Yahoo的免费工具YSlow工具 (yslow.org) 根据23个Web性能规则来分析站点，它会提供一个分数和改进建议。
- 对于移动站点，可以试试Blaze创建的Mobitest (www.blaze.io/mobile)，它也是免费的，可以测试站点在不同移动设备上的性能。
- 还有一些慢速连接模拟器，使用它们你就可以获得不同网络状况下的用户体验。Sloppy (www.dallaway.com/sloppy) 是一个Web工具，你可以输入一个网址，然后选择一个网速模式（有各种缓慢的速度）。Mac OS用户可以试试Slowy (slowyapp.com)。

进一步阅读

还有一些其他的技术对这本书来说太有技术含量了（坦白说，是对于我），我认为如果你只看这本书，并不能成为一个站点性能高手。但是如果你能看看下面的书，会对你有很大的帮助：

- Google站点上的Make the Web Faster (code.google.com/speed/) 是学习站点优化的第一站。它汇编了很多优秀的教程和文章，以及测量站点速度的工具。
- Steve Souders 所著（O'Reilly Media出版）的《High Performance Web Sites》和《Even Faster Web Sites》，这两本书提供了很多加快站点速度的实践经验。阅读它们需要对JavaScript和服务器功能有良好的理解。

自我测验

1. “移动Web”使我们的工作变得复杂，如Web设计工作。列举出当你设计和开发站点时，至少需要考虑的3个未知因素。
2. 将左侧的技术或练习与它能解决的问题连起来。

1. ____ 逐步提高	a. 辅助阅读和输入设备
2. ____ 服务端检测	b. 缓慢连接速度
3. ____ 自适应设计	c. 浏览器各种功能
4. ____ WAI-ARIA	d. 决定使用何种设备
5. ____ 站点性能优化	e. 不同的屏幕尺寸
3. Web可访问性策略考虑了四类残疾障碍。列举至少三项，并且提出相应的措施来使网站内容可访问。
4. 什么时候你会使用瀑布图？
5. 自适应网页设计并不能解决所有问题。描述它的好处和不足。

HTML结构化标记

第二部分

本部分内容

第4章

创建简单网页
(HTML概述)

第5章

标记文本

第6章

添加链接

第7章

添加图片

第8章

表格标记

第9章

表单

第10章

HTML 5

第4章 创建简单网页（HTML概述）

第一部分概述了Web设计环境。现在已经学习了基础概念，是时候开始创建真正的网页了。它将是一个非常简单的网页，但即使最复杂的网页也基于这里介绍的原则。

本章中，我们会一步一步地创建一个简单网页，从而感觉到如何用HTML标签来标记文档。你可以跟着练习。

本章将讲述下列内容：

- 对HTML标签如何工作有基本的感觉，包括理解元素和特性。
- 观看浏览器如何解释HTML文档。
- 学习HTML文档的基本结构。
- 样式表使用初探。

不要担心现在就学习具体的文本元素或样式表规则，这些将在后面的章节学习。至于现在，只需关注创建过程、文档的整体结构和新术语。

一个网页，一步一步来

你在第2章中看到过HTML文档，现在你将亲自创建一个，并在浏览器中使用。本章的示例分为5个步骤，涵盖了网页制作的基础知识。

第1步：从内容开始。作为起点，我们将添加一些纯文本内容，观察浏览器如何处理。

第2步：文档结构化。你将学习HTML元素，以及创建文档结构的元素。

本章内容

HTML元素和特性介绍
简单网页标签的详细示例
创建文档结构的元素
简单样式表
发现并修正已损网页

HTML苦行之路

我坚持使用老式的教学方法——手写。只有把它一次一个标记地拼写出来，然后在浏览器中打开，才是真正理解标签如何工作的最好方法。很快你就会有正确标记文档的感觉了。

虽然你可以使用一个Web创作工具，但是理解HTML会使工具变得更易用且高效。另外，你会很高兴地看到源文件，并理解你所看到的。对于发现并修正已损网页或者调整Web工具生成的默认格式，这样做也是至关重要的。

专业的Web开发者往往手动添加标记，因为这样他们可以更好地控制代码，使他们能够仔细思考到底使用什么元素，你会发现这么做是值得的！

第3步：确定文本元素。你将用适当的文本元素来描述内容，并学习HTML的正确使用方法。

第4步：添加图像。通过添加图像到网页，你将学习特性和空元素。

第5步：使用样式表改变外观。这个练习让你尝试使用CSS来格式化内容。

到我们完成的时候，你将编写完图4-1所示网页的源文件。它不是很漂亮，但你必须从这里开始。

在示例的整个过程中，将频繁地在浏览器中检查我们的工作——或许比现实中更为频繁——但因为这是HTML的介绍，所以在整个过程中，查看对源文件每一个修改的原因及其影响，将会帮助我们学习。

开始之前，启动文本编辑器

本章乃至整本书中，我们都要手写HTML文档，所以你需要做的第一件事是启动文本编辑器。操作系统会提供文本编辑器，如Notepad（记事本，Windows）或Textedit（Macintosh），它们都是用于文本编辑的。只要能保存为.html扩展名的纯文本（ASCII）文件，使用任何编辑器都可以。如果你有所见即所得的Web编写工具（如Dreamweaver），从现在开始，可以把它先放到一边。我想让你对手写标签文档有所感悟（见侧栏“HTML苦行之路”）。

本节展示了如何在Notepad和Textedit中打开新文档。即使你以前用过这些程序，浏览一下特殊的设置，这些设置会有助于练习。我们将从Notepad开始，Mac用户可以跳过这一段。



图4-1：本章中，我们将一步一步地编写这个网页的源码

在Notepad（Windows用户）中创建新文档

下面是用Windows 7系统中的Notepad创建新文档的步骤（图4-2）：

1. 打开Start（开始）菜单，进入Notepad（在Accessories（附件）程序组）①。
2. 单击Notepad将打开一个新的文档窗口，你可以准备开始打字了。②
3. 接着，要使扩展名可见。这一步不是HTML文档必需的，但它能使文件类型更清晰可见。从“Tools（工具）”菜单③选择“Folder Options...（文件夹选项）”，并选择“View（查看）”选项卡④，找到“Hide extensions for known file types（隐藏已知文件类型的扩展名）”，使该选项不被选中⑤，单击“OK（确定）”保存偏好设置，这样扩展名就可见了。

注意：在Windows 7中，单击Alt键可以显示访问“Tools and Folder Options”的菜单。在Windows Vista中，标签名是“Folder and Search Options”。



图4-2：在Notepad中创建新文档

在TextEdit（Mac OS X用户）中创建新文档

默认情况下，TextEdit创建“多信息文本”，也就是说，文档隐藏了样式格式化（文本变粗、设置字号等）的指示。当TextEdit的窗口上方有一个格式工具栏时，你就可以认为它是处于“多信息文本”模式下（简单的纯文本模式没有这个工具栏）。HTML文档需要变为纯文本文档，所以需要改变格式，如图4-3所示。

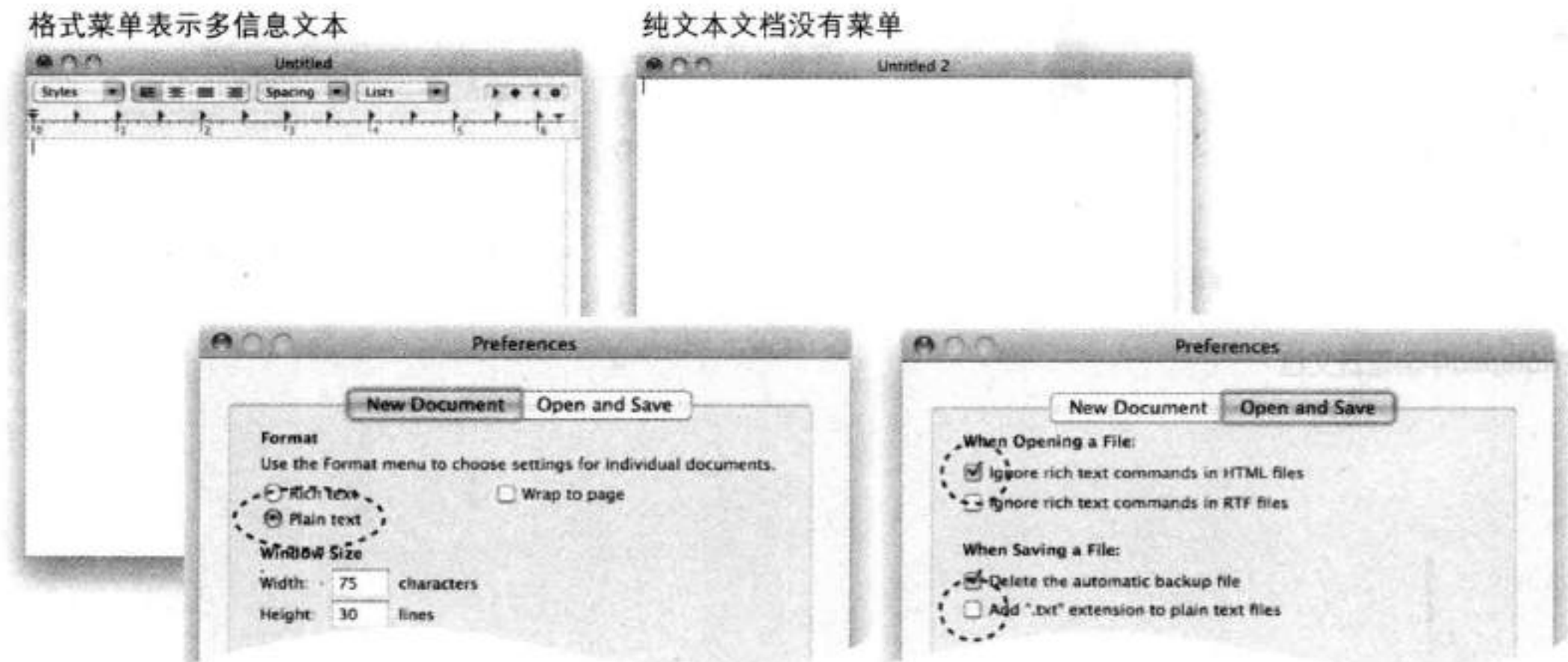
1. 使用Finder在Applications folder中查找TextEdit。找到后，双击程序名称或图标，启动程序。
2. TextEdit打开一个新文档。可以从顶部的文本格式化菜单上看出，你正处于富文本模式1。下面是如何变换模式的步骤。
3. 打开TextEdit菜单的Preferences（偏好设置）对话框。
4. 下面三个设置需要调整：

在“新建文稿”标签下选择“Plain text（纯文本）”。

在“打开和存储”标签下，选择“Ignore rich text commands in HTML files（在HTML文件中忽略富文本指令）”并且关闭“Append ‘.txt’ extensions to plain text files（给纯文本文件添加“.txt”扩展名）”。

5. 完成之后，单击左上角的红按钮。
6. 退出并重新启动，使用新的纯文本设置打开文档。新文档不再出现格式菜单，你就可以把你的文件保存为HTML文档。当然你也可以把一个文档转化成多信息文本：当你不需要编写HTML时，选中“多信息文本”即可。

图4-3：启动TextEdit和选择“纯文本”设置



第1步：从内容开始

既然已经打开新文档，开始编写吧。网页通常从内容开始，因此我们的示范也从它开始。练习4-1中我们将演示输入纯文本内容，并将文档保存在新文件夹中。

练习4-1 输入内容

1. 将下面主页的内容输入文本编辑器的新文件中。照你所看到的原样输入即可。为了后面的演示，请保持换行也与例子一致。这个练习的原始文本可以在www.learningwebdesign.com/4e/materials/中找到

Black Goose Bistro

The restaurant

The Black Goose Bistro offers casual lunch and dinner fare in a hip atmosphere. The menu changes regularly to highlight the freshest ingredients.

Catering

You have fun... we'll handle the cooking. Black Goose Catering can handle events from snacks for bridge club to elegant corporate fundraisers.

Location and Hours

Seekonk, Massachusetts;

Monday through Thursday 11am to 9pm, Friday and Saturday, 11am to midnight

2. 从File（文件）菜单选择“Save（保存）”或“Save as（另存为）”，弹出“Save As”对话框（图4-4）。你要做的第一件事是创建一个新的文件夹，存放这个网站的所有文件（换句话说，它是本地根目录）。

Windows：单击顶部的文件夹图标来创建新文件夹。

Mac：单击“New Folder（新建文件夹）”按钮。

Windows 7



Mac OS X



图4-4：保存index.html到一个名为“bistro”的新文件夹

命名习惯

当命名文件时，遵守下面的规则和习惯是非常重要的：

文件使用正确的后缀。HTML和XHTML文件必须以.html结束。Web图像根据文件格式.gif.png或.jpg（.jpeg也是可接受的）来命名。

不要在文件名内使用字符空格。常用的做法是，用短下划线和短横来在视觉上隔开文件名中的单词，比如robbins_bio.html或robbins-bio.html。

避免特殊字符。如？、%、#、/、:、;、.等。文件名只限使用字母、数字、下划线、连字符和句点。

根据服务器配置，文件名可能区分大小写。没有特殊要求时，在文件名中坚持使用小写字母，使文件名易于管理。

保持文件名简短。短的文件名可以保持字符数和你的HTML文件尺寸在控制之中。如果你真的赋予文件一个冗长、多单词的名称，你可以用下划线隔开，如a-long-document-title.html，以加强可读性。

自定习惯。对大型网站形成统一的命名方案，还是很有用的。例如，总是使用小写字母，并用下划线隔开单词。这样，你后面制作链接的时候，就不用花时间猜测到底是哪个文件名。

浏览器忽略了什么

在浏览器上显示的时候，将忽略源码中的一些信息，包括：

多个空格。当浏览器遇到多个连续空白字符时，它只会显示一个空格。所以如果文档包含：

```
long, long ago
```

那么浏览器将会显示：

```
long, long ago
```

换行（回车）。浏览器把回车转化为空白符，然后根据上一条“忽略多个空白符”的规则进行处理，因此换行对于格式化页面没有效果。在文本流中，文本和元素连续排列，直至遇到块元素，比如标题（h1）或段（p），或者换行（br）元素。

制表符。制表符会被转化为空格，所以你应该明白怎么回事了吧？

不可识别的标签。对于浏览器不理解或者指定错误的标签，浏览器会简单忽略。根据不同的元素和浏览器，将会产生不同的结果。浏览器可能什么都不显示，也可能像普通文本一样显示标签的内容。

注释中的文本。浏览器不会显示特殊的`<!--`和`-->`标签之间的文本，它们用于表示注释。参考本章后面的侧栏“添加隐藏的注释”。

将新文件夹命名为**bistro**，并将文本文件保存为bistro中的**index.html**。对于Windows用户，还要在“Save as type（保存类型）”后面选择“All Files（所有文件）”，以防止Notepad添加一个“.txt”扩展名到文件。文件要以.html结尾以作为一个Web文件被浏览器识别。侧栏“命名习惯”中对此有很多建议。

3. 让我们在浏览器中看一看**index.html**。启动你最喜欢的浏览器（我正在使用Google Chrome），从File（文件）菜单选择“Open（打开）”或“Open File（打开文件）”找到**index.html**，选中文档并在浏览器中打开。你可以看到如图4-5所示的网页的内容。我们将在下节谈论这个结果。

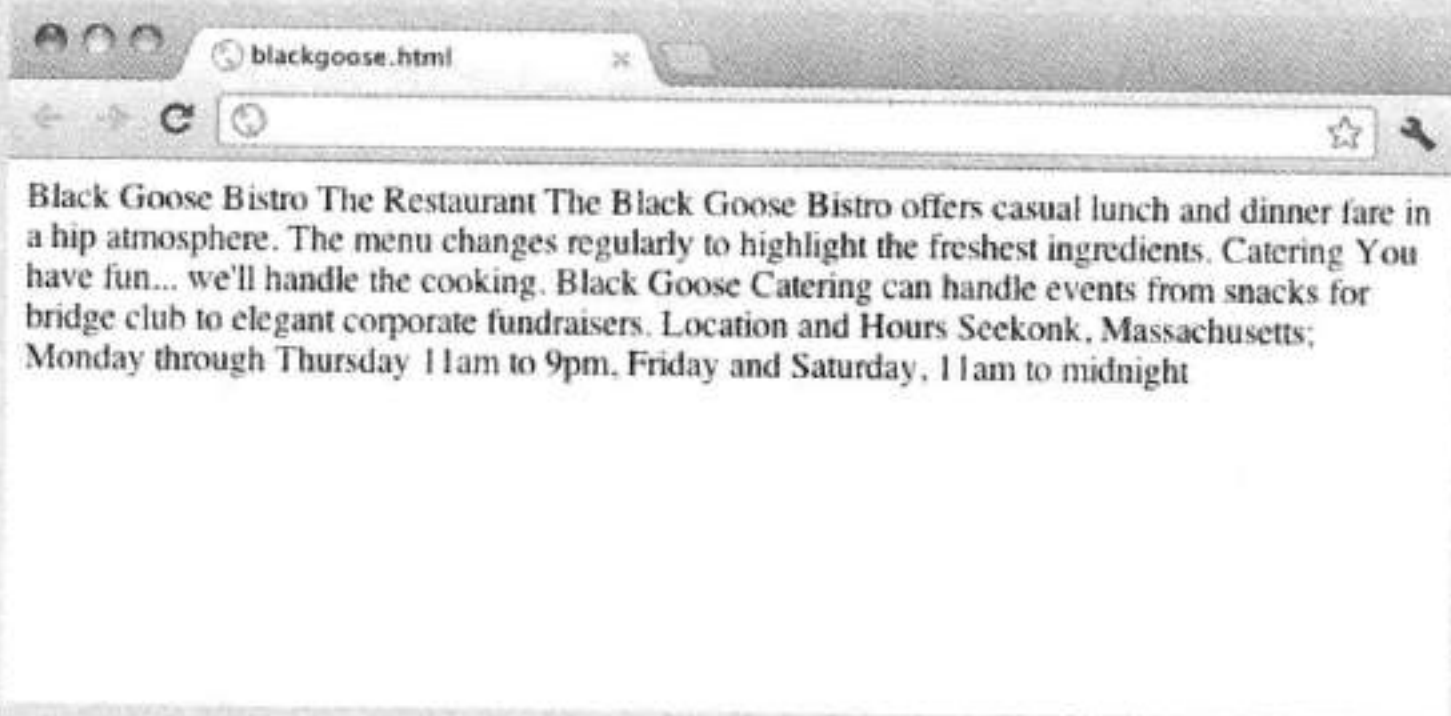


图4-5：在浏览器中预览主页内容

从第1步学到了什么

我们的内容看起来并不那么好（图4-5）。所有的文字挤到一块去了——并不像源文档中看到的。这里还要学习两件事。第一件事显然是，浏览器忽略了源码中的换行。（侧栏“浏览器忽略什么”列出了在浏览器上不显示的源码的其他信息。）

第二件事是，我们看到，简单地输入一些内容并命名文档为.html远远不够。浏览器显示文件中的文本时，我们还不能确定内容的结构。这就是HTML发挥作用的地方了。我们将使用标记来添加结构：首先给HTML文档本身（发生在第2步）添加，然后给网页内容（第3步）添加。一旦浏览器理解了内容的结构，它就可以以更有意义的方式显示。

第2步：文档结构化

我们已经将内容保存在一个.html文档中——现在我们准备开始添加标记。

介绍HTML元素

在第2章中，你看到了带有打开标记（比如段的<p>）和关闭标记（</p>）的HTML元素的例子。在开始向文档添加标记之前，让我们先看一个HTML元素的结构（它的syntax），并巩固一些重要的术语。通用的HTML元素如图4-6所示。



图4-6：HTML元素的组成部分

元素通过文本源码中的标记识别。标记由尖括号（< >）内的元素名（通常是一个冗长的描述性名称的缩写）组成。浏览器知道，尖括号内的任何文本都被隐藏，并且不会显示在浏览器窗口上。

元素名出现在打开标签（也称作开始标记）中，又前置一个斜线（/）出现在关闭（或称作结束）标记。关闭标记工作方式就像元素的“关闭”开关。注意，不要在结束标记中使用类似的反斜线字符（见提示“斜线与反斜线”）。

添加到内容周围的就是标记（markup）。注意到元素是由内容和标记（开始标记和结束标记）组成，这一点很重要。然而，并不是所有元素都有内容。一些被定义为空元素（empty），如用于添加图像到网页的img元素。我们将在本章后面谈论空元素。

最后一件事就是大小写。在HTML中，元素的大小写并不重要。所以，、和在浏览器看来都是可以接受的。然而，在XHTML中（比HTML更为严格的版本），所有的元素都必须小写以确保其有效。许多Web开发者和我一样，都喜欢严格的XHTML标记法则并且坚持用小写标记。

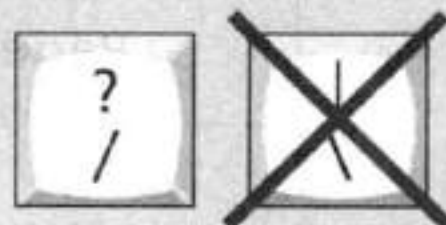
元素由内容和标记组成。

提示

斜线与反斜线

HTML标记和URL都使用斜线字符（/）。标准QWERTY键盘中斜线字符位于问号（?）下方。

很容易将斜线与反斜线（\）搞混。反斜线在竖线字符（|）下方。反斜线不能在标记和URL中工作，所以小心不要使用。



基本文档结构

图4-7展示了一个HTML 5文档的最小轮廓。之所以这么说，是因为需要的HTML元素仅仅是title。我觉得这对于初学者是个很好的样例。如果你要写一个严格的合法的XHTML，必须包含所有的元素（除了meta）。让我们看看图4-7中的内容。

注意：在HTML 5之前，使用meta元素来指定字符集的语法规则比较复杂。如果你使用HTML 4.0.1或者XHTML 1.0标准来编写文档，你的meta元素看起来应该是这个样子：

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

- ① 样例中的第一行并不是一个元素，对此请不要怀疑，我还没有糊涂。它只是一个文档类型声明（document type declaration），也叫做DOCTYPE声明，它用来说明这个文档是一个HTML 5文档。在第10章中我会详细介绍DOCTYPE声明，但是现在你只需知道，写这个声明只是为了让现在的浏览器按照HTML 5的规约来解释这个文档。
- ② 整个文档都包含在一个html元素中。html元素称为根元素，因为它包含文档中的所有元素，并且不能被其他任何元素所包含。html元素既可以用于HTML，也可以用于XHTML文档。
- ③ 在html元素中，文档分成head和body。文档的head元素（有时也称为页头）包含关于文档本身的描述性信息，比如title（标题）、它用的样式表、脚本和其他类型的“meta”信息。
- ④ meta元素在head元素中，提供了这个文档本身的信息。meta元素可以提供各种信息，在这个例子中，它指明了文档的字符编码（也就是字母、数字和字符集）。这里对字符集不作过多解释，你只需要明白指定字符集有很多好处，所以在这个最小的html文档中我也指明了字符集。
- ⑤ head还包含title元素。根据HTML规范，每个文档都必须包含描述性的title。

⑥ 最后，body元素包含所有我们想显示到浏览器窗口中的东西。

你准备好给Black Goose Bistro 主页添加结构内容了吗？打开index.html文档然后开始练习4-2。

图4-7：一个HTML文档的最小结构



练习4-2 添加基本结构

1. 如果还没有打开最新创建的文档`index.html`，那现在打开。
2. 从添加HTML 5 DOCTYPE `<!DOCTYPE html>`开始。
3. 在最开头添加 `<html>`开始标记，在文本末尾添加`</html>`结束标记，从而将整篇文档放入HTML根元素中。
4. 然后，创建文档`head`，它包含网页的标题。在内容之前插入`<head>`和`</head>`标记。在`head`元素内，添加字符编码`<meta charset="utf-8">`和标题“Black Goose Bistro”，标题用开始和结束`<title>`和`</title>`标签包围。

准确地讲，`title`元素是嵌套在`head`元素中的。我们会在后面的章节中讲解嵌套。

5. 最后，将内容装入`<body>`和`</body>`标签之间，从而定义文档的`body`。当你完成之后，源码如下（标记用颜色突出显示）：

```
<!DOCTYPE html>
<html>

<head>
<meta charset = "utf-8" >
<title>Black Goose Bistro</title>
</head>

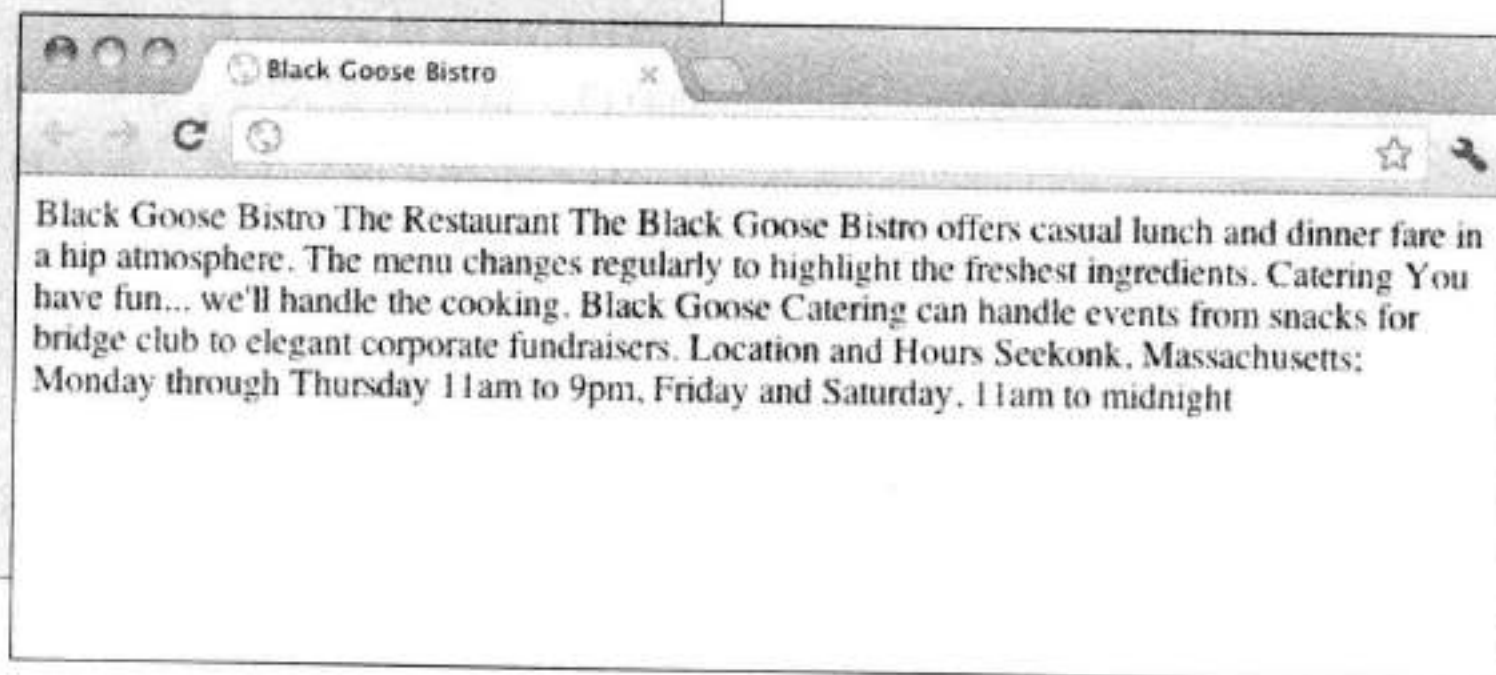
<body>
Black Goose Bistro
The restaurant
The Black Goose Bistro offers casual lunch and dinner fare in
a hip atmosphere. The menu changes regularly to highlight the
freshest ingredients.

Catering Services
You have fun... we'll do the cooking. Black Goose Catering can
handle events from snacks for bridge club to elegant corporate
fundraisers.
Location and Hours
Seekonk, Massachusetts;
Monday through Thursday 11am to 9pm, Friday and Saturday, 11am to
midnight
</body>

</html>
```

6. 将文档保存在**bistro**目录下，这样可以覆盖旧版本。在浏览器中打开文件，如果已经打开，可以单击“refresh（刷新）”或“reload（重新载入）”。图4-8显示了现在的样子。

图4-8：定义了文档结构元素后，浏览器中的主页



别忘了好标题

title元素不仅是每篇文档必需的，也是有用的。标题将显示在用户的书签（Bookmarks）或桌面浏览器的收藏夹（Favorites）列表中。描述性的标题也是加强易用性的一个主要工具。使用屏幕阅读器时，标题将会被人首先听到。搜索引擎也很大程度上依赖文档标题。基于以上原因，为所有文档提供有思想性、描述性的标题，并避免含糊的标题（如“Welcome（欢迎光临）”或“My Page（我的网页）”，是非常重要的。你也可以把标题的长度保持在可控范围之内，这样可以显示在浏览器的标题区中。另一个比较好的习惯是给标题添加更多的信息（比如，公司名字前加上页面描述），这样在浏览器打开多个标签页时，页面标题也还是可见的。

现在除了浏览器在上栏显示文件名称外，并没有其他太大的变化。如果有人想要在这个页面添加书签，那么标题也会添加到她所喜欢的清单中（见侧栏“别忘了好标题”）。但内容依然会保持原样，因为我们并没有给浏览器发出该如何格式化内容的指令。我们接下来会讲到如何设置格式。

第3步：确定文本元素

由于目前你没有什么标签经验，要添加这些标签你可能不知道从哪开始，如标题和副标题（h1和h2）、段落（p）和强调文本（em），就像练习4-3中所做的。然而，因为我们是刚开始，所以我想花点时间讨论一下，在使用HTML标记内容的时候，哪些该做，哪些不该做。

介绍……语义（表意）标记

HTML的目的是给内容增添内涵和结构，但往往不能为如何改善外观（表现层）提供指导。

标记内容时的任务是选择合适的HTML元素，为当前的内容进行最有意义的描述。在术语里，这叫做语义（表意）标记（semantic markup）。例如，网页中的一级标题应该标记为h1，因为它是网页中最重要的标题。不要担心h1在浏览器中的样子——你可以用样式表轻易地改变它。对于选择元素来说，重要的是要以使内容更有意义为原则来选择。

除了增添内容的内涵，标记还使文档结构化。元素一个接一个地排列，或者嵌套在另一个元素中，这些方法创建了元素间的关系。你可以认为这就是一个文档概要（技术上，这叫做DOM，即文档对象模型）。这个文档层次很重要，因为它为浏览器处理文档提供了依据。文档结构也奠定了使用样式表添加表现层指令，以及使用JavaScript添加行为的基础。我们将在第三部分讨论层叠样式表和第四部分JavaScript综述中，更详细地讨论文档结构。

虽然HTML从诞生之日起，就一直严格用于意义和结构，但是在Web的早期阶段，这个使命遇到了阻碍。由于没有合适的样式表系统，作者就用HTML的扩展来改变字体、颜色和对齐的外观。这些表现层扩展至今还保留着，所以你在“view source（查看源代码）”时，有可能会碰到。然而在本书中，为了与现代Web设计中基于标准的新方法保持一致，我们仅关注正确使用HTML。

好了，说得够多了。现在我们开始完成练习4-3中的内容。

练习4-3 定义文本元素

1. 如果还没有打开 `index.html`，那就在文本编辑器中打开。
2. 文本的第一行“Black Goose Bistro”是网页的主标题，所以我们把它标记为一级标题（`h1`）元素。将打开标签 `<h1>` 放在行首，而关闭标签 `</h1>` 放在后面，如下所示：

```
<h1>Black Goose Bistro</h1>
```

3. 网页还有三个副标题。用相似的方法，将它们标记为二级标题（`h2`）元素。我会给你演示第一个，“Catering”和“Location and Hours”这两个你可以用相同的方法完成。

```
<h2>The Restaurant</h2>
```

4. 每个 `h2` 元素后面都跟着一小段文字，所以我们用相同的方法将它们标记为段（`p`）元素。下面是第一个，剩下的你来完成。

```
<p>The Black Goose Bistro offers casual lunch
and dinner fare in a hip atmosphere. The menu
changes regularly to highlight the freshest
ingredients.</p>
```

5. 最后，在“Catering”部分，你可以自己随意添加内容。为了让这段文本突出，将它标记为强调文本（`em`）元素，如下。

```
<p>You have fun... <em>we'll handle the cooking
</em>. Black Goose Catering can handle events
from snacks for bridge club to elegant corporate
fundraisers.</p>
```

6. 既然我们已经标记完了文档，一如继往，保存文档并在浏览器中打开（或刷新）网页。你可以看到几乎与图4-9一样的网页。如果不是，请检查标记，确认没有缺失任何尖括号，没有在关闭标签中缺失斜线。



图4-9：在内容中添加HTML元素标记后的主页

现在我们已经有了基本的了解。元素已经被确定，浏览器现在可以以更好的方式显示文本了。关于图4-9，还有一些重要事项需要注意。

块元素和内联元素

虽然这些是显而易见的，但还是值得指出，标题和段落元素都是从新行开始的，没有像之前一样挤在一起。这是因为在默认下，标题和图表都是块元素（`block element`）的实例。浏览器处理块元素时，把它们当做矩形盒子里，在网页中堆积。每个块元素都是从新行开始，默认情况下，元素整体上下都通常还有一些空间。在图4-10中，块元素的边缘以红色突出显示。

作为对比，看看我们标记为强调（`em`）的文本。它没有从新行开始，仍然在段落的流中。这是因为 `em` 元素是内联元素（`inline element`）。内联元素并不在新行开始，它们只是随流而动。在图4-10中，内联的 `em` 元素用淡蓝色突出显示。

添加隐藏的注释

通过把注意事项标记为注释，你可以为你和其他人在源码文档中留下信息。注释标签（`<!-- -->`）之间的任何内容都不会显示在浏览器上，也不会对源码的其他部分产生任何影响。

```
<!--这是注释 -->
<!--这是一个
多行注释
这里结束-->
```

注释对于标注和组织长篇文档很有用，特别是一个团队的开发者共享时。在本例中，注释用于指出源码中包含导航栏的部分。

```
<!--开始全局导航 -->
<ul>
...
</ul>
<!--结束全局导航 -->
```

记住，虽然浏览器不会在网页上显示注释，但读者可以通过“view source（查看源代码）”看到。所以你要保证你留下的注释要适合每个人。在站点发布之前，删除那些你留给同事的注释是一个好习惯，这样做也可以减小文件大小。

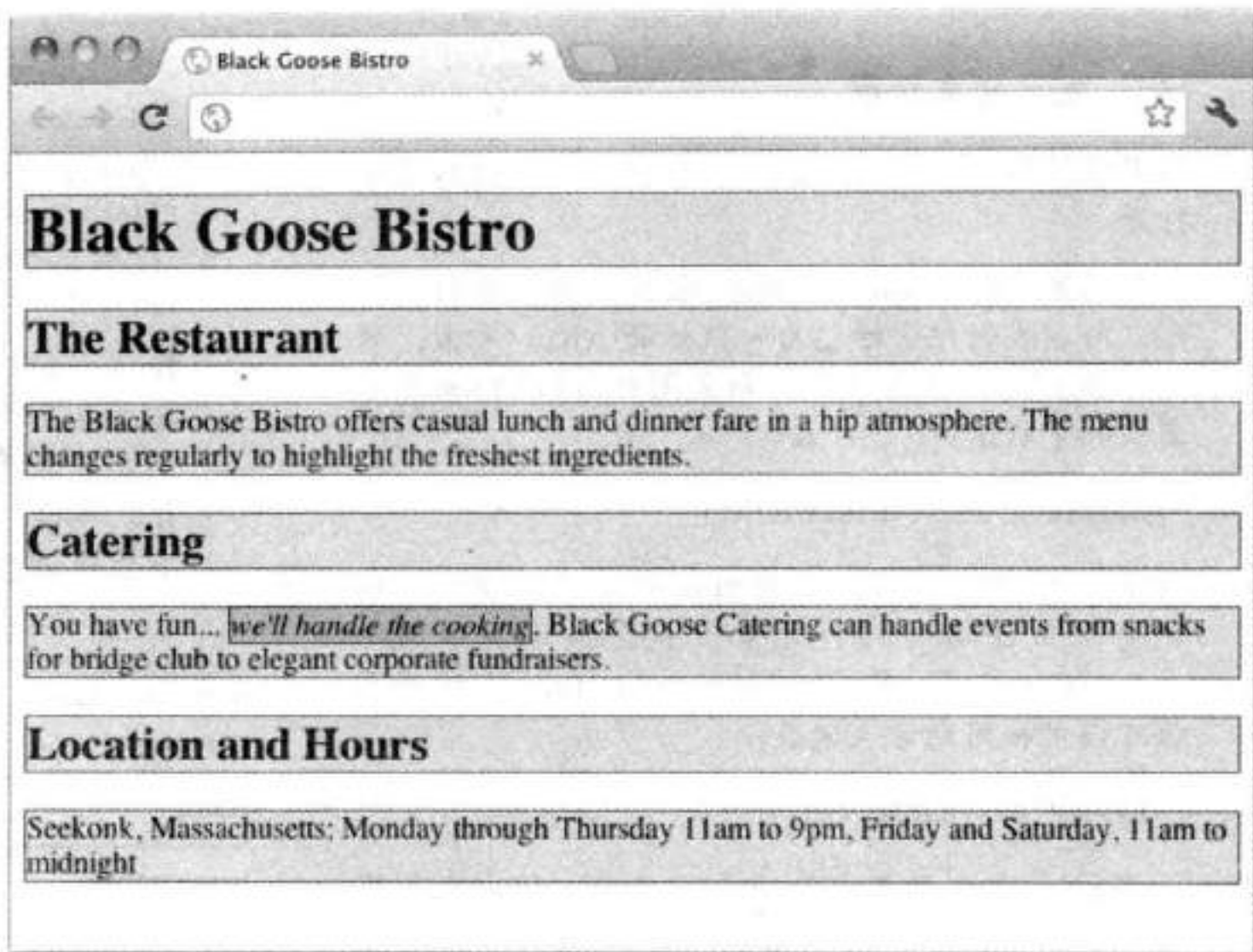


图4-10：突显之处展示了主页中元素的结构

默认样式

关于图4-9和图4-10中标记过的网页，你要注意的另一件事是，浏览器试图制造网页的视觉层次感，使一级标题成为网页中最大最粗的，二级标题稍小，依次类推。

浏览器如何判断h1的外观？它使用样式表来判断！所有的浏览器都有自己内置的样式表（规约称为user agent style sheets），内置样式表可以描述元素的默认显示方式。各个浏览器之间的默认显示总是相似的（比如，h1元素总是大而粗），但是却有些不同（块引用可能缩进，也可能不缩进）。

如果你认为在浏览器中显示的h1太大、太沉闷，可以用样式表规则调整。我反对只是为了使标题更好看，而用其他的元素来标记标题（比如，使用h3代替h1，虽然标题看起来不那么大了）。在样式表没有得到普遍支持的时代，元素只能这样滥用。既然现在已经有样式表可以控制设计，那你就应该总是根据如何精确描述内容来选择元素，而不用担心浏览的默认显示问题。

我们马上就会用样式表调整网页的外观，首先给网页添加一些图像。

第4步：添加图像

没有图像的网页有什么意思？在练习4-4中，我们将使用元素来给网页添加图像。图像将在第7章中讨论。但现在，我们先介绍两个更基本的标记概念：空元素和特性。

空元素

到目前，在Black Goose Bistro主页使用的几乎所有的元素都是遵守图4-1中的语法的：一些文本内容被开始标签和结束标签围着。

然而，还有少数元素没有文本内容，因为它们只是用来提供一些简单的指令。这些元素还是“空的”。图像元素就是这种元素——它告诉浏览器，从服务器获取一幅图片文件并插入文档文本流中的某个位置。其他的空元素包括换行(`br`)元素、水平线(`hr`)元素，以及提供文档信息但不影响显示内容的元素（如我们先前用到过的`meta`元素）。

图4-11显示了一个空元素的简单例子（可以对比图4-4）。如果你正在写的是XHTML文档，语法略微有些不同（参见侧栏“XHTML中的空元素”）。

```
<element-name>
```

例子：`br`元素，插入一行

```
<p>1005 Gravenstein Highway North<br>Sebastopol, CA 95472</p>
```

图4-11：空元素结构

特性

让我们返回到用空元素添加一个图像的例子。很明显，一个标签本身并没什么用——无法知道该用哪幅图像。而这恰是特性发挥作用的地方。特性是用来阐明或修改元素的指示。对于元素，`src`（“source（源）”的简写）特性是必需的，它通过URL提供图像文件的位置。

特性的语法如下：

```
attributename="value"
```

特性跟在元素名的后面，使用一个空格分隔，在一个非空元素中，特性只在开始标签中出现：

XHTML中的空元素

在XHTML中，所有的元素，包括空元素都必须关闭（用准确的术语来讲，叫“终结（terminated）”）。在右尖括号之前，添加前置空格的斜线，可以结束一个空元素，如、`
`、`<meta />`和`<hr />`。下面是同样的例子，但这次使用XHTML语法。

```
<p>1005 Gravenstein Highway  
North <br />Sebastopol, CA  
95472</p>
```



```
<element attributename="value">
<element attributename="value">Content</element>
```

你可以按照任意顺序添加多个特性，只需要用空格把它们分隔开。

```
<element attribute1="value" attribute2="value">
```

图4-12显示了img元素的结构，其中有一些必需的特性。



图4-12：包含特性的img元素

关于特性，下面是我们需要了解的：

- 特性只出现在开始元素中，位于元素名之后，不能在结束元素中。
- 元素中可以应用多个特性，它们在开始标签中用空格隔开。它们的顺序不重要。
- 绝大多数特性都有值，并其后跟着等号 (=)。在HTML中，一些特性的值可以减少到一个表述文字。比如，复选特性 `checked`，它可以在表单载入时让一个复选框选中。然而，在XHTML中，所有的特性都必须有明显的值 (`checked="checked"`)。你可能听到这种特性，它叫做布尔特性，用来表示一个特征是打开还是关闭。
- 特性值可以是数字、单词、字符串、URL，或者其他特殊用途的计量值。这些都取决于特性的目的。在本书中，你将会看到所有的例子。
- 总是把特性值放在引号内。虽然引号并不是HTML中所有元素必需的，但对于XHTML的元素，引号是必需的。许多开发者即便编写HTML都会使用引号，以保持风格一致。只要使用时保持一致，单引号和双引号都是可接受的，不过，习惯上使用双引号。
- 一些特性是必需的，比如，`img`元素中的`src`和`alt`特性。
- 每个元素的可用特性名都定义在HTML规范中；换句话说，你不能为元素自创特性。

现在，你应该做了充足的准备，在下面的练习中试着来尝试将带特性的元素添加到Black Goose Bistro页面。我们在这儿插入一些换行。

练习4-4 添加图像

1. 如果你接着练习一路做下来了，那你需要做的第一件事就是拿一个图像文件的副本放到你的硬盘里，这样，在本地打开文件的时候，你可以在某个位置看到它。图像文件由本章的材料页面中提供。你还可以从www.learningwebdesign.com/4e/chapter4/bistro获取示例图像。在鹅图像上右键单击（在Mac机上是Ctrl-click（译者注：按住ctrl键时单击）），从弹出菜单中选择“Save to disk（保存到硬盘）”（或类似的其他方式），如图4-13所示，并把文件命名为blackgoose.png。一定要把图像保存到index.html所在的bistro文件夹中。
2. 一旦拿到了图像，通过输入img元素及其特性，就可以将图像插入一级标题的开头，如下所示：

```
<h1>Black Goose
Bistro</h1>
```

src特性提供了将要插入的图像的名称，alt特性提供了在图像无法获取时应该显示的文本。这两个特性是每个img元素所必需的。



Windows用户：
右键单击图像来访问弹出菜单，并选择保存图像。

Mac用户：
按住Control键单击图像来访问弹出菜单，然后选择保存图像。不同的浏览器会有不同的弹出菜单内容。

图4-13：从Web上的网页保存图像文件

3. 我想把这个图像显示在标题的上方，可以在元素后添加一个换行(
)，从而使标题文本换行显示。
4. 把最后一段话分成三行，以便一目了然。在你想换行的地方添加
。试着实现图4-14的效果。
5. 现在保存index.html，在浏览器中打开或刷新。网页应该如图4-14所示。否则，检查图像文件blackgoose.png是否与index.html在同一文件夹。如果是，再检查，确保在img元素标记中没有缺失字符，如闭引号或闭括号。

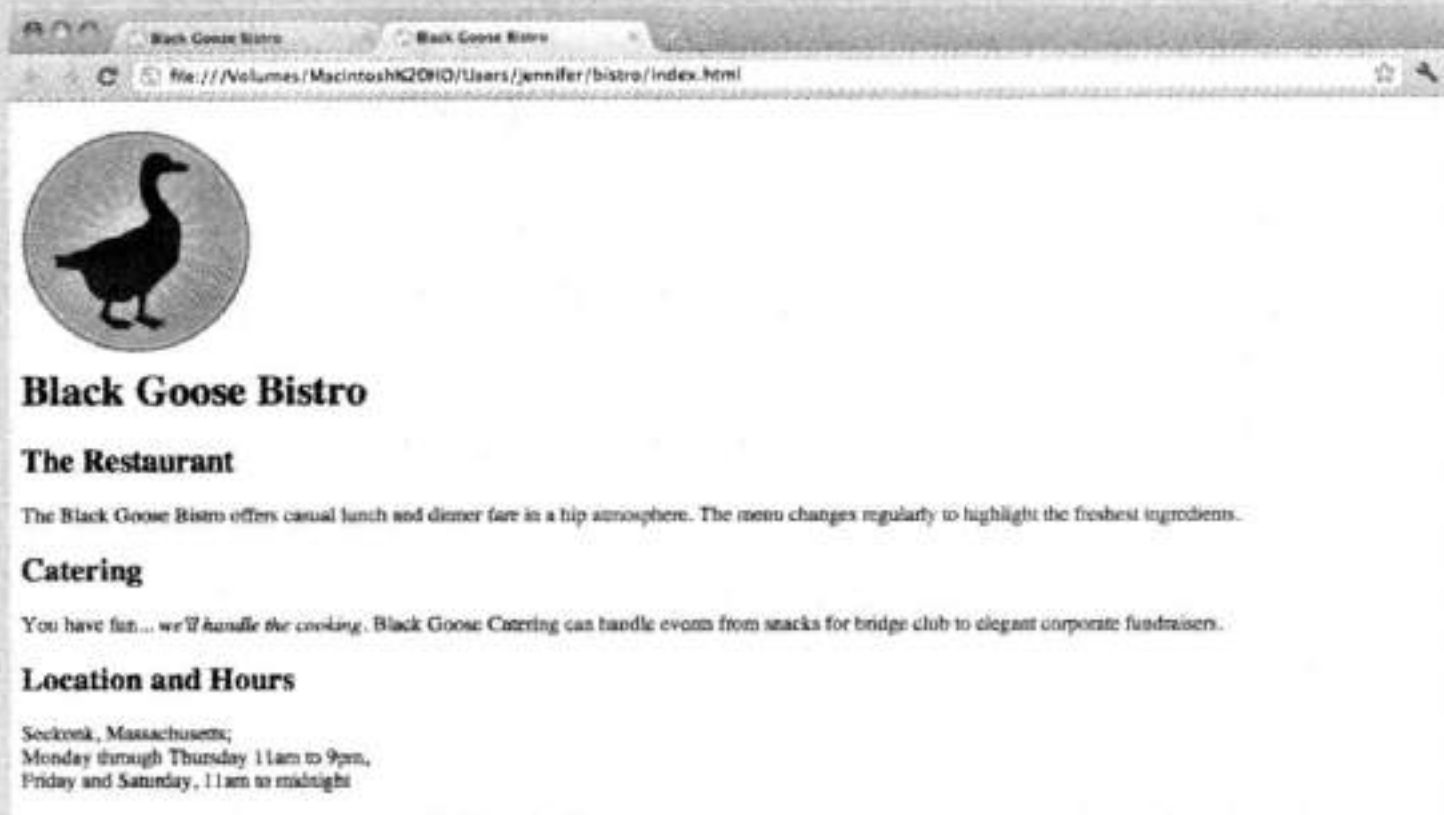


图4-14：含有标志图像的Black Goose Bistro网页

第5步：使用样式表改变外观

根据网站的内容和目的，你可以认为浏览器中文档的默认显示是完全适当的。但是，我想稍稍美化Black Goose Bistro的主页，给潜在的客户留下极佳的第一印象。“美化”只是改变外观的一种说法，这是层叠样式表（CSS）的任务。

在练习4-5中，我们使用一些简单的样式表规则，改变了一些文本元素和网页背景的外观。不用急着立即理解这些知识，CSS的详细内容我们将在第三部分学习。但我想你至少应该了解一点：在使用标记创建的结构之上，加一个表现“层”是什么意思。

练习4-5 添加样式表

1. 如果还没有打开`index.html`，请打开。
2. 我们将使用`style`元素，将内嵌样式表应用到网页中。（这只是添加样式表的方法之一，其他的方法包含在第11章中。）

`style`元素位于文档的`head`元素中。最开始时，添加`style`元素到文档中，如下所示：

```
<head>
  <meta charset="utf-8">
  <title>Black Goose Bistro</title>
  <style>

  </style>
</head>
```

3. 现在，如你所见过的，样式规则的类型已经在`style`元素中了。如果你不能理解它的意思，也不用担心，你将在第三部分学习所有的样式规则。

```
<style>

body {
  background-color: #faf2e4;
  margin: 0 15%;
  font-family: sans-serif;
}

h1 {
  text-align: center;
  font-family: serif;
  font-weight: normal;
  text-transform: uppercase;
```

```
border-bottom: 1px solid #57b1dc;
margin-top: 30px;
}

h2 {
  color: #d1633c;
  font-size: 1em;
}

</style>
```

4. 现在可以保存文档，并在浏览器中查看。它应该如图4-15中的网页。否则，检查样式表代码，确保没有弄丢分号或花括号。



图4-15：应用CSS样式规则之后的Black Goose Bistro网页

我们将要完成Black Goose Bistro的网页了。你不仅编写了你的第一个用样式表完成的网页，而且还学习了元素、特性、空元素、块和内联元素、HTML文档的基本结构，以及标记的正确用法。这一章还是不错的。

当网页出错时

上个示例非常顺利，但在手写HTML标记时，很容易出现小错误。不幸的是，仅仅丢失一个字符都可能会破坏整个网页。下面我故意破坏我的网页，这样你可以看清发生了什么情况。

注意：对于某些块元素（如标题或段落），在闭标签中省略斜线（等效于省略闭标签本身），并不会产生显眼的效果。浏览器将新块元素的开始，当做上一个块元素的结束。

图4-16：当省略斜线时，浏览器不知道元素什么时候结束，如本例

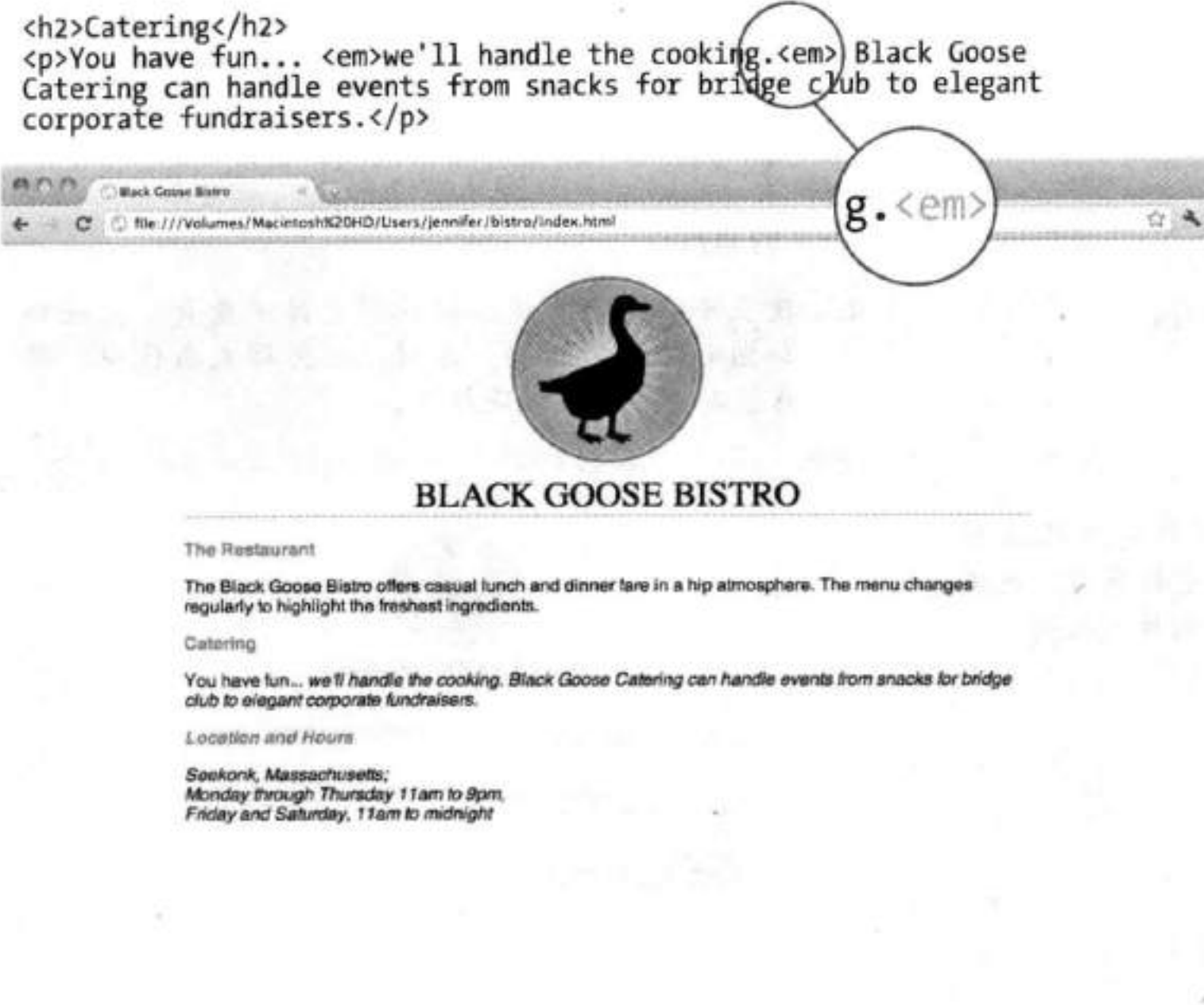


图4-17：缺失结束括号，使后面的所有内容成为这个标签的一部分，造成显示错误



如果我在关闭强调标签（）中忘记输入斜线（/），会怎么样？只是一个字符出错（图4-16），网页中剩下部分都显示为强调（斜体）文本。这是因为没有这个斜线，就等于没有告诉浏览器“关闭”强调格式，所以它一直保持强调格式。

我调整好了斜线，但这次，如果我意外省略了第一个<h2>标签末尾的括号（图4-17），看看会发生什么。

看看标题是如何丢失的？这是因为没有关闭标签括号，浏览器会假设接着的所有文本——直到发现下一个闭括号（>）——是那个<h2>标签的一部分。浏览器不会显示标签中的任何文本，所以标题消失。浏览器会忽略不认识的元素名，然后解析下一个元素。

在第一个HTML文档中犯错并修改，是一个好的锻炼机会。如果你完全正确地编写了第一个网页，我建议你看我一样修改代码，来看浏览器对各处修改的反应如何。这对

后面发现错误并修正网页非常有用。我在侧栏“还有问题？”中列出了一些常见的错误。注意这些错误并不只针对初学者而言，即使是专家，也总会犯这样的小错误。

验证你的文档

专业的网页开发者找到标记错误的方法是对文档进行验证。这是什么意思呢？无论你使用的是哪个版本的HTML(有很多版HTML，我们会在第10章介绍)，验证文档就是检验你的标记，就是要确保你的标记符合规则。没有错误的文档就是有效的。有效的文档在不同的浏览器都是一致的，这就更容易显示，也更便于用户访问。

但是，浏览器并不要求文档一定是有效的（换句话说，浏览器会尽它所能来显示文档，不管有没有错误），但是错误的文档在不同的设备上显示的情况是无法预知的。

如何使你的文档是有效的呢？你应该自己检查，或者问问你的朋友，毕竟人人都会犯错，谁也不可能准确地记住每一条规则。但是你可以使用一个验证器来验证你的文档是否符合你遵循的HTML版本。验证器可以做下面一些事：

- 包含一个DOCTYPE声明。没有文档类型声明，验证器无法知道应该验证哪个版本的HTML或XHTML。
- 文档字符编码的说明。
- 必须的规则和特性。
- 非标准元素。
- 错误匹配的标签。
- 嵌套错误。
- 书写和其他微小错误。

开发者可以使用很多有用的工具来检查和改正HTML文件中的错误。W3C在validator.w3.org提供了一个在线的免费的验证程序。至于HTML 5文件，你可以用[HTML 5.validator.nu](http://HTML5.validator.nu)来验证。浏览器开发者工具像Firefox的Firebug插件，或者Safari和Chrome的插件，都可以用来验证。所以，你能够方便地检查你的工作成果。如果你用Dreamweaver来设计站点，也有一款内置的验证程序。

自我测验

现在开始确认你是否理解了标记的基础知识。使用本章所学到的，回答下列问题。答案在附录A中。

1. 标签和元素有什么不同？
2. 写出一个HTML 5文档的最简结构。
3. 下列文件名是否对于Web文档可接受，请在“是”或“否”上画圈表示。如果不能接受，请说明理由。

还有问题？

下面是一些典型问题，在创建网页并在浏览器中查看时会突然出现。

我修改了文档，但在浏览器中重新载入网页时，跟以前的完全一样。

这可能是因为在重新载入前，你没有保存文档，或者你可能把文档保存在另一个目录中了。

网页有一半不见了。

如果在标签中缺失关闭括号(>)或引号，就可能发生。这是手写HTML时常见的错误。

我使用img元素放入了图片，但显示的是叉号图标。

叉号图标意味着两件事。首先，可能意味着浏览器找不到图像。确保图像文件的URL是正确的（我们将在第6章中进一步讨论URL），确保图像文件真的在你指定的文件夹中。如果在，确保它是Web浏览器能显示的格式之一（GIF、JPEG或PNG），并使用正确的后缀（分别是.gif、.jpeg或.jpg、.png）。

- a. *Sunflower.html*

是

否
- b. *index.doc*

是

否
- c. *cooking home page.html*

是

否
- d. *Song_Lyrics.html*

是

否
- e. *games/rubix.html*

是

否
- f. *%whatever.html*

是

否
4.

下列所有标记例子都有错误。请指出错误，并修正。
- a.

``
- b.

`<i>Congratulations!<i>`
- c.

`linked text</a href="file.html">`
- d.

`<p>This is a new paragraph<\p>`
5.

如何在 HTML 文档中标记下列注释，从而不在浏览器窗口中显示？
- product list begins here

元素回顾：文档结构

本章介绍了元素是如何构建文档结构的。在练习中引入的其他元素都将在后续章节中进一步深入学习。

元素	介绍
body	指明文档中包含内容的body部分
head	确认文档的head部分，包含文档的信息
html	根元素，包含所有其他元素
meta	提供文档的信息
title	设置网页标题

第5章 标记文本

如果你的内容已经准备好了（你确认准备好了吗？），而且已经加入标记（html、head、title和body）来结构化文件，那么接下来该准备识别内容中的元素了。本章介绍了标记文本内容的必选元素。这些元素没有你想得那么多，一般来说，只需要使用少数几个元素就足够了。本章内容比较多，涵盖了很多东西。

当开始学习元素时，我想重申的一个重点是：根据语义选择元素，也就是说，使用最准确的元素来描述内容的含义。如果你不喜欢它的外观样式，可以使用样式表来改变。根据语义来标记的文档对于绝大多数浏览环境都是有效的，无论用户是用台式机访问，还是用移动设备访问，又或者通过辅助屏幕阅读器来访问。而且它还允许机器阅读器（如搜索引擎索引程序）正确地解析你的内容，来获取页面上相对重要的部分。

了解了这些原则，就该看看HTML的文本元素了。首先从其中最简单的元素——段落（paragraph）开始。

重要注意： 我会根据W3C（www.w3.org/TR/html5/）的HTML 5标准来介绍标记，这个标准出现在2012年年中我写本书的时候。实际上WHATWG（whatwg.org）提供了另一种版本（没有版本编号）的HTML。这两个HTML几乎是相同的，但也有一定的差异。我指出的元素和特性都来源于同一个标准。这两个HTML标准经常变化，所以我建议你在网上检查某些元素是否已添加或删除。

你可能听说过，并不是所有的浏览器都支持HTML 5。确实如此。但HTML 5的绝大多数元素几十年前就在早期的HTML版本中出现过，因此它们得到普遍支持。对于HTML 5中的可能还没有得到普遍支持的新元素，我会添加这个标识图：NEW IN HTML 5。因此，如果我没有明确指出这个标识图，你就可以认为这里介绍的标记在所有浏览器都可以得到支持。

本章内容

为你的内容选择最好的元素

段落和标题

三类列表

将内容组织成区段

文本（内联）元素

泛型元素div和span

特殊字符


```
<p>...</p>
```

段落元素

段落

段落是文本文档最基础的元素。可以用p元素表示段落。简单地在段落的开始插入一个<p>开始标记，在结束的地方插入</p>结束标记即可，如下所示。

```
<p>Serif typefaces have small slabs at the ends of letter strokes. In  
general, serif fonts can make large amounts of text easier to read.</p>
```

```
<p>Sans-serif fonts do not have serif slabs; their strokes are square  
on the end. Helvetica and Arial are examples of sans-serif fonts. In  
general, sans-serif fonts appear sleeker and more modern.</p>
```

浏览器通常要在一个新行中展示段落，同时段落与段落之间会默认保留一些空间（这些段落看起来就像一个块，这是CSS中的一个术语）。段落可能包含文本、图像或者其他的内联元素（标准中叫做短语内容）。但是不包括标题、列表或者其他的看起来像块一样的元素。

在HTML中，省略</p>结束标记是可以接受的。当遇到下一个块元素时，浏览器就会假定它结束了。但是在XHTML中，必须有结束标记。许多网页设计者（包括我）为了保持连贯一致和简洁，即使在HTML中，都倾向于结束段落和所有元素。我推荐大家也这么做。

标题

在第4章用h1和h2元素来表示Black Goose Bistro页面的标题。其实有从h1到h6共六级标题。当你给内容添加这些标题时，浏览器就会用它们来生成文档大纲。辅助阅读设备（如屏幕阅读器）可以利用文档大纲来帮助用户更快地浏览页面。此外，搜索引擎的算法会分析标题级别（在级别高的标题中的信息被赋予更高权重）。正因如此，最好在开始时使用h1，然后按数值大小逐级递减，从而创建一个逻辑清晰的文档结构和大纲。

下面的例子展示了四种标题级别的标记。其余的标题级别也可以用相同的方式标记。

```
<h1>Type Design</h1>
```

```
<h2>Serif Typefaces</h2>
```

```
<p>Serif typefaces have small slabs at the ends of letter strokes. In  
general, serif fonts can make large amounts of text easier to read.</  
p>
```

```
<h3>Baskerville</h3>
```

```
<h4>Description</h4>
```

```
<p>Description of the Baskerville typeface.</p>
```

```
<h1>...</h1>
```

```
<h2>...</h2>
```

```
<h3>...</h3>
```

```
<h4>...</h4>
```

```
<h5>...</h5>
```

```
<h6>...</h6>
```

标题元素

注意：你必须给文档中所有文本指派元素。换句话说，所有文本必须包含在某种元素中。没有包括在标签中的文本称为“裸露的”（naked）或“匿名的”（anonymous）文本，这将导致整个文档无效。关于文档验证的更多信息，请参考第4章。

注意：除了标题之外，HTML 5还有一个新的大纲系统。可以看看侧栏“区域内容”了解详情。

```

<h4>History</h4>
<p>The history of the Baskerville typeface.</p>

<h3>Georgia</h3>
<p>Description and history of the Georgia typeface.</p>

<h2>Sans-serif Typefaces</h2>
<p>Sans-serif typefaces do not have slabs at the ends of strokes.</p>

```

这个例子中的标记可以创建下面的文档大纲：

1. Type Design
 1. Serif Typefaces
 - + 文本段落
 - 1. Baskerville
 1. Description
 - + 文本段落
 2. History
 - + 文本段落
 - 2. Georgia
 - + 文本段落
2. Sans-Serif Typefaces
 - + 文本段落

默认情况下，例子中的标题将显示为粗体文本，开始的h1字体非常大，后面各级别标题依次减小，如图5-1所示。可以使用样式表来指定标题级别的外观。

注意：除非特别说明，本书所有的截屏都是在Mac上使用Chrome浏览器截取的。

图5-1：四种标题级别的默认显示

h1 — Type Design

h2 — Serif Typefaces

Serif typefaces have small slabs at the ends of letter strokes. In general, serif fonts can make large amounts of text easier to read.

h3 — Baskerville

h4 — Description

Description of the Baskerville typeface.

h4 — History

The history of the Baskerville typeface.

h3 — Georgia

Description and history of the Georgia typeface.

h2 — Sans-serif Typefaces

Sans-serif typefaces do not have slabs at the ends of strokes.

指示主题转变

`<hr>`

水平分割线

如果你想在两个主题之间添加分隔符，那么可以在文本块之间插入hr元素，这个元素在HTML 5中叫做“段落级别主题中断”（paragraph-level thematic break）。hr是页面区域或者文本段落中的逻辑分隔符，它无需引入新的标题，就可以在文本的区域或段落间加入一个逻辑分隔符。

在HTML 5之前版本的HTML中，hr被定义为“水平线”，因为它会在页面插入一条水平线。浏览器会把hr显示为一条暗色水平线，并在线上、线下都加入一些空行。hr是一个新的语义，但现在有一个新的语义目的。如果一条装饰线是你所要的，最好在元素前后通过CSS来创建一个彩色边框。

hr元素是空元素——你只需要把它放在需要分开的主题处，如下面的例子和图5-2所示。注意，在XHTML中，hr元素末尾必须要有一条斜线：`<hr />`。

```
<h3>Times</h3>
<p>Description and history of the Times typeface.</p>
<hr>
<h3>Georgia</h3>
<p>Description and history of the Georgia typeface.</p>
```

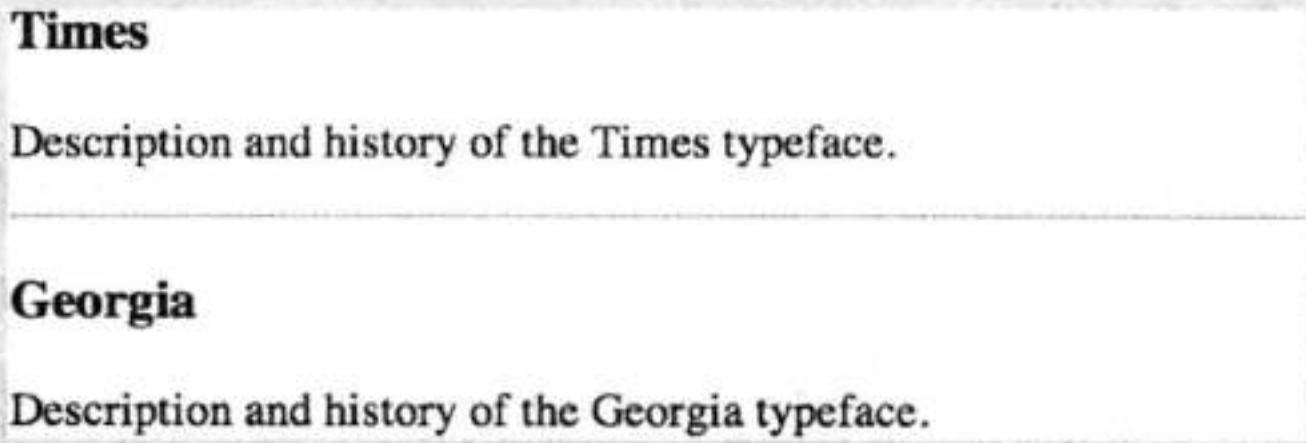


图5-2：水平线的默认显示

群组标题

标题往往会有子标题，就像本书第4章的标题那样：

创建一个简单网页

(HTML概述)

`<hgroup>...</hgroup>`
群组标题

在过去，添加一个子标题是比较麻烦的事情。例子中第一行“创建一个简单网页”是h1，但是如果把第二行作为h2，就会在文档大纲中引入一个新的标题级别。你只能把它标记为一个段落，但是这样做，语义就不同了。

正是这个原因，HTML 5引入了群组元素，可以将标题编为一组（注1）。当前支持群组元素的只有排名最高的浏览器，其余的还做不到。下面是如何使用群组元素来实现如同第4章标题那样的效果。使用这个标记，只有h1“创建一个简单网页”会显示在文档大纲中。

```
<hgroup>
  <h1>Creating a Simple Page</h1>
  <h2>(HTML Overview)</h2>
</hgroup>
```

列表

人类生来就会制作列表，HTML中为三种主要的列表形式提供元素：

- **无序列表。**信息无序显示。
- **有序列表。**按信息重要程度显示。
- **定义列表。**列表由一对对的名字和值组成，可以应用于术语和解释中，但并不局限于此。

所有的列表元素——列表本身和其中的每一项——都像块来显示，也就是说，默认它们都从一个新行开始并且在上面和下面都有一些空间。但是这些可能随着CSS而改变。本节中会详细讲述每种列表类型。

无序列表

只要是与例子、名称、组件、想法或选项的列表相关的都可以用作无序列表。事实上，多数列表都归于此类。默认情况下，无序列表显示时，列表项前都有项目符号，但是可以使用样式表修改，接下来你会看到。

要确定一个无序列表，需要将它标记为ul元素。开始标记在列表首项之前，而结束标记在列表末项之后。然后，各个列表项像例子中一样用li开始和结束标记包围起来，从而被标记为列表项（li）元素。注意，源码中没有项目符号，它们是由浏览器自动添加的（见图5-3）。

```
<ul>
  <li><a href="">Serif</a></li>
  <li><a href="">Sans-serif</a></li>
  <li><a href="">Script</a></li>
  <li><a href="">Display</a></li>
  <li><a href="">Dingbats</a></li>
</ul>
```

注1：虽然群组元素很有用，但是它的前景未知。如果你想在发布的网站中使用这个元素，最好首先查看HTML 5标准。

支持提示：浏览器IE8及之前的IE中还不支持群组元素（hgroup）（参见侧栏“IE对HTML 5的支持”）。老版本的Firefox和Safari（之前的3.6和4版）也没有按照标准提供支持，但是它们并没有完全忽视这个元素，可以对这个元素应用样式。

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

无序列表

```
<li>...</li>
```

无序列表中的列表项

注意：无序列表中（即ul开始和结束标记之间）唯一允许存在的就是一个或多个列表项。不能将其他元素放在里面，也不能有未标签化的文本。然而，可将任何元素（包括其他块元素）放入列表项（li）中。

嵌套列表

任何列表都可以嵌套在另一个列表中；只需要放在列表项中。下面这个例子展示了将一个无序列表嵌套在第二个有序列表项中的方法。

```
<ol>
  <li></li>
  <li>
    <ul>
      <li></li>
      <li></li>
      <li></li>
    </ul>
  </li>
</ol>
```

当将无序列表嵌入另一个无序列表的时候，浏览器会自动改变二级列表的项目符号样式。遗憾的是，当嵌套一个有序列表时，编号样式不会改变。你需要自己使用样式表来设置编号。

- Serif
- Sans-serif
- Script
- Display
- Dingbats

图5-3：无序列表的默认显示。项目符号由浏览器自动添加

下面的例子很酷。可以采用相同的无序列表标记，通过应用不同的样式表，从根本上改变它的外观，如图5-4所示。在图5-4中，我关闭了那些项目符号，添加了自己的项目符号，使列表项水平排成一行，甚至使它们看起来像图片按钮。标记部分完全保持不变。

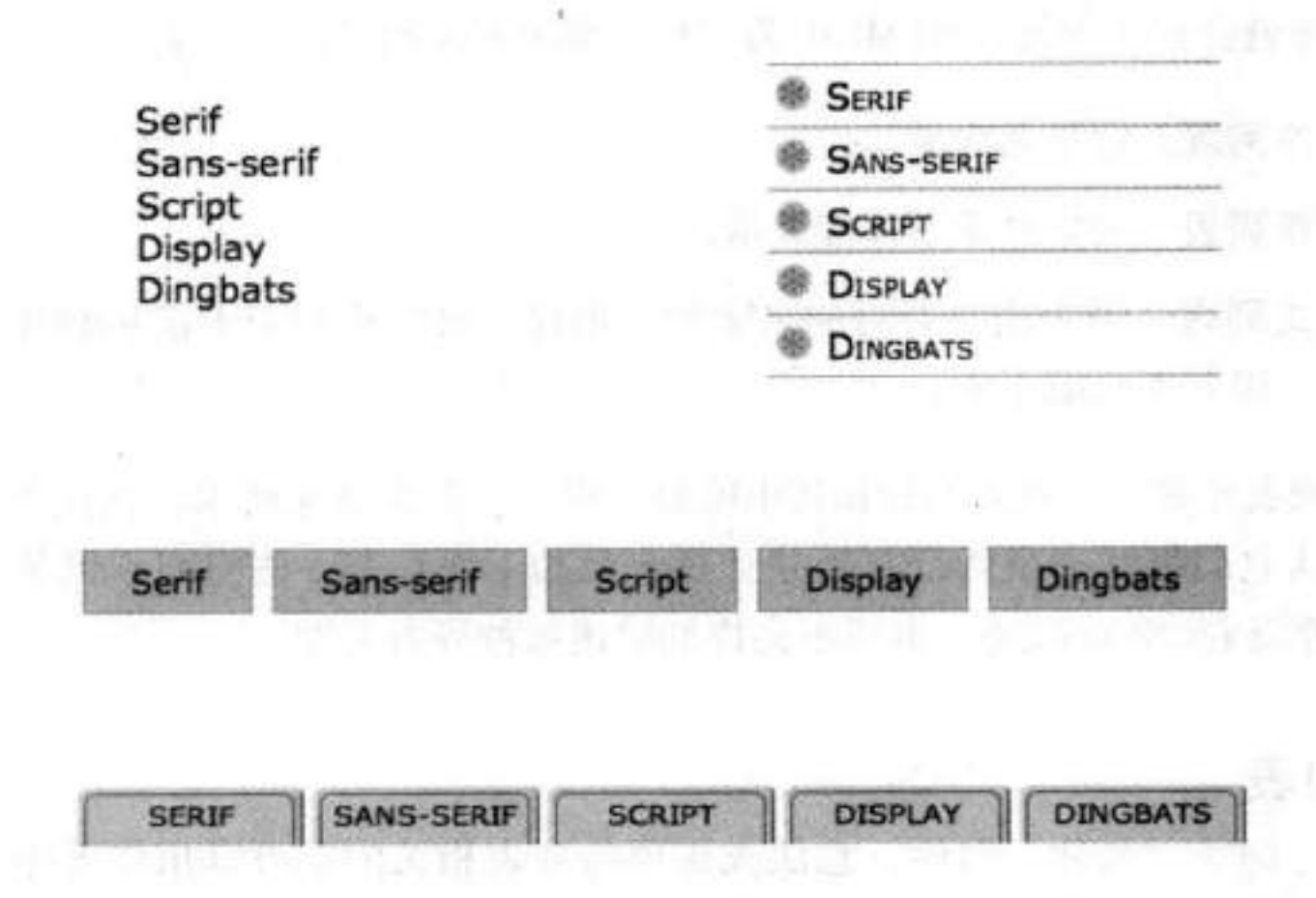


图5-4：使用样式表，可以赋予相同的无序列表截然不同的外观

有序列表

有序列表用于以特定顺序出现的列表项，比如，一步一步地指示或驾驶指令。它们的工作方式类似于前面描述的无序列表，只不过它们用ol元素定义（当然是“ordered list”（有序列表）的缩写）。浏览器自动在有序列表项之前添加数值，而不是项目符号，所以不需要在源码中编号。这样不需要重新编号，就可以容易地重新排列列表项。

有序列表必须包含一个或多个列表项元素，如例子和图5-5所示。

```
<ol>
  <li>Gutenberg develops moveable type (1450s)</li>
  <li>Linotype is introduced (1890s)</li>
  <li>Photocomposition catches on (1950s)</li>
  <li>Type goes digital (1980s)</li>
</ol>
```

1. Gutenberg develops moveable type (1450s)
2. Linotype is introduced (1890s)
3. Photocomposition catches on (1950s)
4. Type goes digital (1980s)

图5-5: 有序列表的默认显示。编号是由浏览器自动添加的

如果你想让列表编号不从1开始, 你可以用`ol`元素中的`start`特性来指明编号的起始数。如下所示:

```
<ol start="17">
  <li>Highlight the text with the text tool.</li>
  <li>Select the Character tab.</li>
  <li>Choose a typeface from the pop-up menu.</li>
</ol>
```

结果列表想将连续编号为17、18和19。

定义列表

定义列表用于任意类型的键值对(表项-定义), 比如术语与其定义、问题及其答案, 或者其他的相关信息。这个结构与我们上面介绍的两种列表有些许不同。整个定义列表都标记为`dl`元素。`dl`元素的内容是一些用于表示表项的`dt`元素和用于表示值的相应`dd`元素。虽然HTML 5中只有一种定义列表用法, 但是为了便于记忆, 我认为把`dt`当做“表项”, `dd`当做“定义”是有帮助的, 因为在英文中表项(term)首字母为t, 定义(definition)首字母为d。

下面的例子说明了一些类型设置及其定义(如图5-6所示)。

```
<dl>
  <dt>Linotype</dt>
  <dd>Line-casting allowed type to be selected, used, then recirculated
into the machine automatically. This advance increased the speed of
typesetting and printing dramatically.</dd>

  <dt>Photocomposition</dt>
  <dd>Typefaces are stored on film then projected onto photo-sensitive
paper. Lenses adjust the size of the type.</dd>

  <dt>Digital type</dt>
  <dd><p>Digital typefaces store the outline of the font shape in a
format such as Postscript. The outline may be scaled to any size for
output.</p>
<p>Postscript emerged as a standard due to its support of
graphics and its early support on the Macintosh computer and Apple
laser printer.</p>
</dd>
</dl>
```

改变项目符号和编号

可以使用`list-style-type`样式表属性, 来改变列表的项目符号或编号。例如, 对于无序列表, 可以把默认的点改为正方形或空心圆, 或者用自己的图像替换, 或者干脆都去掉。对于有序列表, 可以将数字改为罗马数字(I、II、III. 或 i、ii、iii.)、字母(A、B、C、或 a、b、c.) 或者其他的编号方案。事实上, 只要列表按照语义正确标记, 就没有必要显示符号和数字编号。使用CSS改变列表样式的相关知识将在第18章中讲述。

```
<dl>...</dl>
```

定义列表

```
<dt>...</dt>
```

一个名字, 如一个表项或者标签

```
<dd>...</dt>
```

一个值, 如一个描述或者定义

分段根元素

`blockquote`是一种分段的根元素。分段根元素的标题不会被主文档大纲包含。也就是说，在一个`blockquote`中，可以有比较复杂的标题层级，而不需担心这样会影响文档的总体结构。其他的分段根元素包括`figure`、`details`、`fieldset`（用于组织表单域）、`td`（表格单元）和`body`（主体元素）元素。主体元素有自己的大纲，它的大纲可以成为文档的大纲。

Linotype

Line-casting allowed type to be selected, used, then recirculated into the machine automatically. This advance increased the speed of typesetting and printing dramatically.

Photocomposition

Typefaces are stored on film then projected onto photo-sensitive paper. Lenses adjust the size of the type.

Digital type

Digital typefaces store the outline of the font shape in a format such as Postscript. The outline may may be scaled to any size for output.

Postscript emerged as a standard due to its support of graphics and its early support on the Macintosh computer and Apple laser printer.

图5-6：定义列表的默认显示。定义通过缩进与表项区分开

只有`dl`元素才允许包含`dt`和`dd`元素。一个表项可以有多个定义，反之亦然。不能将标题和内容组织元素（如段落元素）放入`dt`中，但是`dd`可以包含任何类型的流内容（flow content）。

更多内容元素

目前已经学习了段落、标题和列表元素，但是还有一些不太容易归类的特殊文本元素需要增加到HTML工具箱中，包括长引用（`blockquote`）、预格式化文本（`pre`）和独立的流内容元素（`figure`和`figcaption`）。这些元素的共同点是，HTML 5标准都把他们看做一种“分组内容”（类似地，还有`p`、`hr`、列表元素和稍后会讲的泛型`div`）。另一个共同点是浏览器默认会把它们显示为一个块元素。

长引用

如果你有一个长引用、证明书，或者从其他地方复制的小节，你应该把它们标记为`blockquote`。`blockquote`元素的内容可以包含在其他元素中，如段落、标题或列表，如下所示（见侧栏“分段根元素”）：

```
<p>Renowned type designer, Matthew Carter, has this to say about his profession:</p>
```

```
<blockquote>
```

```
<p>Our alphabet hasn't changed in eons; there isn't much latitude in what a designer can do with the individual letters.</p>
```

```
<p>Much like a piece of classical music, the score is written down - it's not something that is tampered with - and yet, each conductor interprets that score differently. There is tension in the
```

```
<blockquote>...</blockquote>
```

长的块级引用

```
interpretation.</p>
</blockquote>
```

图5-7展示了blockquote的默认显示。这个可以用CSS改变。

Renowned type designer, Matthew Carter, has this to say about his profession:

Our alphabet hasn't changed in cons; there isn't much latitude in what a designer can do with the individual letters.

Much like a piece of classical music, the score is written down. It's not something that is tampered with, and yet, each conductor interprets that score differently. There is tension in the interpretation.

图5-7: blockquote元素的默认显示

预格式化文本

在很早以前，我们就知道浏览器忽略源文档中的空白元素（如回车和字符空格）。但在一些类型的信息中，如示例代码或诗中，空格对表达含义很重要。基于以上目的，产生了预格式化文本（pre）元素。它是唯一的显示与拼写完全一致的元素——包括所有的回车和多个字符空格都保留。默认情况下，预格式化文本以等宽字体显示（所有字符都是相等宽度，也称作monospace），如Courier字体。

例子中的pre元素如图5-8所示。图5-8的第二部分展示了相同的内容，但为了与pre作比较，我们以段落（p）元素标记。

```
<pre>
This is          an          example of
    text with a    lot of
                    curious
                    white space.
</pre>

<p>
This is          an          example of
    text with a    lot of
                    curious
                    white space.
</p>
```

注意：还有一个内联元素q，是文本流中的短引用。将在本章后面谈论。

```
<pre>...</pre>
```

预格式化文本

注意：CSS属性white-space:pre可以用于保留源码中的空格和回车。不像pre元素，使用white-space属性格式化后的文本不能用等宽字体显示。


```
This is          an          example of
    text with a      lot of
                        curious
                        white space.
```

This is an example of text with a lot of curious white space.

图5-8：浏览器中显示空白，如同源码中一样，预格式化文本是唯一一个具有这样效果
的元素。相比之下，段元素会忽略其中的回车，多个字符空白只留一个

figure

figure元素用于给文本添加图片或者阐明文本中的某些内容。一个figure可能包含一个图片、一段视频、一段代码、文本或者一个表格——几乎是网页内容中的任何流内容——而且是一个独立的单元。也就是说，如果figure元素从原来的流文件的某个位置移除（比如，移到一个侧栏或者附录），无论是原来的流文件还是figure都还是有意义的。

虽然可以很简单地把一个图像置入一个文本中，但是如果能用figure标签把这个图像包起来，会使这个图像的意义更明确。而且可以对figure应用样式，而不影响页面上的其他图片。

```
<figure>
  
</figure>
```

使用可选的figcaption元素可以为figure添加一段说明，可以添加在figure内容的上方或者下方。

```
<figure>
  <pre><code>
    body {
      background-color: #000;
      color: red;
    }
  </code></pre>
  <figcaption>
    Sample CSS rule.
  </figcaption>
</figure>
```

在练习5-1中，可以自己标记一个文档，可以试着使用之前讲过的那些基本文本元素。

<figure>...</figure>
联系信息

<figcaption>...</figcaption>
联系信息

特别注意： figure和figcaption元素在IE 8及其之前的版本中是不支持的（见本章稍后的侧栏“Internet Explorer对HTML 5的支持”）。老版本的Firefox和Safari（分别是3.6版和4版）没有按照HTML 5规约提供支持，但是可以应用样式。

练习5-1 标记一个食谱

Black Goose Bistro 的所有者决定建立一个博客来分享食谱和通知。在这个练习中，我们将帮他们编写内容标记。

下面你将看到一个食谱网页的纯文本。由你来添加文档结构元素，内容的每个块与哪个元素最符合，都由你决定。你可以使用段落元素、标题元素、列表元素，并且

至少要使用一个特殊内容元素。

你可以直接在网页上编写标签，也可以使用文本编辑器，并在浏览器中查看结果，文本文件可以从www.learningwebdesign.com/4e/materials在线获得。结果代码在附录A中。

Tapenade (Olive Spread)

This is a really simple dish to prepare and it's always a big hit at parties. My father recommends:

"Make this the night before so that the flavors have time to blend. Just bring it up to room temperature before you serve it. In the winter, try serving it warm."

Ingredients

1 8oz. jar sundried tomatoes
2 large garlic cloves
2/3 c. kalamata olives
1 t. capers

Instructions

Combine tomatoes and garlic in a food processor. Blend until as smooth as possible.

Add capers and olives. Pulse the motor a few times until they are incorporated, but still retain some texture.

Serve on thin toast rounds with goat cheese and fresh basil garnish (optional).

组织网页内容

到目前已经详细讲过了这些元素：段落元素、标题元素、独立的流内容元素等。在HTML 5之前，没有办法将这些元素组成较大的部分，除非把它们包裹在一个划分（div）元素（我将在后面详细介绍div）中。HTML 5引入了新的元素来提供这种语义（表意），包括区段（section，也可以叫做章节）、独立内容（article）、导航（nav）、侧栏（aside）、页眉（header）和页脚（footer）。新的元素名称是基于Google的研究，你可以看看给泛型分隔元素提供的前20个名字（code.google.com/webstats/2005-12/classes.html）。奇怪的是，规范将从前的address元素定义为一种区段，这里也会对此进行讲解。

目前的桌面和移动浏览器可以很好地支持本节中所讨论的元素，但在IE 8及其更早的版本上还不能支持。在侧栏“Internet Explorer对HTML 5的支持”中可以了解详细信息。

Internet Explorer对HTML 5的支持

当前大多数浏览器都支持HTML 5的新语义（表意）元素，对于无法支持的浏览器，就需要创建一个样式表规则来告诉浏览器把这些元素当做一个块级的元素来处理，这样浏览器就可以了。

```
section, article, nav, aside, header, footer,
hgroup { display: block; }
```

遗憾的是，这种方案在Internet Explorer 8及其更早的版本中是不适用的，IE 9及之后的版本都支持。早期版本的IE浏览器不仅无法识别新的元素，也无法使用任何样式。解决的办法是使用JavaScript来创建每个元素，使IE浏览器知道它的存在，并允许嵌套和样式。如下所示是创建区段元素（section）的一个JavaScript命令：

```
document.createElement("section");
```

幸运的是，Remy Sharp创建了一个脚本，它将所有的HTML 5元素全部包装成一个副本，这个脚本可以称为

“HTML 5 Shiv”，它在Google运行的服务器上，所以你可以在你的文档中指向这个脚本。为了确保新的HTML 5元素在IE8及其更早的版本中正常工作，将这段代码复制到你的文档的头部，并且使用样式表来使这些新元素看起来像一块内容（block）：

```
<!--[if lt IE 9]>
<script src="http://html5shiv.googlecode.com/svn/trunk/html5-els.js"></script>
<![endif]-->
```

在[HTML 5doctor.com/how-to-get-HTML 5-working-in-ie-and-firefox-2/](http://HTML5doctor.com/how-to-get-HTML-5-working-in-ie-and-firefox-2/)中可以找到HTML 5 Shiv的更多资料。

这个HTML 5 Shiv脚本也是Modernizr polyfill脚本的一部分，Modernizr polyfill脚本可以将HTML 5和CSS3功能添加到老版的浏览器中。你可以在modernizr.com读到更多的信息。第20章也会进行讨论。

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

区段

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

独立的内容

注意：HTML 5规约建议，如果组装这些元素仅是简单地提供样式提供一个钩子，那么不如使用泛型div元素。

区段和独立内容

长文档划分成更小的部分时更容易使用。例如，书籍分为章节，报纸为地方新闻、体育、漫画提供不同的版面等。section元素可以将长长的网页文件分割成专题栏目。section元素内有一个标题，其他有意义的内容都会组织在一起。

section元素具有广泛的用途，它可以将整个网页划分为主要章节区段，或将一篇文章分成主题章节。在下面的示例中，与印刷术资源信息有关的文件已根据资源类型分为两部分。

```
<section>
  <h2>Typography Books</h2>
  <ul>
    <li>...</li>
  </ul>
</section>

<section>
  <h2>Online Tutorials</h2>
  <p>These are the best tutorials on the web.</p>
  <ul>
    <li>...</li>
  </ul>
</section>
```

可以为独立的工作或者能够在于不同上下文中重用的内容使用article元素。这个元素对于杂志或报纸上的文章、博客文章、评论，或其他可以在外部摘录的内容是很有用的。你可以认为它是一个特殊的区段元素。

如果你认为某个内容可以出现在其他的网站上，那么就可以为这个内容使用独立内容元素。

为了使事情变得有趣，一个长的`article`可以分成不同的区段，如下所示：

```
<article>
  <h1>Get to Know Helvetica</h1>
  <section>
    <h2>History of Helvetica</h2>
    <p>...</p>
  </section>

  <section>
    <h2>Helvetica Today</h2>
    <p>...</p>
  </section>
</article>
```

同时，一个`section`也可以分成一些独立的内容。

```
<section id="essays">
  <article>
    <h1>A Fresh Look at Futura</h1>
    <p>...</p>
  </article>

  <article>
    <h1>Getting Personal with Humanist</h1>
    <p>...</p>
  </article>
</section>
```

`section`元素和`article`元素很容易混淆，尤其是因为它们可以互相嵌套。但是要记住，如果内容是独立于文档的其他部分，可以在其他地方使用，最好标记为`article`。

分段元素

`section`和`article`的另一个共同点是，它们都是HTML 5的分段元素。当浏览器解析到一个文档中的分段元素时，它会给这个文档的大纲自动创建一个新的项目。在以往的HTML版本中，只有标题（`h1`、`h2`等）会生成新的大纲项目。新的元素`nav`（尤其是它）和`aside`也都是分段元素。

在新的HTML 5大纲系统中，不管分段元素在文档中的位置如何，它都可以有自己的内部标题层次。这就使得将`article`及其内部大纲拿出来，放在另一个文件流是可能的，并且不会破坏主文件的大纲。新的大纲算法的目标是使标记能够满足内容使用和在现代Web上重用的需求。

截止写本书时，还没有浏览器支持HTML 5大纲系统，所以为了使你的文件对所有用户都是可访问和逻辑结构清晰的，最好按级次递减来使用标题元素，在分段元素中也如此。

想要了解更多信息，我推荐html5的博士文章“Document Outlines”，由Mike Robinson撰写，他更详细地阐述了这个问题（HTML5doctor.com/outlines）。

此外，Roger Johansson的文章“HTML 5 Sectioning Elements, Headings, and Document Outlines”介绍了一些使用分段元素的潜在的陷阱（www.456bereastreet.com/archive/201103/html5_sectioning_elements_headings_and_document_outlines/）。

`<aside>...<aside>`
有联系却无关的内容

侧栏（或边栏）

`aside`元素可以指与主要内容有联系但无关的内容。在印刷中，它相当于一个侧栏，但是这个元素名字不是`sidebar`，这是因为“`sidebar`”更像是一个外观的描述，而不是语义（表意）的词汇。虽然如此，侧栏是使用`aside`元素的好地方。对于展示引用、背景信息、链接列表、标注以及任何与文档有关（但并不是非常相关）的内容都可以使用`aside`。

在下面的例子中，`aside`元素用于展示与主文档有关的链接列表。

```
<h1>Web Typography</h1>
<p>Back in 1997, there were competing font formats and tools for making
them...</p>
<p>We now have a number of methods for using beautiful fonts on web
pages...</p>
<aside>
  <h2>Web Font Resources</h2>
  <ul>
    <li><a href="http://typekit.com/">Typekit</a></li>
    <li><a href="http://www.google.com/webfonts">Google Fonts</a></li>
  </ul>
</aside>
```

`aside`元素没有默认的显示形式，所以需要把它放在一个块元素中，然后使用样式表规则来调整它的外观和布局。

`<nav>...</nav>`
主导航链接

导航

`nav`是一个新元素，它为开发者建立一个站点的导航提供了语义（表意）上的方式。本章开头介绍过一个可以用作字体目录站点的顶级导航的无序列表。把那个列表（`list`）放在一个`nav`导航元素中，会使它一目了然。

```
<nav>
<ul>
  <li><a href="">Serif</a></li>
  <li><a href="">Sans-serif</a></li>
  <li><a href="">Script</a></li>
  <li><a href="">Display</a></li>
  <li><a href="">Dingbats</a></li>
</ul>
</nav>
```

但并不是所有的链接列表都可以放在`nav`标签中。标准中的解释是，导航`nav`元素应该用在一个站点或一长节或文章的主要导航链接上。

从可访问性的角度看，`nav`元素是非常有用的。一旦屏幕阅读器或者其他设备兼容HTML 5标准，用户就可以很容易地获取（或者跳过）导航章节。

页眉和页脚

由于网页作者一直在各自的文档中标明页眉和页脚，所以引入header和footer元素是很自然的。下面先从页眉元素开始。

页眉

header元素用在网页开始或者文章顶部的介绍性材料上。关于header应该包含什么没有明确的说明，任何对于页面或者文章有介绍意义的都可以。在下面的例子中，文档header包含了一个图标、站点名称和导航。

```
<header>
  
  <hgroup>
    <h1>Nuts about Web Fonts</h1>
    <h2>News from the Web Typography Front</h2>
  </hgroup>
  <nav>
    <ul>
      <li><a href="">Home</a></li>
      <li><a href="">Blog</a></li>
      <li><a href="">Shop</a></li>
    </ul>
  </nav>
</header>

...page content...
```

当在一个单独的文章中使用header元素时，应该包含文章标题、作者和出版日期，如下所示：

```
<article>
  <header>
    <h1>More about WOFF</h1>
    <p>by Jennifer Robbins, <time datetime="11-11-2011"
      pubdate>November 11, 2011</time></p>
  </header>
  <p>...article content starts here...</p>
</article>
```

页脚

footer元素用于指明页面或文章结束时的信息，如作者、版权信息、相关文档或者导航。footer元素可以应用在整个文档，或者某个特定文章上。如果页脚直接包含在body元素中，而不是在body内容之前或之后，那么它就可以应用于整个页面或应用上。如果页脚包含在一个分段的元素中（section、article、nav或aside），它就只会是相应分段的页脚。

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

网页、章节、文章的介绍材料

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

网页、章节、文章的页脚

警告： Header或footer元素都不允许再嵌套header或footer元素。

注意：虽然它叫做“页脚”，但并不是说它一定要出现在文档或者分段元素的最后。只要使用的语义（表意）正确，就可以放在文档或分段元素的开始。

在下面的例子中，会介绍用`footer`标记的一篇文章或者博客底部的典型列表信息：

```
<article>
  <header>
    <h1>More about WOFF</h1>
    <p>by Jennifer Robbins, <time datetime="11-11-2011"
      pubdate>November 11, 2011</time></p>
  </header>
  <p>...article content starts here...</p>
  <footer>
    <p><small>Copyright &copy;2012 Jennifer Robbins.</small></p>
    <nav>
      <ul>
        <li><a href="">Previous</a></li>
        <li><a href="">Next</a></li>
      </ul>
    </nav>
  </footer>
</article>
```

地址

```
<address>...</address>
```

联系信息

最后简单说一下，`address`元素用来为作者或者文档的维护者创建一个联系信息。`address`一般放在文档或者一节或文章的结尾部。`address`可以放在`footer`元素中。

值得注意的是，`address`元素不能用于网页的“旧地址”，如邮件地址。它是用于作者联系信息（虽然邮件地址也是联系信息）。下面的例子是它的用法示例。“`a href`”是链接地址的标记，第6章将讲述它。

```
<address>
  Contributed by <a href="../authors/robbins/">Jennifer Robbins</a>,
  <a href="http://www.oreilly.com/">O'Reilly Media</a>
</address>
```

内联元素综述

既然我们已经学习了很多内容，就可以使用HTML 5中的文本级别语义（表意）元素对它们进行分类。之前也提过内联元素，因为它们显示在文本流中，且不会引起换行。在HTML 5之前的HTML中，它们也是如此。

文本级别（内联）元素

虽然你能够给文档添加所有类型的信息，但HTML 5中也只有几十个这种元素而已。表5-1是这些元素的完整列表。

注意：在本章结束部分的练习5-3中，你会有机会使用这些元素。

表5-1：文本级别语义（表意）元素

元素	描述
a	超链接（详情参见第6章）
abbr	缩写
b	引起视觉注意，如关键字（粗体）
bdi	定义文本的文本方向
bdo	定义文本显示方向（由左到右ltr，从右到左rtl）
br	换行
cite	引用，可以引用某个标题，如书的标题
code	计算机代码示例
data	机器可读的日期、时间、权重和其他测量值
del	删除文本；表明是对文档的编辑
dfn	定义或者第一次出现的术语
em	强调文本
i	斜体文本
ins	插入文本；表明是对文档的一个插入
kbd	键盘；由用户输入的文本（技术文档使用）
mark	前后相关的文本
q	短的内联引用
ruby,rt,rp	使用东亚文本和文字提供解释或发音帮助
s	不正确的文本（加删除线）
samp	程序样本
small	小字显示，如法律或者版权提示（用小一点尺寸的字体）
span	一般分段内容
strong	重要的内容
sub	下标
sup	上标
time	机器可读的时间数据
u	下划线文本
var	程序（或技术文档）中的一个变量
wbr	单词换行

内联元素的前世今生

由于缺少样式表系统，自Web诞生以来就有的很多内联元素都是用来改变文本的可视化格式的。如果想要一个黑体文本，你可以把文本标记为b。斜体怎么办？可以使用i元素。事实上，曾经有一个font元素，仅用于改变字体、颜色和大小。毫不奇怪，HTML 5去掉了用于表象的font元素。然而，也有许多曾经的表象内联元素（例如，表示下划线的u元素和表示删除线的s元素）一直保持在HTML 5中，并被赋予新的语义（表意）定义（b元素现在是“关键词”，s元素是“不准确的文字”）。

一些内联元素是纯粹语义（表象）的（如abbr或time），并没有默认的显示样式。对于这些元素，如果你想改变它们的显示方式，就需要使用一个CSS规则。

在本节中描述的元素，如果有一个内联元素的定义和预期的浏览器默认显示方式，我会提供二者。

废弃的HTML 4.01的文本元素	
HTML 5取消了HTML 4.01中许多废弃（淘汰和不建议使用）的元素。为了彻底理清，如果你想把它们当作遗产来继续使用，可以看看下表。但是没有任何理由继续使用它们，因为大多数这些元素要么有类似的样式表属性可以实现，要么浏览器支持很差。	
元素	描述
acronym	表示一个首字母简写（如NASA）;可以使用abbr代替
applet	插入一个Java applet
basefont	为文档建立一个默认字体设置
big	使文本比默认文本稍微大一些
center	使内容水平居中
dir	目录列表（由无序列表替代）
font	字体外貌、颜色和大小
isindex	插入一个搜索框
menu	菜单列表（由无序列表替代）；新版menu用于排列表单控件
strike	带删除线的文本
tt	打字机文本；显示为等宽字体

`...`
强调内联文本

强调文本
使用em元素来强调句子的一部分。em元素的位置会影响句子的意思。看看下面的句子，字面是相同的，但是强调的意义不一样。

```
<p><em>Matt</em> is very smart.</p>  
<p>Matt is <em>very</em> smart.</p>
```

第一句强调的是“谁”很聪明，第二句强调的是“多么”聪明。

em文本总是默认显示为斜体（如图5-9所示），当然，你可使用样式表随意改变外观。屏幕阅读器会用不同的声调来表示强调文本，所以使用em元素只是为了表达意义，并不是要使文本成为斜体。

`...`
特别强调内联文本

重要文本
strong元素表示一个词和短语是很重要的。下面的例子中，strong元素标明了需要额外注意的说明。


```
<p>When checking out of the hotel, <strong>drop the keys in the red box by the front desk</strong>.</p>
```

可视化浏览器总是把strong文本显示为粗体。屏幕阅读器可以用不同

的声调来读重要内容，所以把文本标记为**strong**，只是为了意义上的明确，并不是为了把它变为粗体。

下面是em和strong的一个简单例子，如图5-9所示，应该和你所想的一样吧？

Matt is very smart.

Matt is *very* smart.

When returning the car, **drop the keys in the red box by the front desk.**

图5-9：em和strong文本的默认显示

原先的表象元素在HTML 5中有新的语义（表意）定义

只要谈论粗体和斜体文本，就可以看看原先的b和i元素现在是什么样的。元素b、i、u、s和small过去都是用来提供排版指令（粗体、斜体、下划线、删除线和较小的文字）。尽管它们原先是用于表象的，但是在HTML 5中，它们已经有新的基于语义的定义。浏览器会用默认方式来显示它们（见图5-10）。然而，如果你想要改变字体样式，使用样式表规则是一个适当的解决方案。

下面介绍这些元素和它们的正确用法，以及样式表替代方案。

b

HTML 4.01定义：粗体

HTML 5定义：关键词、产品名称和其他短语，需要脱颖而出，而不需增加重要性的文字。

CSS替代方案：对粗体文本，可以使用font-weight。如font-weight:bold

例如：

```
<p>The slabs at the ends of letter strokes are called <b>serifs</b>.</p>
```

i

HTML 4.01定义：斜体

HTML 5定义：表示该文本与周围的文本不同，应该用不同的声音。比如，其他语言的一个短语、一个技术术语或一种思想。

```
<b>...</b>
```

关键词或看上去重要的文本（粗体）

```
<i>...</i>
```

不同的语调（斜体）

```
<s>...</s>
```

不正确的文本（带删除线的文本）

```
<u>...</u>
```

注释文本（下划线）

```
<small>...</small>
```

法律文本，小字打印（更小的字体）

注意：这样我想到屏幕阅读器如何读取文本。如果我不希望以一个响亮的、强调的语气读出这些文本，但又确实要用粗体，那么b元素比strong元素可能更合适。

CSS替代方案：对于斜体文本，可以使用font-style。如font-style:italic

例如：

```
<p>Simply change the font and <i>Voila!</i>, a new personal-ity.</p>
```

s

HTML 4.01定义：带删除线的文本

HTML 5定义：表示该文本不正确。

CSS属性：对选中的文本加一条删除线，可以使用text-decoration。如text-decoration: line-through

例如：

```
<p>Scala Sans was designed by <s>Eric Gill</s> Martin Majoor.</p>
```

u

HTML 4.01定义：下划线

HTML 5定义：有一些实例中，下划线是有语义含义的，比如，在中文的正式名字下加下划线，或者拼写检查后，拼写错的词下会加下划线。注意，下划线文本容易与链接地址混淆，除了一些必要情况，应该避免使用。

CSS属性：对于带下划线的文本，可以使用text-decoration。如text-decoration: underline

例如：

```
<p>New York subway signage is set in <u>Halvetica</u>.</p>
```

small

HTML 4.01定义：以比周围其他文字小的字体显示

HTML 5定义：表示是主内容的附录或者侧栏提示，比如，文档底部的法律样的“小字”。

CSS属性：为了使文本小一点，可以使用font-size。如font-size:80%

例如：

```
<p><a href="">Download <b>Jenville Handwriting Font</b> </a></p>
<p><small>This font is free for commercial use.</small>
</p>
```

The slabs at the ends of letter strokes are called **serifs**.

Simply change the font and *Voila!*, a new personality!

Scala Sans was designed by ~~Erie-Gill~~ Martin Majoor.

New York subway signage is set in Halvetica.

Download Jenville Handwriting Font

(This font is free for personal and commercial use.)

图5-10: b、i、u、s和small元素的默认显示

短引用

可以使用引用（q）元素来标记短引用，比如，文本流中的“To be or not to be”，如下例所示（见图5-11）。

Matthew Carter says, <q>Our alphabet hasn't changed in eons.</q>

根据HTML规范，浏览器应该在q元素两侧自动添加引号，这样就不需要在源码中使用引号。但是浏览器IE 7和之前版本对此不支持。幸运的是，在写本书时，那些不自动添加引号的浏览器的使用率只有5%~8%，在你读到本书时，这样的浏览器会更少。如果你关心这些低使用率的浏览器的用户，那么最好在你的源码中添加引号。

Matthew Carter says, "Our alphabet hasn't changed in eons."

图5-11: 几乎所有的浏览器都会给q元素增加引号

缩写和首字母简写

用abbr元素表示标记简写和缩写，可以给搜索引擎、屏幕阅读器和其他设备提供有用的信息。缩写是将以句点结束的单词（如将Connecticut简写为Conn.）缩短。简写是用短语中单词的首字母组成的缩写形式（如WWW或者USA）。title特性可以用于缩写，如下面的例子所示。

```
<abbr title="Points">pts.</abbr>
<abbr title="American Type Founders">ATF</abbr>
```

引用

cite元素用于表示其对他文档的引用，如书、杂志、文章标题等。引用默认以斜体显示，下面是一个例子：

<p>Passages of this article were inspired by <cite>The Complete Manual of Typography</cite> by James Felici.</p>

<q>...</q>

短内联引用

提示

嵌套元素

可以对一个文本字符串（例如，一个短语既是一个引用，又是另一种语言文字）使用两个元素，但需确保它们嵌套正确。这意味着内部元素，即结束标记，必须完全包含在外部元素中，并且不重叠。

<q><i>Je ne sais pas.</i></q>

<abbr>...</abbr>

缩写或者首字母简写

注意：在HTML 4.01中，**acronym**元素用于首字母简写，但是在HTML 5中，由于两者都可以使用abbr，acronym就取消了。

<cite>...</cite>

引用

<code><dfn>...</dfn></code> 定义术语	定义术语 在出版界，单词或术语的第一次出现和定义实例，经常会以特殊的风格标示。本书中，定义的术语设置为蓝色文字。在HTML中，可以用dfn元素标识它们，使用样式表在视觉上格式化它们。 <code><p><dfn>Script typefaces</dfn> are based on handwriting.</p></code>
<code><code>...</code></code> 代码	程序代码元素 许多内联元素用于描述科技文档中的一部分，如代码（code）、变量（var）、程序示例（samp）和用户键盘输入（kbd）。对于我来说，它是科学世界中HTML起源的精巧的提醒器（Tim Berners-Lee于1989年在CERN粒子物理实验室开发了HTML，用于共享文档）。
<code><var>...</var></code> 变量	
<code><samp>...</samp></code> 程序示例	
<code><kbd>...</kbd></code> 用户键盘输入文本	
<code><sub>...</sub></code> 下标	下标和上标 下标（sub）和上标（sup）元素可以将选中的文字以小一点的大小来显示在基线下方或者上方。这些元素对于指明化学式或者数学式是很有用的。
<code><sup>...</sup></code> 上标	

图5-12显示了下标和上标在浏览器上的典型显示。

```
<p>H<sub>2</sub>O</p>
<p>E=MC<sup>2</sup></p>
```



图5-12：下标字和上标字

<code><mark>...</mark></code> 内容相关的文字	突出显示的文本 新的mark元素表示一个字可能被认为与读者特别有关。有人可能会使用它调出搜索项的结果页面，或者引起对文本的注意，表明当前页面都属于一个系列。一些设计师（和浏览器）会给标记文本浅色背景，就像是用荧光笔标记的，如图5-13所示。
--	--

```
<p> ... PART I. ADMINISTRATION OF THE GOVERNMENT. TITLE IX.
TAXATION. CHAPTER 65C. MASS. <mark>ESTATE TAX</mark>. Chapter 65C:
Sect. 2. Computation of <mark>estate tax</mark>.</p>
```

... PART I. ADMINISTRATION OF THE GOVERNMENT. TITLE
IX. TAXATION. CHAPTER 65C. MASS. ESTATE TAX. Chapter
65C: Sect. 2. Computation of estate tax.

图5-13: 搜索项会被标记为mark元素, 并会使用样式表提供一个黄色背景, 从而使它们更容易被读者找到

时间和机器可读信息

当我们看到“11月4日中午”时, 我们知道这是一个日期和时间。但对于一个电脑程序来说, 文本内容可能并不明显。time元素可以用方便人舒适阅读的格式来标记日期和时间, 而且也会用方便计算机使用的标准化的方式编码。元素的内容给人提供信息, datetime属性会以机器可读的方式呈现相同的信息。

time元素显示日期、时间或日期时间组合。它可能用来传递日期时间到应用程序, 比如, 把日期和时间信息保存到个人日历事件中。它可能会被搜索引擎用于查找近期公布的文章。或者, 它可能用来重新设计时间信息的格式(例如, 将18:00改成6 p.m.)。

datetime属性以标准时间格式来指定日期或时间的信息, 如图5-14所示。它先显示日期(年、月、日), 然后显示时间(T表示时间), 列出了小时、分钟、秒(可选)、毫秒(也可选)。最后, 时区由符号(-)或(+)后的数字表示落后或领先格林尼治标准时间(GMT)多少小时。例如, “-05:00”表示, 这是东部标准时区, 比GMT落后5小时。

网页超文本技术工作小组(WHATWG)HTML规约还有一个pubdate属性, 用来表示一个文档的发布时间, 就像下面的例子。在写本书时, 其中的pubdate属性还没有包含在W3C HTML 5规约中, 但是如果使用广泛, 以后应该会纳入规约。

Written by Jennifer Robbins (<time datetime="2012-09-01T 20:00-05:00"
pubdate>September 1, 2012, 8pm EST</time>)

WHATWG还有一个可以帮助计算机理解内容的数据元素, 它可以用于各种类型的数据, 包括日期、时间、度量、重量等。它使用value属性提供机器可读的信息。下面是几个例子:



特别提示: Internet Explorer 8及其早期版本不支持mark元素(见本章前面的侧栏“Internet Explorer对HTML 5的支持”)。旧版本的Firefox和Safari(分别早于3.6版和4版)没有根据规范提供支持, 但允许你对它应用样式。

<time>...</time>

时间数据

注意: time元素并不是用来标记那些无法精确获取的“时间”的, 如“去年结束时”或者“世纪之交”。

图5-14: 标准日期时间格式

例如: 太平洋标准时间
2012年12月25日 下午3点可以
表示为2012-12-25T15:00-8:00


```
<data>...</data>
```

机器可读数据

特别提示：在写本书时，time和data还都是新元素，没有得到普遍的支持。然而，除了IE 8和之前的版本之外，可以对它们使用样式，这样浏览器就可以识别。

```
<ins>...</ins>
```

插入文本

```
<del>...</del>
```

删除文本

```
<data value="12">Twelve</data>
<data value="2011-11-12">Last Saturday</data>
```

我不打算在此展开阐述data元素的更多细节，因为在写本书的时候，关于这个元素的讨论还在进行中，而且W3C也没有将其纳入HTML 5规约。而且作为一个初学者，你几乎没必要处理机器可读的数据。但是看看标记是如何为你的同伴或者为计算机程序和脚本提供有用的信息的，未尝不是一件有趣的事情。

插入和删除文本

ins和del元素用来标记一个文档的编辑部分，即那些插入或者删除的部分。这两个元素依靠样式规则来显示（即，默认情况下没有可依赖的浏览器）。ins和del元素都可以包含内联或者块元素，这取决于它们包含的是什么样的内容。

```
Chief Executive Officer: <del title="retired">Peter Pan</del><ins>Pippi Longstockings</ins>
```

添加空行

换行符

```
<br>
换行
```

有时候，你可能需要在流文本中使用换行符。前面已经介绍过浏览器如何忽视源代码中的换行符，所以需要专门的指令来告诉浏览器“在这里添加一个换行符”。

内联的换行元素（br）就是这个用途。br元素可以用来在地址或者诗中换行。它是一个空元素，也就是说，它没有任何内容。只需要在流文本中，你想添加换行的地方加上br元素即可（XHTML中是
），如下例所示，效果见图5-15。

```
<p>So much depends <br>upon <br><br>a red wheel <br>barrow</p>
```

So much depends
upon

a red wheel
barrow

遗憾的是，br元素很容易混淆（可以参看侧边的“警告”）。使用CSS white-space属性（第12章中会介绍）来给源代码添加换行符应该是一个较好的替代方案。

图5-15：每个br元素处都插入了一个换行符

调整非西方语言

由于网页是“世界范围的”，所以有一些元素用来满足非西方语言的需求。

改变方向

bdo（双向重写）元素允许将一个从右向左读的短语（**rtl**）来显示（如希伯来语或阿拉伯语）显示在一个“从左往右”读的流文本中，反之亦可。

```
This is how you write Shalom: <bdo dir="rtl">&#x05E9;&#x05DC;&#x05D5;&#x05DD;</bdo>
```

bdi（双向隔离）元素也类似，它可以隔离读取方向不同的文本，例如由用户添加的名字或评论。

东亚语言提示

HTML 5中提供了**ruby**、**rt**、**rp**元素来给东亚语言添加

注释。**Ruby**注释以小字显示在象形文字的上方，用于提供读音和翻译。在**ruby**元素中，**rt**元素表示有帮助的**ruby**文本。支持**ruby**文本的浏览器会以小一些的字体把它们显示在主文本的上方。对于不支持**ruby**的浏览器，可以把**ruby**文本插入圆括号中，每个圆括号都用**rp**元素括起来。不支持的浏览器就会把这些文本也显示在同一行上，并加上圆括号。支持的浏览器会忽视**rp**元素的内容，然后在上方显示**rt**文本。在写本书时候，浏览器对**Ruby**系统的支持还不一致。

```
<ruby>
  字 <rp>(</rp><rt>han</rt><rp>)<rp>
  汉<rp>(</rp><rt>zi</rt><rp>)<rp>
</ruby>
```

这个例子来自HTML 5工作草案（whatwg.com），按照MIT版权许可使用。

单词换行

<wbr>

单词换行

单词换行元素（**wbr**）标记一个单词合适的换行位置（标准里说的是“单词换行时机”）。它可以避免浏览器错误地换行，允许创作者为单词指定合适的换行位置。单词在需要的时候才会在**wbr**元素指定的位置换行（如图5-16所示）。如果还有足够的空间，单词就不会换行。浏览器支持这个元素已经很久了，但是才刚刚被标准收录。

```
<p>The biggest word you've ever heard and this is how it goes:
<em>supercali<wbr>fragilistic<wbr>expialidocious</em>!</p>
```

The biggest word you've ever heard and
this is how it goes: *supercalifragilistic*
expialidocious!

图5-16：当一行没有足够的空间来显示一个单词时，就会在**wbr**元素指定的位置换行

警告：不应该使用**br**强制换行来实现一个列表的效果。比如，不要这么做：

```
<p>Times<br>
Georgia<br>
Garamond
</p>
```

如果这是一个列表，应该通过样式表关闭项目符号，使用语义正确的无序列表元素。

```
<ul>
  <li>Times</li>
  <li>Georgia</li>
  <li>Garamond</li>
</ul>
```


练习5-2 识别内联元素

下面是Black Goose Bistro博客的一篇文章，可以利用它来试试不同的内联元素。你是否能正确地找到使用下面这些元素的地方：

b br cite dfn em i q small time

由于标记往往是主观的，你的标记可能和附录A中的例子并不完全一致，但是这些元素在这篇文章都有机会使用。另外，有一个短语应该用两个元素来标记（请注意嵌套的方式）。

你可以在本页直接填写标签。如果你想使用文本编辑器，并且在浏览器中查看结果，可以从www.learningwebdesign.com/4e/materials下载文本文件。附录A中有参考答案。

<article>

<header>

<p>posted by BGB, November 15, 2012</p>

</header>

<h1>Low and Slow</h1>

<p>This week I am extremely excited about a new cooking technique called sous vide. In sous vide cooking, you submerge the food (usually vacuum-sealed in plastic) into a water bath that is precisely set to the target temperature you want the food to be cooked to. In his book, *Cooking for Geeks*, Jeff Potter describes it as ultra-low-temperature poaching.</p>

<p>Next month, we will be serving Sous Vide Salmon with Dill Hollandaise. To reserve a seat at the chef table, contact us before November 30.</p>

<p>blackgoose@example.com

555-336-1800</p>

<p>Warning: Sous vide cooked salmon is not pasteurized. Avoid it if you are pregnant or have immunity issues.</p>

</article>

泛型元素 (div和span)

如果我们所谈论过的元素不能准确地描述内容，那么该怎么办？毕竟世界上有无数种类型的信息，但是你已经看到了，没有那么多表意元素。幸运的是，HTML提供了两种泛型元素，能够通过自定义来完美描述内容。div (division (区) 的简写) 元素表示内容的分隔，而span元素则表示当前没有文本级别元素适用的词和短语。可以用id或class特性为泛型元素命名，马上就会详细介绍。

div和span没有自身固有的表现层性质，但是只要你喜欢，就可以用样式表来格式化内容。事实上，泛型元素是基于标准的Web设计的首要工具，因为它们能使作者准确地描述内容，并为样式规则添加许多“钩子”。它们同时还能够允许页面上的元素被JavaScript访问和操纵。

我们将花一些时间来学习div和span (还有id和class特性)。下面介绍创作者如何使用它们来结构化内容。

使用div分区

div元素用来创建页面元素或者内容的逻辑分组。它表明它们同属于某种概念单元或应被CSS或JavaScript视为一个单位。通过把相关内容标记为一个div，并赋予它一个唯一的id标识，或者指明它是class的一部分，就可以对元素进行分组。下面介绍div元素的几个例子。

本例中，div元素用作窗口，将图像和两个段落组合到“listing”中。

```
<div class="listing">
  
  <p><cite>The Complete Manual of Typography</cite>, James Felici</p>
  <p>A combination of type history and examples of good and bad type
  design.</p>
</div>
```

通过将这些元素放入一个div中，可以清楚地表示：它们在概念上是相关的。这也能够定义“listing”中的p元素，跟网页中其他的段落区分开。

下面是div的另一个常见用法，为了上下文、结构和布局的需要，将网页分割成几个部分。本例中，把一个标题和几个段落放入一个div中，并标识为“news”部分。

```
<div id="news">
  <h1>New This Week</h1>
  <p>We've been working on...</p>
  <p>And last but not least,... </p>
</div>
```

既然有名为“news”的元素，那我就可以使用样式表，将其作为一栏置

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

泛型块级元素

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

泛型内联元素

标记提示

可以将div元素嵌套在其他div元素中，但不要走极端。我们应该总是努力地保持标记尽可能简单，所以只有当逻辑结构、样式或脚本需要时，才应该添加div元素。

于网页的右边或左边。你可能会想到，难道不能用一个section元素来这么做吗？当然可以，因为我们在HTML 5中已经有了更好的语义分组元素，事实上，开发者现在很少用泛型元素div了。

内联span

span与div元素有一样的好处，只不过它用于不引入换行的内联元素。因为span是内联元素，所以只能包含文本和其他内联元素（换句话说，不能将标题、列表、内容组织元素放入span）。下面介绍一些例子。

虽然没有telephone元素，但是能够用span来赋予其电话号码的意义。本例中，每个电话号码都标记为span，并归类为“tel”。

```
<ul>
  <li>John: <span class="tel">999.8282</span></li>
  <li>Paul: <span class="tel">888.4889</span></li>
  <li>George: <span class="tel">888.1628</span></li>
  <li>Ringo: <span class="tel">999.3220</span></li>
</ul>
```

你可以看到，加了标签的span给其中的文本添加了意义，否则，这些文本只是数字组成的随机字符串。而且span元素允许我们在整个站点对电话号码应用相同的样式（比如，可以使用CSS white-space:nowrap声明来确保不出现换行）。这样信息不仅能为人所识别，而且（理论上）也能为计算机程序识别，程序知道如何处理“tel”类的信息。事实上，一些值（包括“tel”）已经在名为微格式的标记系统中标准化了，这也使得网页内容对于软件更有用（见侧栏微格式和元数据）。

id和class值

id和class的特性值应该以字母（A~Z或a~z）或下划线开头（Internet Explorer 6及其更早版本对下划线支持有问题，所以通常也要避免使用下划线）。不能包含字符空格和特殊字符。字母、数字、连字号、下划线、冒号和句点都是可用的。同时，特性值是区分大小写的，所以“sectionB”不能与“Sectionb”互换。

id和class特性

在上个例子中，看到了元素标识符：id和class，它们用于给泛型div和span元素命名。然而，每个标识符都有特殊的用途，知道它们之间的差别很重要。

id标识符

id特性用于给文档中的元素指派一个唯一的标识符。换句话说，id的值在文档中必须只能使用一次。这样能使得为某个元素分派名称显得很有用，就像它是数据的一部分。关于提供id特性名称的信息，请参考侧栏“id和class值”。

本例使用书的ISBN来唯一标识每个清单。两个书籍清单不能共用一个id。

```
<div id="ISBN0321127307">
  
```

```

<p><cite>The Complete Manual of Typography</cite>, James Felici</p>
<p>A combination of type history and examples of good and bad type.
</p>
</div>

<div id="ISBN0881792063">
  
  <p><cite>The Elements of Typographic Style</cite>, Robert Bringhurst
  </p>
  <p>This lovely, well-written book is concerned foremost
  with creating beautiful typography.</p>
</div>

```

Web作者也将id用于识别网页的不同部分。使用这种方法，就不可能在文档中有多个“main”、“links”、“news”或其他相同名称的id。

```

<section id="main">
  <!-- main content elements here -->
</section>

<section id="news">
  <!-- news items here -->
</section>

<aside id="links">
  <!-- list of links here -->
</aside>

```

不只用于div

id和class特性可以用于几乎所有的HTML 5元素中，而不仅限于div和span。例如，可以将一个有序列表标识为“directions”，而不用将其装入一个div中。

```

<ul id="directions">
  <li>...</li>
  <li>...</li>
  <li>...</li>
</ul>

```

在HTML 4.01中，id和class特性可用于除了base、basefont、head、html、meta、param、script、style和title之外的所有元素。

微格式和元数据

正如你所看到的，HTML中的元素不足以描述所有类型的信息。一些开发人员认为，如果class名称可以标准化（例如，对电话号码信息始终使用“tel”），那么就可以建立系统来描述数据，从而使其更加有用。这个系统称为微格式。微格式可以通过为id、class和rel属性建立标准值来扩展HTML标记的语义，而不需要创造全新元素。

有一些微格式的“词汇表”可以用于识别内容，如联系信息（hCard）或日历项目（hCalendar）。网站Microformats.org是了解微格式的好地方。为了给你总体思路，下面的例子使用hCalendar微格式词汇来描述某件事，这样浏览器就可以自动将其添加到日历程序中。

```

<section class="vevent">
  <span class="summary">O'Reilly Emerging
  Technology Conference</span>,
  <time class="dtstart" datetime="20110306">Mar 6
  </time> -
  <time class="dtend" datetime="20110310">10,
  2011</time>

```

```

<div class="location">Manchester Grand Hyatt,
San Diego, CA</div>
<a class="url" href="http://events.example.com
pub/e/403">Permalink</a>
</section>

```

hCard词汇可以识别典型的联系信息（以vCard格式存储）、包括地址（adr）、邮编（postal-code）、国家（region）和电话号码（tel）等。然后，浏览器可以使用一个服务从网页中抓取信息，并将其自动添加到地址簿。

关于微格式还有很多内容本书无法涵盖。但有必要提及的是，在W3C中还有两个额外的、更复杂的系统用于在网页中添加元数据，即RDFa和Microdata。目前尚不清楚它们未来会如何，但我认为这个元数据目前可以先不考虑。但是如果你想了解更多，可以查看WebSitesMadeRight.com，那里有关于这三个选项的很多介绍材料和教程：websitesmaderight.com/2011/05/html5-microdata-microformats-and-rdfa-tutorials-and-resources/。

class标识符

class特性用于组合相似的元素；因此，不同于id特性，多个元素可以共用一个class名。使多个元素使用同一个class，从而可以使用一个样式表，一次将样式应用到所有定义了标签的元素中，或者可以使用一个脚本控制全部元素。让我从本书前面的例子中归类元素开始。在第一个例子中，我给每个div都添加class特性（如listing），并将它们归类为“descriptions”。

```
<div id="ISBN0321127307" class="listing">
  <header>
    
    <p><cite>The Complete Manual of Typography</cite>, James Felici</p>
  </header>
  <p class="description">A combination of type history and examples of
good and bad type.</p>
</div>

<div id="ISBN0881792063" class="listing">
  <header>
    
    <p><cite>The Elements of Typographic Style</cite>, Robert Bringhurst
</p>
  </header>
  <p class="description">This lovely, well-written book is concerned
foremost with creating beautiful typography.</p>
</div>
```

提示

id特性用于识别。

class特性用于归类。

注意，同一元素是如何同时拥有class和id标识符的。一个元素也可以同时属于多个class。当有多个class值时，可以用空格将它们分隔开。本例中，可以将div归类为“book”，从而与文档中其他地方的“cd”或“dvd”类的清单（listing）区分开。

```
<div id="ISBN0321127307" class="listing book">
  
  <p><cite>The Complete Manual of Typography</cite>, James Felici</p>
  <p class="description">A combination of type history and examples of
good and bad type.</p>
</div>

<div id="ISBN0881792063" class="listing book">
  
  <p><cite>The Elements of Typographic Style</cite>, Robert Bringhurst
</p>
  <p class="description">This lovely, well-written book is concerned
foremost with creating beautiful typography.</p>
</div>
```

这里应该很好地介绍了如何使用div和span元素与给文档提供意义和组织文档的class和id属性。在第三部分的样式表章节中，会更多地用到它们。

特殊字符

在继续向前学习之前，还有一个文本相关的主题。

常见的特殊字符，如版本标志©，不是标准ASCII字符集的一部分。标准字符集只包括字母、数字和一些基本字符。其他的字体，如小于号或者尖括号（<），都是不可用的，但是如果将它用于HTML文档，那么浏览器会解析为标记的开端。

源码中的这类字符必须转义（escape）。转义并不意味着输入字符本身，而用数字或已命名的字符引用（character reference）来表示它。当浏览器看到这些字符引用时，在网页显示时，会将这些地方替换为正确的字符。

有两种方法引用特殊字符：利用指派的数值（数字实体）；使用预定义的字符名简写（称为命名实体）。所有的字符引用都以“&”开头，以“;”结尾。

为了讲得更清楚，这里举个例子。我想给网页添加版权符号。典型的Mac键盘命令Option+G在我的文字处理程序中起作用，但是浏览器或者其他的软件无法正确理解这个命令。作为替代，必须将命名实体©（或等效数值©）用于符号出现的所有地方（见图5-17）。

```
<p>All content copyright &copy; 2012, Jennifer Robbins</p>
```

或

```
<p>All content copyright &#169; 2012, Jennifer Robbins</p>
```

HTML定义了上百种命名实体，它们是标记语言的一部分，也就是说，你不能创建自己的实体。表5-2列出了一些常用的字符引用。如果想看全部的，Web Standards Project在www.webstandards.org/learn/reference/charts/entities/网站上提供了字符引用的完整列表。

All content copyright © 2007, Jennifer Robbins

图5-17：当文档显示在浏览器上时，字符引用替换为特殊字符

注意：在XHTML中每个&都必须转义（特别是出现在特性值中时），从而不会被解析为字符实体的开头。例如：

```

```


非换行空格

一个需要了解的有趣字符就是非换行空格（ ）它的用途是放在两个单词之间以确保不会换行。举个例子，如果这样标记我的名字：

Jennifer Robbins

我能确保它们总是在一行里。

表5-2：常见特殊字符及其引用

字符	描述	命名	数值
	字符空格（非换行空格）	 	
&	和符号	&	&
'	撇号	'	'
<	小于号（用于显示网页中的标记）	<	<
>	大于号（用于显示网页中的标记）	>	>
©	版权	©	©
®	注册商标	®	®
™	商标	™	™
£	英镑	£	£
¥	元	¥	¥
€	欧元	€	€
—	短破折号	–	–
—	长破折号	—	—
‘	左单引号	&lquo;	‘
’	右单引号	&rquo;	’
“	左双引号	“	“
”	右双引号	”	”
·	项目符号	•	•
...	水平省略号	…	…

提示

将你的HTML源代码按层次来缩进，会便于以后查找和更新。

小结

到目前，你已经学习了如何标记元素，见过了给文本内容添加结构和意义的所有HTML元素。万事俱备，只欠实践。练习5-3是个机会，你可以尝试目前学过的所有东西：文档结构元素、块元素、内联元素、分段元素和字符实体。祝你玩得开心！

练习5-3 Black Goose博客页面

既然已经给你介绍了所有的文本元素，那么你应该能用它们来标记Black Goose Bistro站点的博客页面了。原始文本如下（第二篇文章在练习5-2中已经使用内联元素标记好了）。你可以直接在本页编写，也可以下载这个纯文本文件（www.learningwebdesign.com/4e/materials）。附录A中有标记结果，*materials*目录中也有。

有了这个文本文件后，按照列出的指令来修改吧。修改结果如图5-18所示。

The Black Goose Blog

Home
Menu
Blog
Contact

Summer Menu Items

posted by BGB, June 15, 2013

Our chef has been busy putting together the perfect menu for the summer months. Stop by to try these appetizers and main courses while the days are still long.

Appetizers

Black bean purses

Spicy black bean and a blend of mexican cheeses wrapped in sheets of phyllo and baked until golden. \$3.95

Southwestern napoleons with lump crab -- new item!

Layers of light lump crab meat, bean and corn salsa, and our handmade flour tortillas. \$7.95

Main courses

Shrimp sate kebabs with peanut sauce

Skewers of shrimp marinated in lemongrass, garlic, and fish sauce then grilled to perfection. Served with spicy peanut sauce and jasmine rice. \$12.95

Jerk rotisserie chicken with fried plantains -- new item!

Tender chicken slow-roasted on the rotisserie, flavored with spicy and fragrant jerk sauce and served with fried plantains and fresh mango. \$12.95

Low and Slow

posted by BGB, November 15, 2012

<p>This week I am extremely excited about a new cooking technique called <dfn><i>sous vide</i></dfn>. In <i>sous vide</i> cooking, you submerge the food (usually vacuum-sealed in plastic) into a water bath that is precisely set to the target temperature of the food. In his book, <cite>Cooking for Geeks</cite>, Jeff Potter describes it as <q>ultra-low-temperature poaching</q>.</p>

<p>Next month, we will be serving Sous Vide Salmon with Dill Hollandaise. To reserve a seat at the chef table, contact us before November 30.</p>

Location: Baker's Corner, Seekonk, MA

Hours: Tuesday to Saturday, 11am to midnight

All content copyright © 2012, Black Goose Bistro and Jennifer Robbins



图5-18: 完成后的菜单网页

1. 首先，输入文档结构元素（html、head、meta、title和body）。给文档赋予标题“Black Goose Bistro: Blog”。

2. 接下来将做的第一件事是将最高级的标题和链接列表使用header元素标记成文档的页眉（不要忘记结束标记）。在页眉里，标题应该是h1，链接列表使用无序列表（ul）。对于如何让列表项目链接化，我们将在第6章讨论。通过将它指定为站点首要导航（nav元素），可以赋予列表更多的含义。
 3. 博客页面有两篇名为“Summer Menu Items”和“Low and Slow”的文章。将任意一个标记为article。
 4. 现在，第一篇文章已经成型了。让我们为这篇文章创建一个页眉，给文章设置标题（由于我们已经深入了文档的层次，这次应该使用h2），并且添加出版信息（p）。使用time元素为这篇文章标识出版日期，就像练习5-2中那样。
 5. 页眉后的内容是一个很清晰的段落。然而菜单中有一些很有趣的东西。它分成了两个段（开胃菜和主菜），所以要使用section元素进行标记。注意，</section>出现在</article>前，确保元素正确嵌套，不要交叉。最后，使用id特性来标记分段section。将第一个分段命名为“appetizers”，第二个分段命名为“maincourses”。
 6. 分段完成后，就可以标记内容了。在每个分段中，使用h3来标记标题。选取最合适的列表元素来描述菜单项和该项定义。在列表中标记每一项。
 7. 现在可以添加一些细节。使用span元素将每个价格归类为“price”。
 8. 有两个新菜是新项。将两个连字符改变为em字符，将“new item!”标记为非常重要。将每个新菜的标题归类为“newitem”（提示，使用已有的dt元素；这次没必要使用span）。这次你就可以用“newitem”来定位菜单题目，并且使用不同于其他菜单项的样式。
 9. 第一篇文章就差不多处理完了。练习15-2已经基本完成了第二篇文章，在这里你需要给页眉添加适当的标题和出版信息。
 10. 现在看起来不错吧？现在给页面添加footer。将页脚中的每行内容都标记为一个段落（paragraph）。
 11. 在div中添加名为“about”的位置和小时信息。将位置“Location”和小时“Hours”单独显示在一行上。如果你愿意，也可以使用time元素来标记小时信息。
 12. 最后，将版权信息以小字体显示在文档上，并且在“copyright”后加上一个版权符号。
- 保存文件，并命名为**bistro_blog.html**，并在浏览器中检查网页（注意，IE 8及之前版本不支持HTML 5的分段元素）。你做得怎么样呢？

标记提示

- 根据最符合所选文本的意思选择元素。
- 不要忘了使用结束标记来关闭元素。
- 将所有的特性值放入引号内。
- 在添加相同的标记到多个元素时，不要忘了“复制和粘贴”这个好帮手。在整个文档中粘贴前，确保复制内容是正确的。

自我测验

注意啦！下面是一些快速回答的问题。

1. 添加标记，使下面两段之间出现主题分隔符。

```
<p>People who know me know that I love to cook.</p>
```

```
<p>I' ve created this site to share some of my favorite recipes.</p>
```

2. blockquote元素和q元素的不同点是什么？

3. 当把元素输入源文档时，哪个元素可以原样显示空白符？
4. ul元素和ol元素的不同点是什么？
5. 如何移除无序列表的项目符号？（通用方法，不针对特殊情况。）
6. 在文档中提供W3C的全称时，你将使用什么元素？你能写出完整的标记吗？
7. dl元素和dt元素的不同点是什么？
8. id元素和class元素的不同点是什么？
9. article元素和section元素有什么不同？
10. 写出由下列字符实体产生的字符：

— _____	& _____
 _____	© _____
• _____	™ _____

想要更多实践吗？

尝试标记自己的简历。如同在练习5-3中所做的，从纯文本开始，添加文档结构元素、块元素，然后是内联元素。如果没有找到正好匹配你的信息的元素，那就尝试使用div或span创建一个。

元素回顾：文本

下面是本章所学元素的小结。新的HTML 5元素会标记为“（5）”。在写本书时，data元素仅在WHATWG的HTML版本中。

页面分段	
addres	作者联系地址
article(5)	独立的内容
aside(5)	外围的内容（侧栏）
footer(5)	相关内容
header(5)	介绍内容
nav(5)	主要链接
section(5)	概念相关的内容组织（章节或段落）
标题内容	
h1...h6	标题，级别1到级别6
hgroup	标题组织
组织内容	
blockquote	块引用（长引用）
div	一般分隔
figure(5)	相关图像和资源
figcaption(5)	figure的文本描述
hr	段落级别的主题分隔符（水平分隔线）
p	段落
pre	预格式化文本
列表元素	
dd	定义
dl	定义列表
dt	术语
li	列表项（用于ul和ol）
ol	有序列表
ul	无序列表
分隔符	
br	换行
Wbr(5)	单词换行

短语元素（内联元素）	
abbr	缩写
b	引起注意（粗体）
bdi(5)	改变方向
bdo	双向重写
cite	引用
code	代码示例
data(WHATWG)	机器可读的等式
del	被删除的文本
dfn	定义项
em	强调文本
i	（改变语调）斜体
ins	被插入的文本
kbd	键盘输入的文本
mark(5)	高亮文本
q	短内联引用
ruby(5)	ruby注释（中文注音或字符）
rp(5)	ruby中使用插入语
rt(5)	ruby注释信息（定义字符的解释或发音）
s	带删除线；不正确的文本
samp	示例输出
small	注释，小字体
span	泛型文本区域
strong	特别强调
sub	下标
sup	上标
time(5)	机器可读的时间数据
u	引起注意，带下划线
var	变量

第6章 添加链接

如果你为Web创建了一个网页，那么就会想让它指向其他网页和资源，无论是你自己的站点还是别人的。毕竟，链接是Web的一切。本章中，我们将看到使链接生效的标记，包括指向站外、站内和本页内的标记。有一个元素使链接可用，它就是锚（anchor）（a）。

`<a>...`

锚元素（超文本链接）

要使选中的文本成为链接，可以简单地将选中的文本装进`<a>...` 开闭标签中，并使用`href`特性来提供URL地址。下面是创建指向O'Reilly Media网站的链接：

```
<a href="http://www.oreilly.com">Go to the O'Reilly Media site</a>
```

想要图像成为链接，只需要简单地将`img`元素放入锚元素中：

```
<a href="http://www.oreilly.com"></a>
```

几乎所有的浏览器将链接文本显示为蓝色并带下划线。一些过时的浏览器会在链接图片周围画蓝色边框，但是现在多数都不会这样。可访问的链接通常用紫色显示。用户可以在浏览器的`preferences`（首选项）中改变这些颜色，当然，也可以使用样式表来改变网站上链接的外观。具体做法见第13章。

警告： 箴言：如果你选择改变链接的颜色，那么最好整站保持一致，以免让用户迷惑。

当用户在链接文本或图像单击的时候，锚元素中指定的网页就会载入到浏览器窗口。前面展示的连接图像示例如图6-1所示。

从HTML 5开始，你可以把任何元素（甚至是块元素）放入另一个元素中！在HTML 4.01标准和之前的版本中，锚元素仅能为内联元素使用。

本章内容

- 创建到外部网页的链接
- 创建到自己服务器上文档的相对链接
- 链接到网页中的指定点
- 添加“mailto”链接和“tel”链接
- 以新窗口作为目标

快速浏览

锚语法

锚元素的简化结构：

```
<a href="url">链接文本  
或图像</a>
```

在HTML 5中，你可以把任何一个元素（甚至是块元素）放入另一个元素中。



图6-1：当用户单击链接文本和图像的时候，锚元素中指定的网页就会载入浏览器窗口

href特性

你需要告诉浏览器将链接到哪个文档，对吗？herf（hypertext reference，超文本引用）特性向浏览器提供网页地址（即URL）。URL必须放在引号中间。大多数时候，会指向其他的HTML文档；然而，也可以指向其他的Web资源，如图像、音频和视频文件。

因为周围内容与锚标签无关，所以真正的诀窍是保持链接的URL正确。

有两种指定URL的方法：

- 绝对URL提供文档的完整URL，包括协议（http://）、域名，如果需要还有路径名。当指定外面Web（并非在你自己的服务器上）的时候，需要使用绝对URL。

例子：`href="http://www.oreilly.com/"`

有时候，当链接网页的URL路径名很长的時候，链接结尾可能看起来令人相当困惑（图6-2）。但你只需要记住，它的结构依然是带有一个特性的简单容器元素。不要被路径名吓倒。

- 相对URL描述了链接网页相对于当前文档的路径名。当你链接到自己的站点（即相同服务器上）的另一个文档时，可以使用相对URL。它不需要协议和域名，仅仅需要文件路径。

例子：`href="recipes/index.html"`

本章中将使用绝对URL和相对URL，向我的烹饪网站Jen's kitchen（图6-3）添加链接。绝对链接很简单，我们首先使用它。

URL与URI

W3C和开发社区的关注点已经从URL（Uniform Resource Locator，统一资源定位符），转移到了更通用、技术上更准确的URI（Uniform Resource Identifier，统一资源标识符）。然而现在，在街上或工作中，你可能仍然喜欢的是“URL”。

这里简单介绍URL和URI的不同：URL是URI的一种类型，它是使用网络地址来作为资源标识符（URL的L表示地址）。另一种类型的URI是URN，URN是使用名字或者名字空间来作为资源标识符的（URN中的N就是名字空间）。

因为URL更被大家熟悉，所以在本章的讨论中我会一直使用“URL”。你需要知道的是URL是URI的一个子集，而且它们经常可以互换使用。

如果你想了解更多这样的知识。那我建议你看看Wikipedia中的URI，网址是en.wikipedia.org/wiki/Uniform_resource_identifier。

锚开始标签

```
<a href="http://www.amazon.com/s/?ie=UTF8&keywords=
bequet+caramel&tag=googhydr20&index=aps&hvadid=79790
39989&ref=pd_sl_1ah68hbamy_b">Bequet Caramels</a>
```

URL 链接文本 锚结束标签

图6-2：长URL的例子。虽然它可能使锚标签看起来令人迷惑，但结构还是没变

链接到Web上的网页

有许多次，你都想创建指向你在Web发现的网页。它被称为“外部”链接，因为它指向不在自己服务器或站点的网页。要创建外部链接，需要提供绝对URL，以http://开头（协议）。这就告诉浏览器，“到网上去，获取下列文档”。

我想在Jen's kitchen 主页添加一些外部链接（图6-3）。首先，我将列表项“The Food network”指向网站www.foodtv.com。添加锚标签（开始和结束标签），在锚元素中标记链接文本。注意我已经在列表项（li）中添加了锚标签。这是因为只有li元素可以嵌入ul元素中；在HTML中，将a元素直接放入ul元素中是非法的。

```
<li><a>The Food network</a></li>
```

接下来，将站点的完整URL添加到href特性中。

```
<li><a href="http://www.foodnetwork.com">The Food Network</a></li>
```

就这样，瞧，怎么样！现在“The Food network”将显示为链接，当用户单击时，会将用户带到那个站点。

练习6-1 创建外部链接

打开jenskitchen文件夹中的文件 index.html。将列表项“Epicurious”指向网页www.epicurious.com，如下面的例子所示。

```
<ul>
  <li><a href="http://www.foodnetwork.com/">The Food network</a>
</li>
  <li>epicurious</li>
</ul>
```

完成后，保存index.html并在浏览器中打开。如果你有Internet连接，那就可以单击新的链接，来到Epicurious站点。如果链接没有将你带到那里，返回查看，确保标记没有缺失。

标记提示

URL之争

如果你链接到一个很长URL的网页，将URL从浏览器的地址栏上复制下来，并粘贴到你的文档中，是有帮助的。因为这样可以避免漏输入字符，从而损坏整个链接。

试一试

制作Jen's Kitchen



图6-3：完成后的Jen's Kitchen网页

Jen's Kitchen网站的所有文件都可以从www.learningwebdesign.com/4e/materials在线获得。下载整个目录，确保没有改变内容组织方式。

所有练习的结果标记都在附录A中提供。

这个网页不是很好看，但它是一个让你提高链接技能的好机会。

注意：在PC或Mac中，文件都组织到“文件夹（folder）”中，但是在Web开发中，它通常称为等价但更技术性的“目录”。文件夹只是带有可爱图标的目录。

站内链接

你要创建的链接中，有一大部分是站内网页间的：从主页到各部分网页、从各部分网页到内容网页的链接等。这些情况下，可以使用相对URL——可以调用自己服务器内的网页。

如果没有“http://”，浏览器就会在当前服务器寻找链接文档。路径名，是指向具体文件和目录的符号，它告诉浏览器到哪里去找文件。Web路径名遵循Unix习惯，使用正斜线（/）隔开目录和文件名。相对路径名，描述了从当前文档的位置开始，如何获取链接文档。

相对路径名需要一点使用技巧。在我的教学生涯中，没有什么比编写相对路径名更能难倒初学者的了，所以让我们一步一个脚印地前进。我们学习过程中，将有很多我推荐做的练习，请认真对待。

本节中所有路径名的例子都是基于Jen's kitchen站点的结构，如图6-4所示。当你用图表表示网站目录结构时，它往往看起来像个倒置树，根目录在层次顶端。对于Jen's kitchen 站点，根目录名为jenskitchen。换一种方式来看，在Mac的Finder中，有目录及子目录的视图（Windows用户只能一次看到一个目录）。

路径名禁忌

当编写相对路径名时，你应该严格遵守下列规则，以避免常见错误：

- 不要使用反斜线（\）。Web URL路径名只使用正斜线（/）。
- 不要以盘符开头（D:、C:等）。虽然当它们都在你自己计算机上时，网页可以成功链接，但一旦上传到Web服务器，盘符名称就不合适了，还会破坏链接。
- 不要以file://开头。这还表示文件是本地的，当上传到服务器后，会导致链接错误。

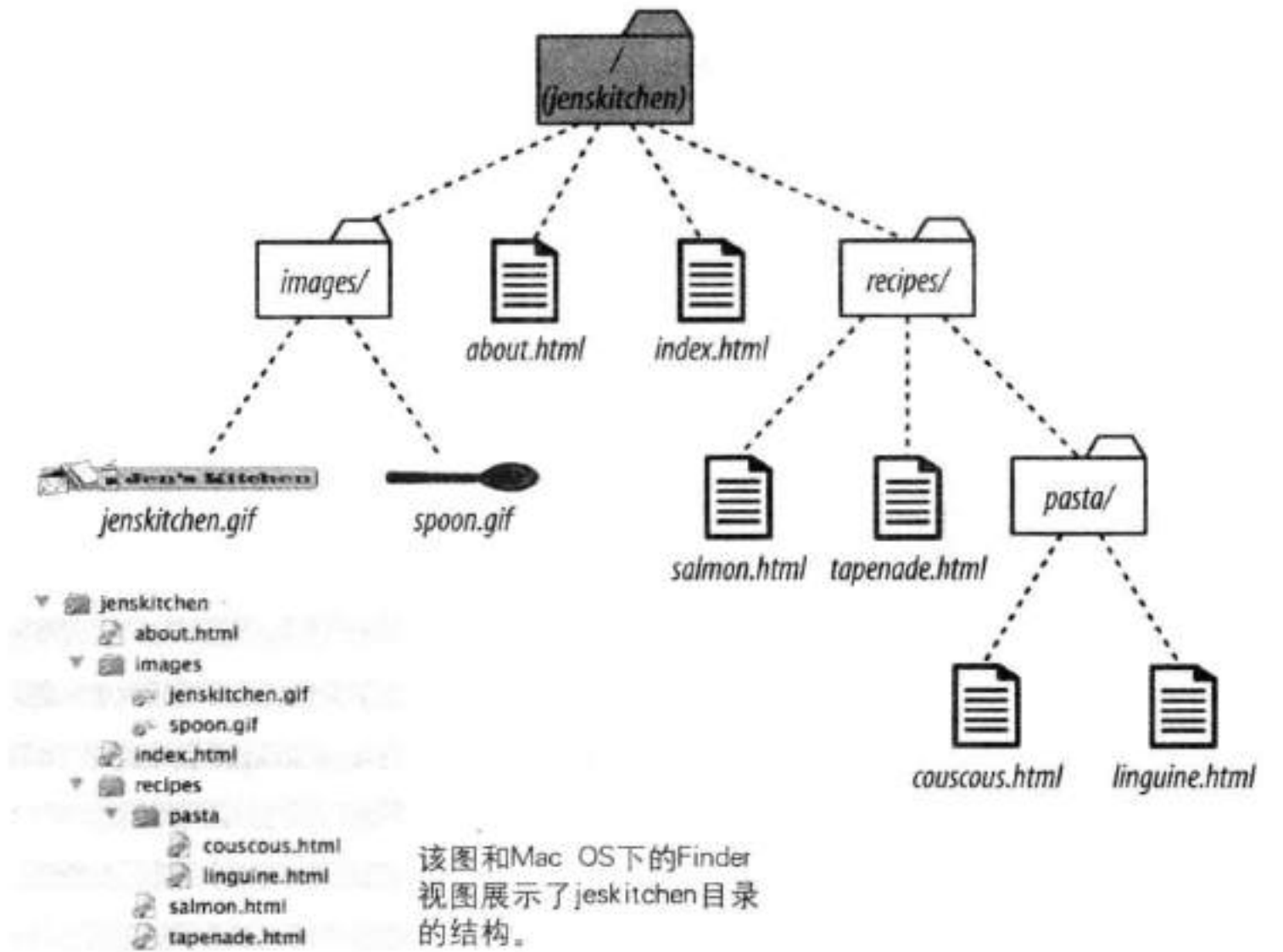


图6-4: jenskitchen网站结构图

目录内链接

最直接的相对URL，就是指向相同目录中的另一个文件。当链接到相同目录中的文件时，你只需要提供文件的名称（它的文件名）。当URL是一个文件名称时，服务器将在当前目录中查找（即当前链接文档所在的目录）相应文件。

本例中，我想创建一个从主页（*index.html*）到综合信息网页（*about.html*）的链接。两个文件在同一目录（*jenskitchen*）中。所以从主页，简单地在URL中提供文件名，就可以创建指向信息页的链接（图6-5）：

```
<a href="about.html">About the site...</a>
```

指向只有文件名的链接，说明链接文件与当前文档在同一目录中。

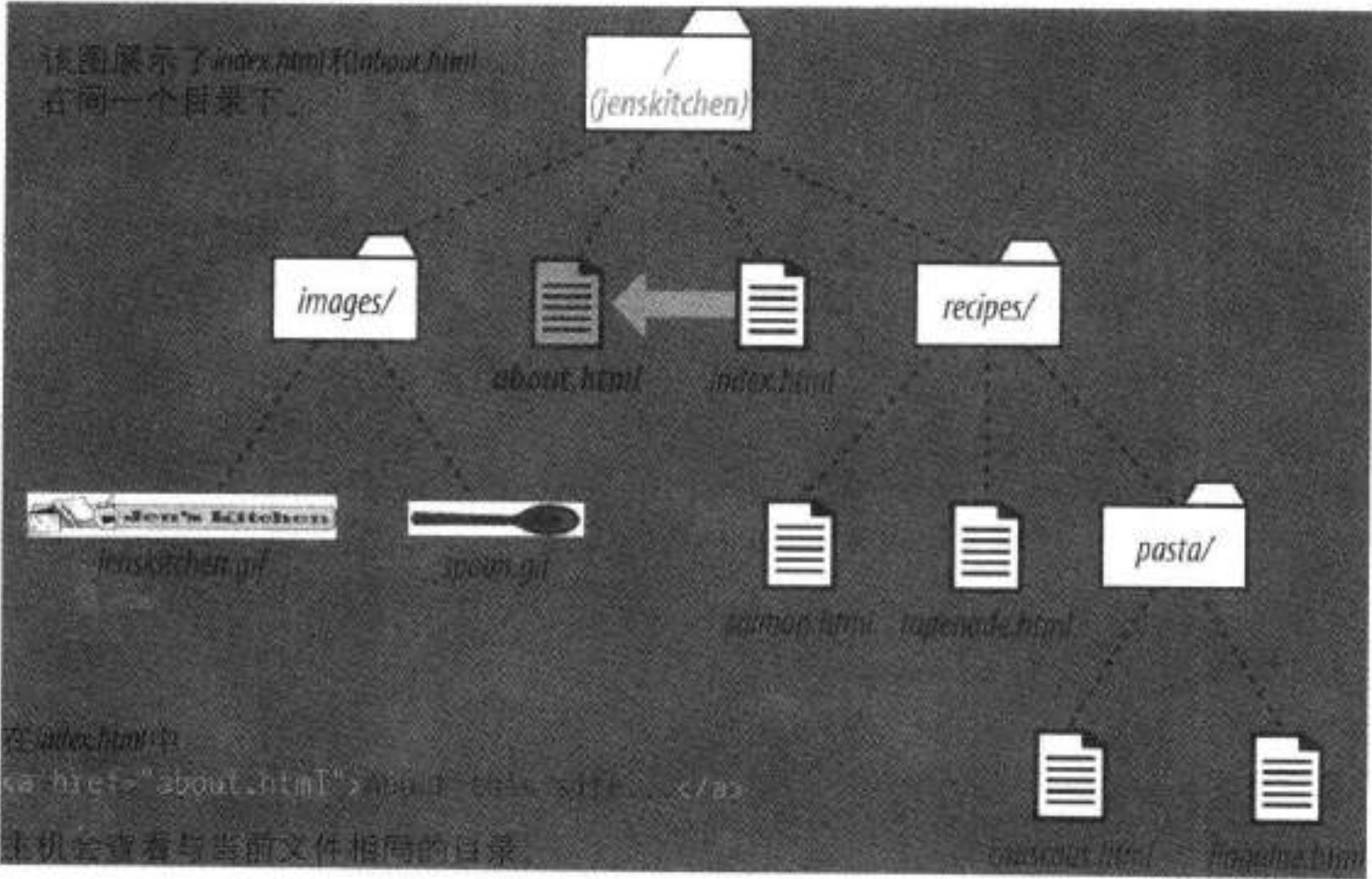


图6-5：编写指向相同目录中其他文档的相对URL

练习6-2 相同目录中的链接

打开 *jenskitchen* 文件夹中的文件 *about.html*。给网页底部的段落“Back to the home page”添加指向主页 *index.html* 的链接。记住锚元素一定要包含在 *p* 元素中。

```
<p>Back to the home page</p>
```

完成以后，保存 *about.html* 并在浏览器中打开。不需要Internet连接，可以在本地测试（也就是说，在你自己的计算机中）。单击这个链接，你就可以回到主页。

链接到低层目录

但是如果文件不在同一目录中怎么办？你需要通过将路径名包含在URL中来告诉浏览器。让我们来看看其工作机制。

回到例子，recipe（配料）文件都存在名为“recipes”的子目录中。我想创建一个链接，从index.html指向recipes目录中salmon.html文件。URL中的路径名告诉浏览器，在当前目录中寻找名为recipes的目录，然后查找文件salmon.html（图6-6）。

```
<li><a href="recipes/salmon.html">Garlic Salmon</a></li>
```

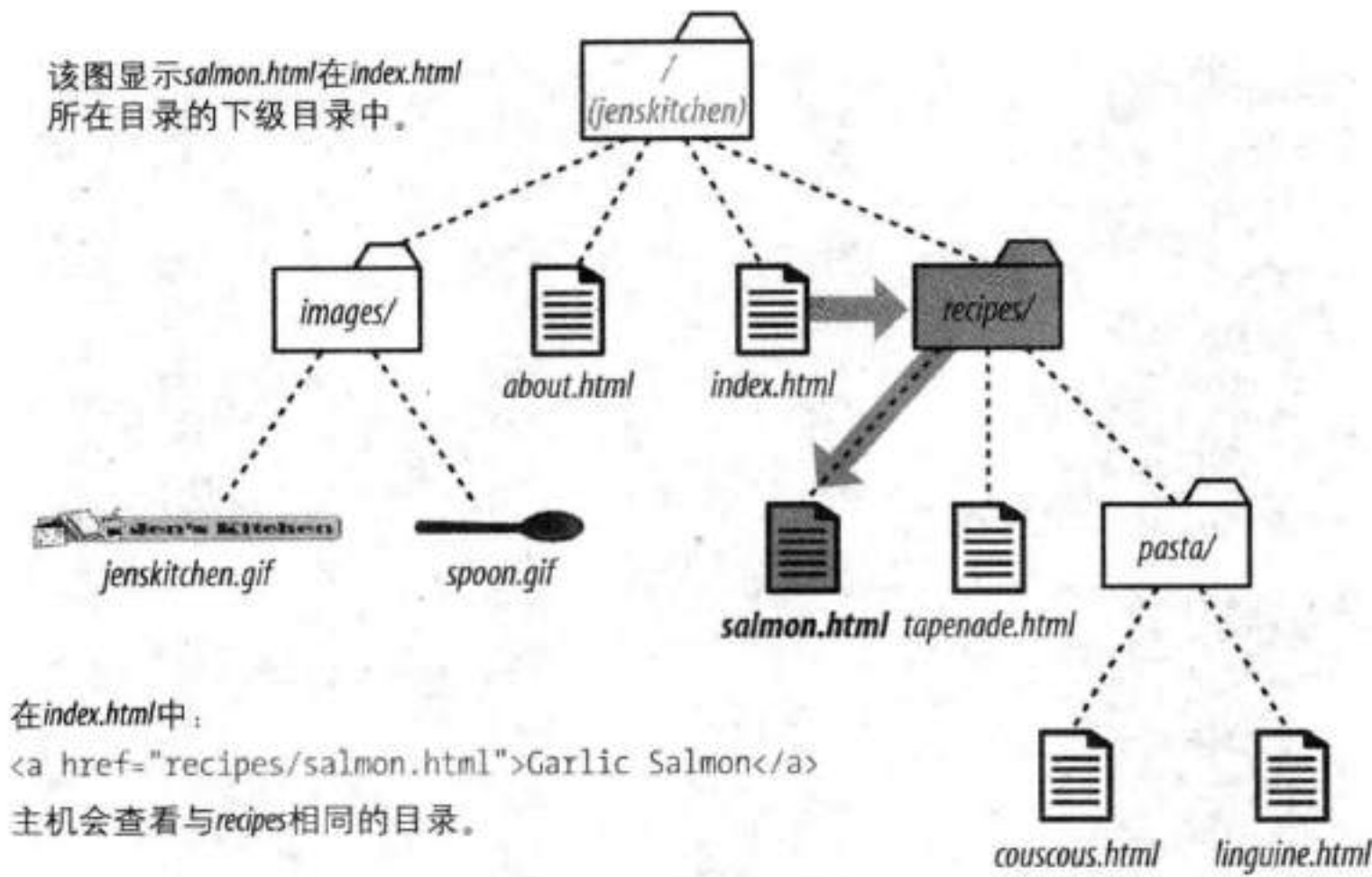


图6-6：编写相对URL，指向比当前文档目录层次低的文档

练习6-3 链接到低层目录

打开 jenskitchen 文件夹中的文件index.html。创建列表项“Tapenade（Olive Spread）”到recipes目录中文件tapenade.html的链接。记住正确嵌套元素。

```
<li>Tapenade (Olive Spread) </li>
```

完成以后，保存index.html并在浏览器中打开。你应该能单击新链接，并看到tapenade的配料网页。如果没有，请确保标记正确并且jenskitchen的目录结构需要与例子的一致。

现在链接到低层的`couscous.html`文件，它位于`pasta`子目录中。我们需要做的只是提供穿过两个子目录到达`couscous.html`的指令（图6-7）：

```
<li><a href="recipes/pasta/couscous.html">Couscous with Peas and Mint</a></li>
```

目录都用正斜线分开。锚标签会告诉浏览器，“在当前目录中寻找名为`recipes`的目录。在其中寻找另一个称为`pasta`的目录，在`pasta`目录中有想要链接到的文件`couscous.html`。”

现在我们已经完成了两个目录层级，那你应该知道路径名是如何组合起来的了。不管深入多少层级的目录，都可以使用相同的方法来提供路径名。我们只需要从当前文件相同位置的目录名开始，使用斜线接上每层目录名，直到发现链接文件名。

当链接到低层目录的文件时，路径名一定要包含到达目标文件所需经过的所有子目录的名称。

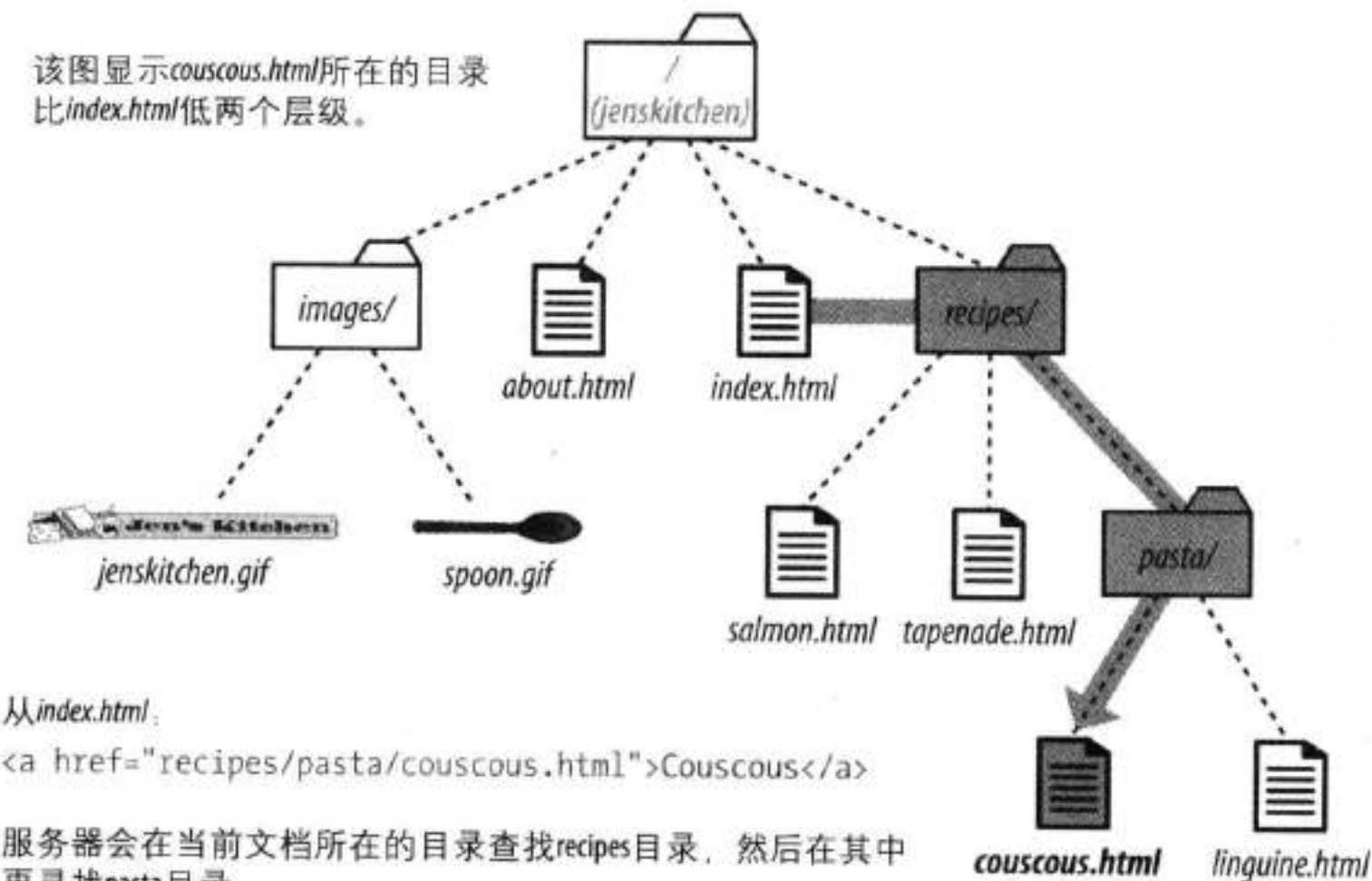


图6-7：编写相对URL，指向比当前文档低两个目录层级的文档

练习6-4 链接到低两层的目录

打开 `jenskitchen` 文件夹中的文件 `index.html`。创建列表项“Linguine with Clam Sauce”到 `pasta` 目录中文件 `linguine.html`（意大利面）的链接。

```
<li>Linguine with Clam Sauce</li>
```

完成以后，保存 `index.html` 并在浏览器中打开。单击新链接，可以得到美味的配料。

路径名中最开始的每个../告诉浏览器到上层目录去寻找文件。

链接到高层目录

目前，一切顺利，对吗？现在来讲一个技巧。这次我们的任务是，创建从salmon配料页到主页的链接，主页在上一层目录。

在Unix中，有一个专用的路径名约定“dot-dot-slash（点点斜）”（../）。当路径名以../开头时，等同于告诉浏览器“回到上一层目录”，然后再找具体文件的路径。如果你对浏览桌面计算机上的文件比较熟悉，那么这样理解比较有用：“../”的效果如同Windows Explorer（资源管理器）上的“Up（向上）”按钮，或Mac OS X中Finder中的向左箭头。

首先，创建从salmon.html回到主页（index.html）的链接。因为salmon.html位于recipes子目录，所以需要回到上一层jenskitchen目录，寻找index.html。这个路径名告诉浏览器“回到上层”，然后在目录中寻找index.html（图6-8）。

```
<p><a href="../../index.html">[Back to home page]</a></p>
```

注意，不需要在路径名中写出上层目录的名字（jenskitchen），可以用../来代替。

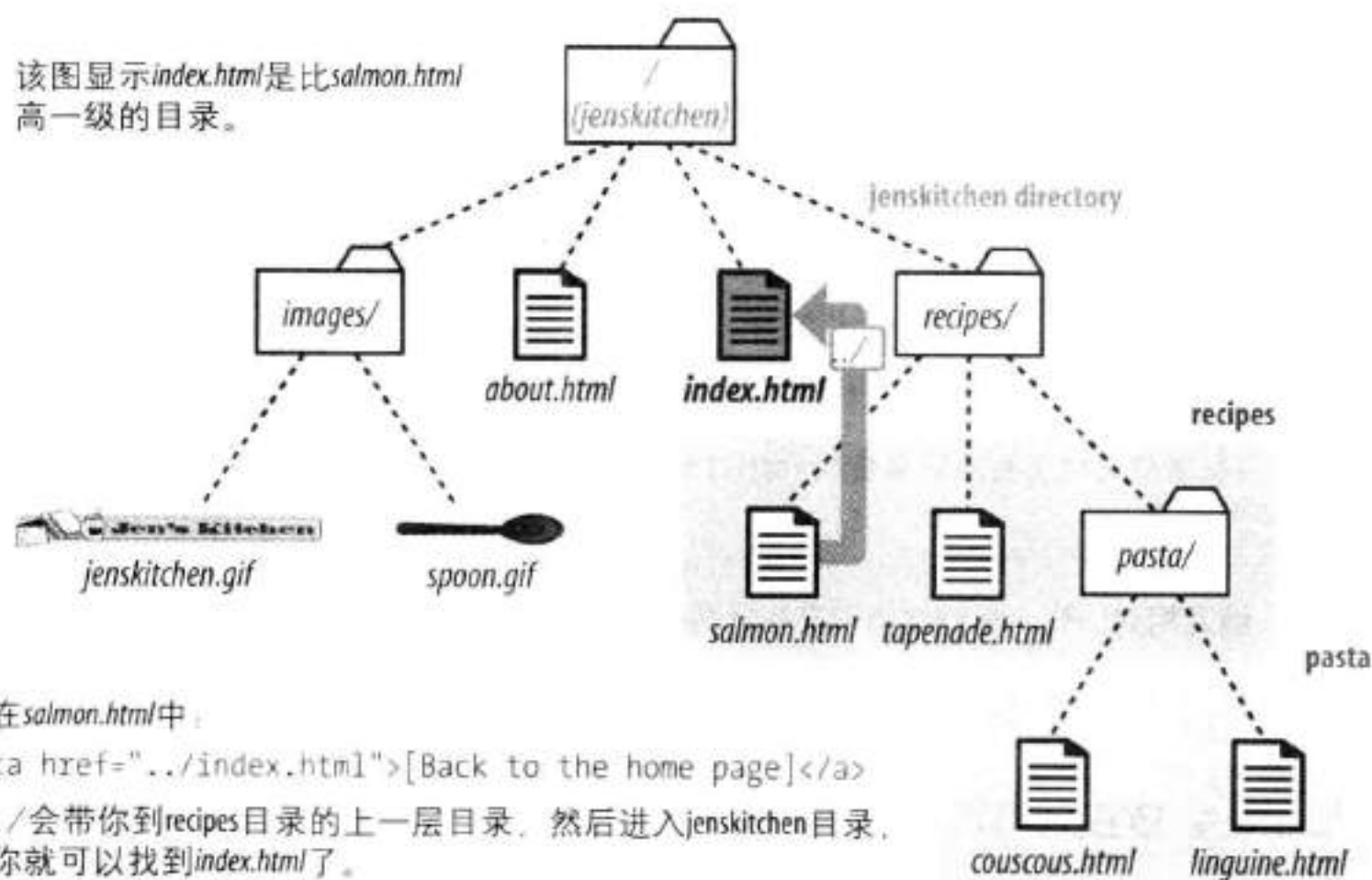


图6-8：编写相对URL，指向比当前文档高一层目录的文档

但是，链接从couscous.html回到主页，该怎么做呢？你能猜出如何跳过两层目录吗？很简单，只需要使用两次dot-dot-slash（图6-9）。

couscous.html网页上回到主页（index.html）的链接如下：

练习6-5 链接到高层目录

打开recipes目录中的tapenade.html文件。在网页底部，你可以看到下面的段落：

```
<p>[Back to the home page]</p>
```

使用本节描述过的符号，使这个文本链接到位于上一层目录的主页（index.html）。

`<p>[Back to home page]</p>`

第一个../回到*recipes*目录，第二个../回到顶层目录，*index.html*就在其中。还是老样子，不需要写出目录名称，使用../就够了。

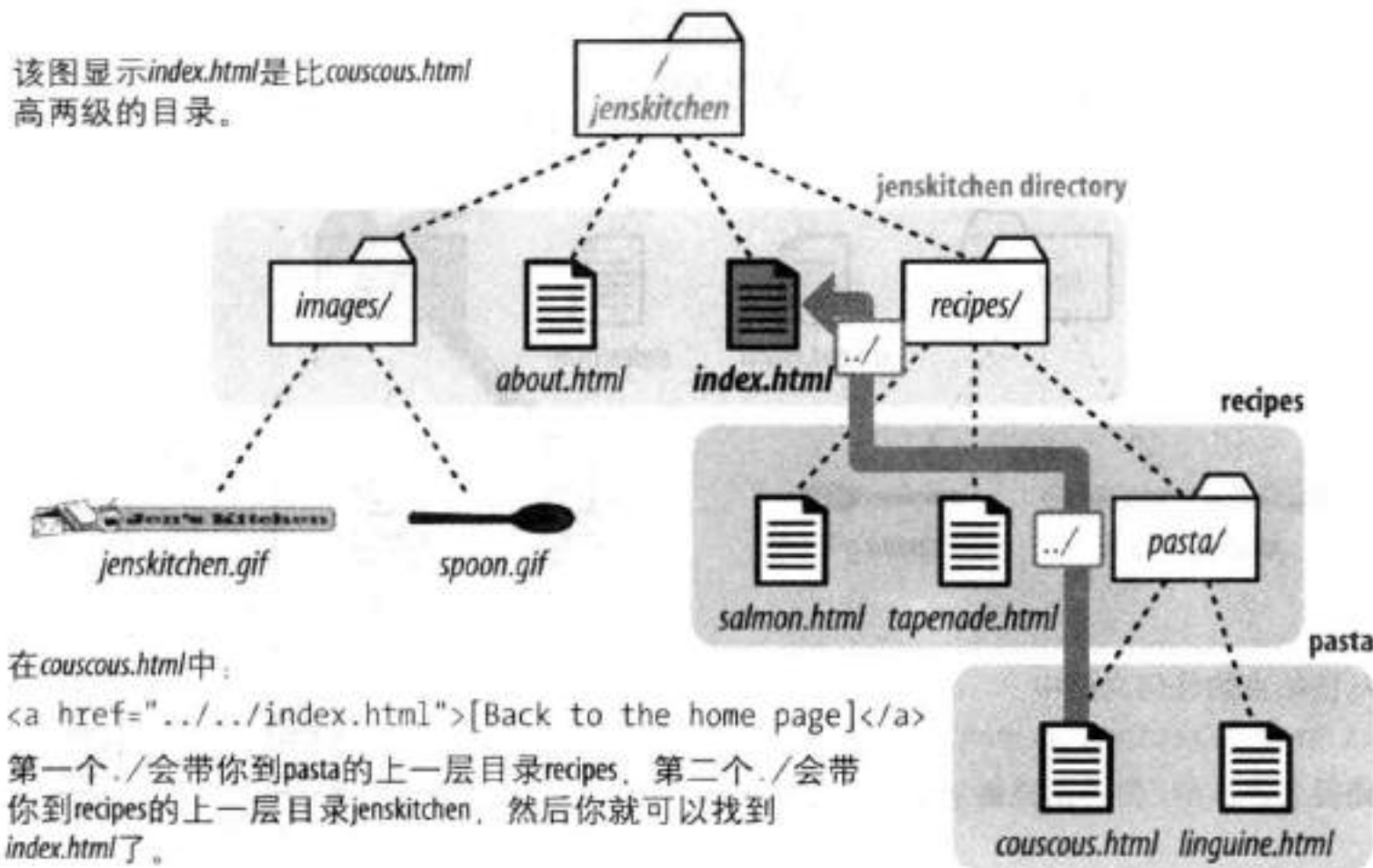


图6-9：编写相对URL，指向比当前文档高两层目录的文件

练习6-6 链接到向上两层目录

好，现在轮到你试一试了。打开文件*linguine.html*，将最后一段链接回主页，同我做的一样，使用../来完成。

`<p>[Back to the home page]</p>`

完成以后，保存文件并在浏览器中打开。现在应该可以链接到主页了。

站点根目录相对路径

所有的网站都有根目录，这个目录包含网站中所有的目录和文件。到目前，我们看到的所有路径名都与带链接的文档相关。编写路径名的另一个方法是从根目录开始，列出子目录的名称，直到发现你想链接的文件。这种路径名称为“站点根目录相对路径（site root relative）”。

在Unix路径名习惯中，在路径名开头使用正斜线表示根目录。下列链接中的相对根路径名告诉我们“到网站的顶级目录，打开*recipes*目录，然后找到*salmon.html*文件”（图6-10）。

`Garlic Salmon`

注意：我承认仍然有时候，在试图解析一个复杂的相对URL的时候，我为每个../默默地念道“上升一层，上升一层”。这样可以帮助我理清路径。

站点根目录相对路径因灵活性而备受欢迎。

注意，你不需要写出URL中的根目录名称（jenskitchen）——仅仅用正斜线（/）代表根目录并将浏览器带到顶级目录。从那里开始，指定浏览器将要寻找的目录。

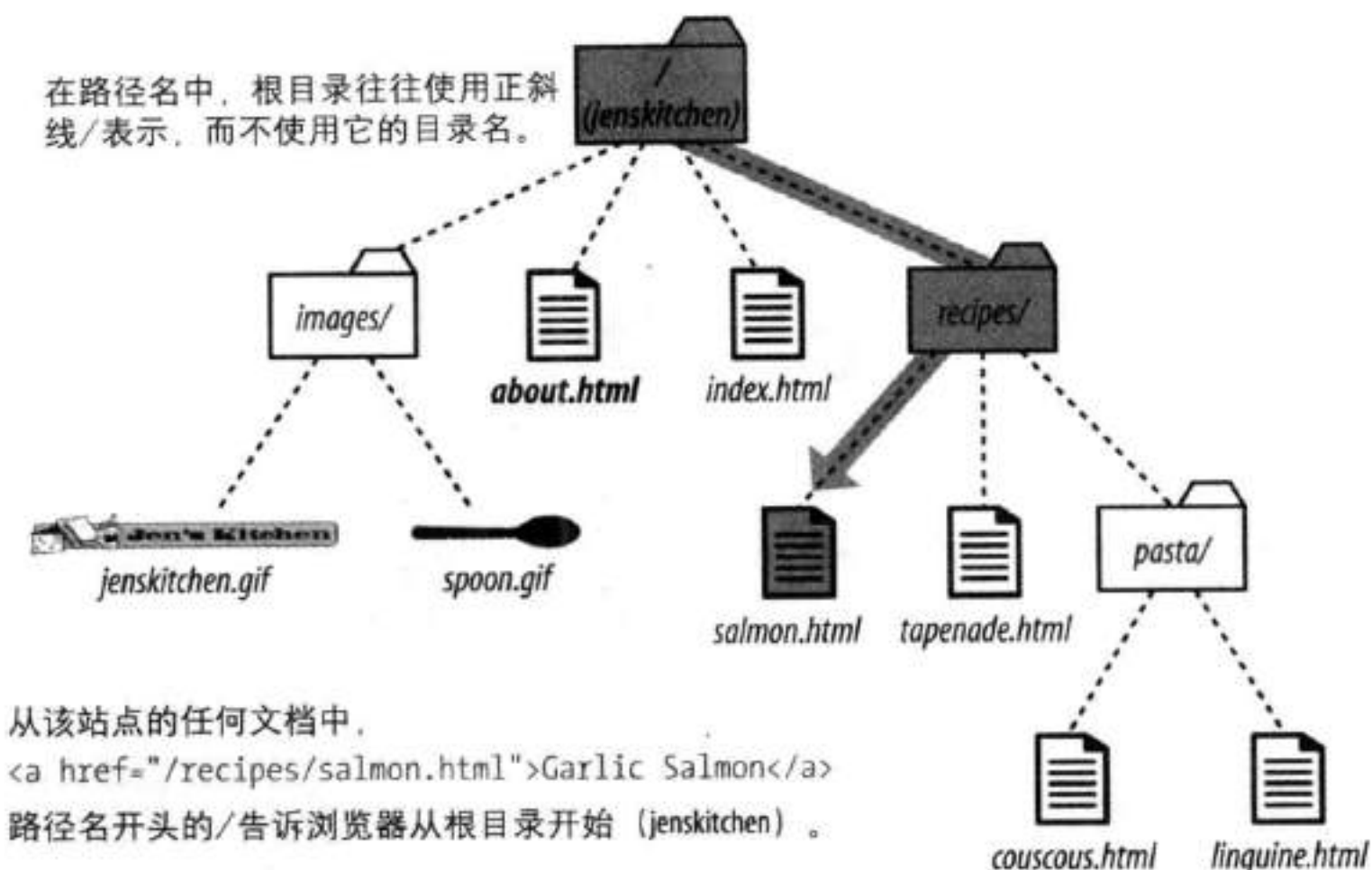


图6-10：编写相对URL，从根目录开始

因为链接是从根目录开始描述路径名的，所以它对服务器上所有的文档一视同仁，不管它位于哪个子目录。对于并不总在相同目录中的内容，或者动态生成的材料，站点根目录相对路径非常有用。它们也使得在文档间复制和粘贴链接变得容易。

然而，也有不好的一面，这些链接不能在本地机器上工作，因为它们也与硬盘相关。你必须等到站点放到最终服务器上，才能检查链接是否有效。

对于图像 情况相同

在指定URL时，img元素的src特性与锚中的href特性工作方式相同。由于你最可能使用自己服务器上的图像，所以图像元素的src特性通常设置为相对URL。

我们来看Jen's kitchen站点的一些例子。首先，为了在index.html中添加图像，标记如下：

```

```

这个URL的意思是“在当前目录（jenskitchen）中寻找image目录，然后在images目录中寻找jenskitchen.gif。”

工具可以给你一些帮助

如果你使用所见即所得（WYSIWYG）的创作工具来创建站点，工具会为你生成相对URL。像Adobe Dreamweaver和微软的Expression Web程序中都有内置的站点管理功能，即使你重新组织了目录结构，它们都可以调整你的相对URL。

现在来看 *piece de résistance*，添加图像到文件 *couscous.html*：

```

```

这比我们以前的所有例子都复杂一点。该路径名告诉浏览器，上升两级目录到顶级目录，之后进入 *images* 目录，寻找名为 *spoon.gif* 的图像。哟，这样也可以！

当然，你也可以用站点根目录相对路径的方式来简化，使用这种方法，*spoon.gif*（或 *images* 目录中的其他文件）的路径名可能这样得到：

```

```

这样付出的代价是，直到站点上传到服务器，才能看到图像的效果，但是一旦这样做了，维护就变得更加容易了。

练习6-7 再试一些

在继续学习之前，你可能想再多练习一些相对URL，以确保自己真正掌握。你可以直接在下面写出答案，或者如果你想测试标记是否有效，那么可以在真实文件中修改。你需要添加用作链接的文本到文件中（例如，第一个问题的“Go to the Tapenade recipe”）。答案见附录A。

1. 在 *salmon.html* 创建指向 *tapenade.html* 的链接。

Go to the Tapenade recipe

2. 在 *couscous.html* 创建指向 *salmon.html* 的链接。

Try this with Garlic Salmon.

3. 在 *tapenade.html* 创建指向 *linguine.html* 的链接。

Try the Linguine with Clam Sauce

4. 在 *linguine.html* 创建指向 *about.html* 的链接。

About Jen's Kitchen

5. 在 *tapenade.html* 创建指向 *www.allrecipes.com* 的链接。

Go to Allrecipes.com

注意：练习6-7中的所有路径名都可以是站点根目录相对路径，但是作为练习，使用相对URL来做吧。

链接到网页中的指定点

你知道可以链接到网页中的指定点吗？这对于信息在长的滚动网页的底部，或者一次单击就需要回到网页顶部的情况来说，会提供快捷方式。链接到网页的指定点，有时称为链接到文档片段（fragment）。

注意：链接到同一个网页的另一个位置在内容很长、需要滚动显示的网页中效果很好，但是在内容较短的网页中效果就不那么突出了。

创建建议

到顶部！

在实践中，当网页的内容很长时，往往需要创建一个链接来指向页面的顶部。这样就可以减少漫长的回滚了。

注意：记住，id值必须以一个字母或者下划线（虽然在一些版本的IE中，下划线可能会有问题）开始。

片段之前使用#号。

链接到网页中的特定点分为两个过程。首先，确定目的地，并设置到那里的链接。在下面的例子中，我在网页顶部创建了一个字母索引，链接到词汇表网页的各个字母部分（图6-11）。当用户单击字母“H”时，它们就跳到网页底部的“H”标题处。

第1步：命名目的地

我总喜欢将这一部分想象为：在文档中插入小旗，这样我可以很容易地回到那里。为了创建目的地，需要使用id特性给文档中的目标元素赋予唯一（unique）的名称（这里的“unique”是指在文档中只出现一次，而不是指独特和有趣）。用Web术语来讲，这个名称叫做片段标识符（fragment identifier）。

你可能记得第5章中的id特性，我们用它来命名div和span元素。这里，我们还是用它来命名元素，使它可以用作片段标识符，即链接的目的地。

下面是词汇表网页源码的例子。因为我想要用户直接链接到“H”标题，所以我给“H”标题添加id特性，并设置特性值为“startH”（图6-11①）。

```
<h1 id="startH">H</h1>
```

第2步：链接到目的地

使用适当的标识符，现在我可能创建到目的地的链接。

在网页顶部，我创建下到“startH”片段的链接②。对于任何链接，我都使用带有href特性的a元素，来提供链接的位置。为了标识我正链接到的片段，我在片段之前使用#号（也称作哈希或编号符号）。

```
<p>... F | G | <a href="#startH">H</a> | I | J ...</p>
```

这样就完成了。现在，当某人单击网页顶部列表中的“H”时，浏览器将跳转，并显示从“H”标题开始的部分③。

- ① 使用id特性标记目标位置。

```
<h2 id="startH">H</h2>
<dl>
<dt>hexadecimal</dt>
<dd>A base-16 numbering system that uses the characters 0-9 and
A-F. It is used in CSS and HTML for specifying color values</dd>
```

- ② 创建到目标位置的链接。名字前的#是必须的，它用来指明这是一个片段名称，而不是一个文件名。

```
<p>... | F | G | <a href="#startH">H</a> | I | J ...</p>
```

- ③

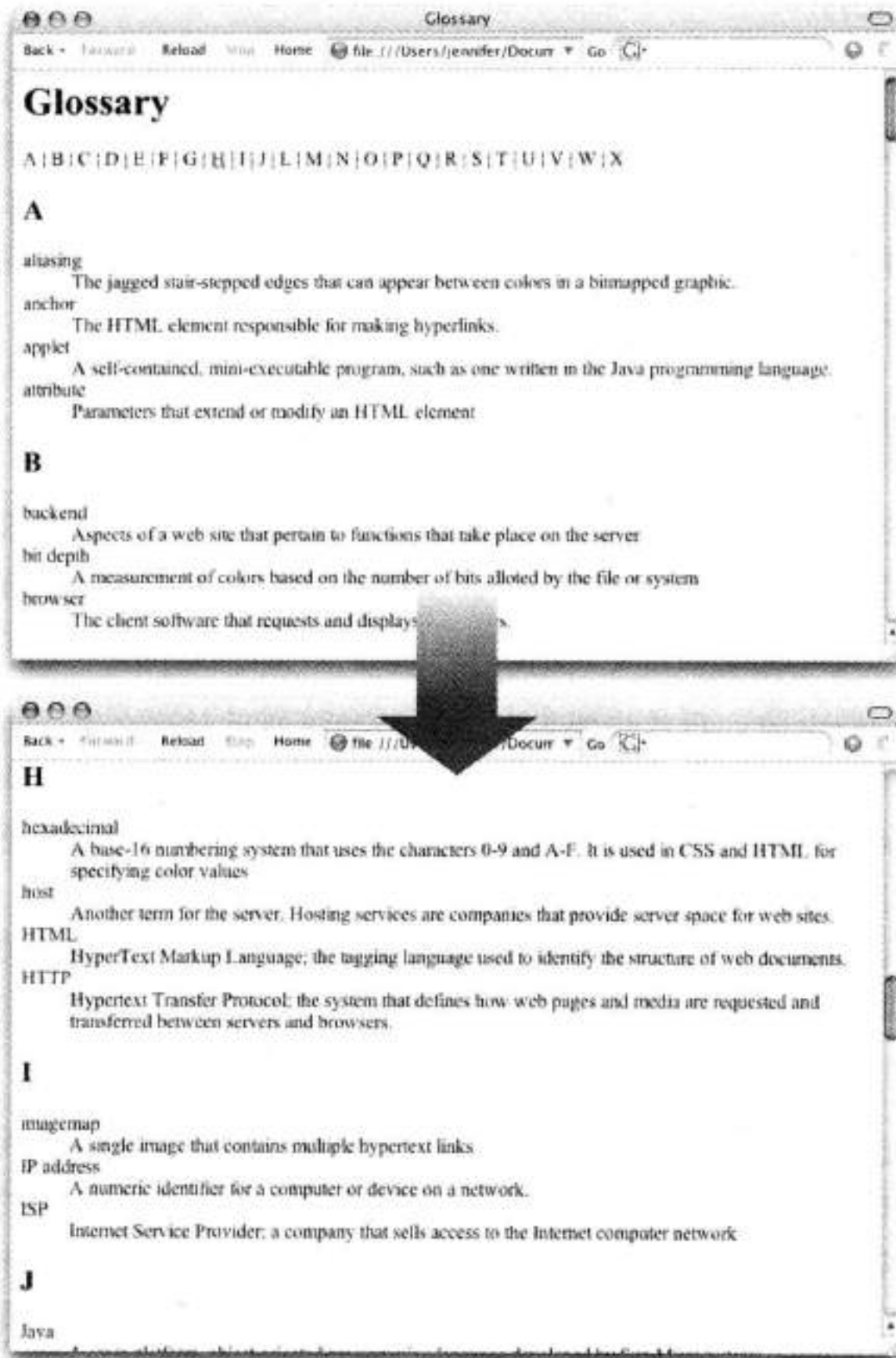


图6-11：链接到一个网页中的特定位置

注意：一些开发者为了帮助他们的兄弟姐妹，在任何专题开始的内容部分积极地加入id来作为锚。这样一来，其他人可以链接到内容的任何部分。

练习6-8 链接到片段

想进行一些链接到特定目的地的实践吗？打开materials文件夹中为本章准备的glossary.html文件。文档如图6-11。

1. 通过使用id特性将h2标题“A”命名为“startA”，将h2标题“A”确定为链接的目的地。

```
<h2 id="startA">A</h2>
```

2. 将网页顶部的字母“A”链接到定义片段。不要忘记#号。

```
<a href="#startA">A</a>
```

对网页顶部的每个字母不断重复步骤1和步骤2，直到你知道自己在做什么（或者直到不能再重复了）。你也可以使用户回到网页顶部。

3. 将标题“Glossary”设置成名为“top”的目的地。

```
<h1 id="top">Glossary</h1>
```

4. 在每个字母部分的最后，添加包含“TOP”的段落元素。使“TOP”成为指向刚才在网页顶部创建的标识符的链接。

```
<p><a href="#top">TOP</a></p>
```

把这段代码复制并粘贴到每个字母部分的最后。现在无论你的用户在文档的什么地方，他都可以很容易地回到网页顶部。

链接到另一文档的片段

在URL（绝对或相对）的结尾添加片段名称，就可以链接到另一文档中的片段。例如，要创建从词汇表网页所在目录的另一文档，到术语表网页“H”标题处的链接，那么其URL如下：

```
<a href="glossary.html#startH">See the Glossary, letter H</a>
```

甚至可以通过在绝对URL结尾放置片段标识符，链接到其他网站上的特定目的地，如下所示：

```
<a href="http://www.example.com/glossary.html#startH">See the Glossary, letter H</a>
```

当然，你不能控制别人网页中的命名片段。为了让你能到达，那些文档的作者必须插入目的点。要知道是否有目的点，以及在哪里，唯一的方法就是“View Source（查看网页的源码）”并在标记中寻找它们。如果外部文档中的片段移动或消失了，网页依然会加载，而浏览转向网页的顶部，如同常规链接一样。

以新浏览器窗口为目标

用户单击链接的时候，会遇上一个问题，他们不能返回。这个难题的传统解决方案是，使链接到的网页在新浏览器窗口中打开。这样，访问者可以看完后离开这个链接，而保留之前的网页中的内容。

在我提供指导之前，我首先要表明态度：我强烈反对这么做。首先，多标签页的浏览器使用户能够方便地找到返回原页面的方法。其次，打开新窗口的缺点是易用性问题。新窗口可能会对一些用户产生迷惑，特别是那些通过屏幕阅读器或其他辅助设备访问网站的用户。至少，新窗口会让他们觉得烦恼，而不是方便，特别是现在。因为通常将浏览器配置为阻止弹出窗口，所以有一定的风险，用户很可能会完全看不到新窗口中的内容。

所以，一定要仔细想好是否需要使用一个独立的窗口，如果你有非常好的理由，我会告诉你怎么做。在新浏览器窗口中打开新链接的方法，取决于你是否想控制窗口尺寸。如果窗口的尺寸无关紧要，你可以只用HTML。然而，如果你想以特殊像素尺寸打开新窗口，那你需要使用JavaScript。

使用标记打开新窗口

要使用HTML标记打开新窗口，可以在锚（a）元素中使用target特性，来告诉浏览器你想要链接到的网页在什么窗口中打开。为_blank或者你

选中的任何名称设置target特性值。记住，使用这个方法，你不能控制窗口的尺寸，而通常会打开与用户浏览器最常打开的窗口相同尺寸的窗口或标签。

设置target="_blank"总会导致浏览器打开新窗口。

例如：

```
<a href="http://www.oreilly.com" target="_blank">O'Reilly</a>
```

如果每个链接都设置target特性值为“_blank”，那么每个链接都将启动新窗口，可能会给用户留下一团乱麻般的窗口。

更好的方法是给target窗口一个具体名称，使后来的链接都可以使用。你可以将窗口命名为任何你喜欢的名称（“new”、“sample”都可以），只要不要以下划线开头就行了。下面的链接将打开名为“display”的新窗口。

```
<a href="http://www.oreilly.com" target="display">O'Reilly</a>
```

如果你将网页上所有链接的target设置为“display”窗口，那么每个链接到的文档都会在同一第二窗口中打开。遗憾的是，如果那个第二窗口隐藏在用户当前窗口后面，那么就会看起来像链接失效了一样。

弹窗

尽管在chrome（toolbars、scrollbars等）浏览器中可以打开一个特殊尺寸的窗口，然而，这需要用JavaScript语言来做。这就是大家所知道的弹窗。它们通常用来做广告。实际上，弹窗已经变成了一个特别讨厌的东西，浏览器的用户都倾向于完全把它们关掉。此外，在使用小的移动手机上网的过程中，弹窗都不会显示出来。

也就是说，如果你有充足的理由打开一个特殊尺寸的新的浏览器窗口，我向你推荐由Peter-Paul Koch写的一篇文章。你可以在www.quirksmode.org/js/popup.html看到它。

邮件链接

下面是一个好玩的链接小诀窍：mailto链接。在链接中使用mailto协议，你可以链接到一个E-mail地址。当用户单击mailto链接时，浏览器将打开一个新的邮件信息，并将地址改为mailto指定的地址。

示例mailto链接如下：

```
<a href="mailto:alklecker@hotmail.com">Contact Al klecker</a>
```


垃圾邮件机器人 (Spam-Bots)

要知道，将邮件地址放在文档源码中，会使这个邮箱很容易收到不清自来的垃圾邮件（称为spam）。制作垃圾邮件列表的人，有时使用自动搜索程序（称为机器人）来搜查Web上的邮件地址。

如果你想把你的email在页面上展示给用户，又不想被机器人看到，你可以以某种方式拆分email地址，既便于人们理解，又不被机器人看懂，比如“jen [-at-] oreilly [dot] com”。

但是这个技巧无法用于mailto链接，因为mailto要求邮件地址必须准确。解决这个缺陷的办法是使用JavaScript来加密email地址，Hivelogic的Enkoder Form (hivelogic.com/enkoder/) 就是这么做的。只需要输入链接文本和邮件地址，Enkoder就会生成代码，你可以直接复制粘贴。

否则，如果你不想受垃圾邮件的困扰，那就不要在HTML文档中出现你的邮件地址。

如你所见，这是一个带有href特性的标准锚元素。href特性值设置为mailto:name@address.com。

浏览器必须事先进行启动浏览器的配置，所以这个效果并不是100%的用户能体验到的。如果你将E-mail本身用作链接中的文本，那么当mailto功能无效时，页面也就保持不变，用户也不会离开原先的页面。

电话链接

要知道，使用智能手机来访问你的网站的用户，可能也会使用智能手机来打电话！为什么不让用户在你的站点上只需要简单的单击就可以打个电话呢？只需要使用tel：就可以，非常简单。

```
<a href="tel:+18005551212">Call us free at (800) 555-1212</a>
```

当移动用户单击链接时，他们会看到一个提示框，询问他们是否确认要打一个电话。大多数设备都支持这个功能，包括iOS，Android，Blackberry，Symbian，Internet Explorer和Opera Mini。iPad和iPod Touch不能打电话，但是它们可以用号码来创建一个新的联系人。当桌面用户单击链接时，什么都不会发生。如果这让你厌烦，你可以使用一个CSS规则来为非移动设备隐藏这个链接（遗憾的是，这不在我们的讨论范围）。

使用电话链接有一些非常好的实践经验：

- 推荐包括完整的国际通用拨号形式，建议为tel:加上国际区号，包括国家代码，因为没有办法知道用户将在哪里访问你的站点。
- 在内容链接里添加电话号码，如果链接不起作用，电话号码仍然可用。
- Android和iPhone有一个电话号码功能，可检测电话号码并自动把它们变成链接。遗憾的是，一些不是电话号码的10位数字也可能会变成链接。如果你的文件有串可能会让人误以为是电话的数字，你可以在你的文档的头部包含如下所示的meta元素，来关闭自动检测。

```
<meta name="format-detection" content="telephone=no">
```

对于Blackberry设备，使用以下命令：

```
<meta http-equiv="x-rim-auto-match" content="none">
```

自我测验

本章最重要的一课就是如何为链接和图像编写URL。这里还有一个机会可以提高你使用路径技能的能力。

使用图6-12中所示的目录层次，写出下列链接和图片的标记。作为示范，我给你填写了第一个。答案都在附录A中。

图表应该为你回答这些问题提供了足够的信息。如果你想亲自总结出目录层次，那么可以从本章材料的test目录中获取这个目录结构。这些文档都是虚拟文件，没有内容。

1. 在`index.html`（网站主页）中，写出链接到`tutorial.html`的标记。

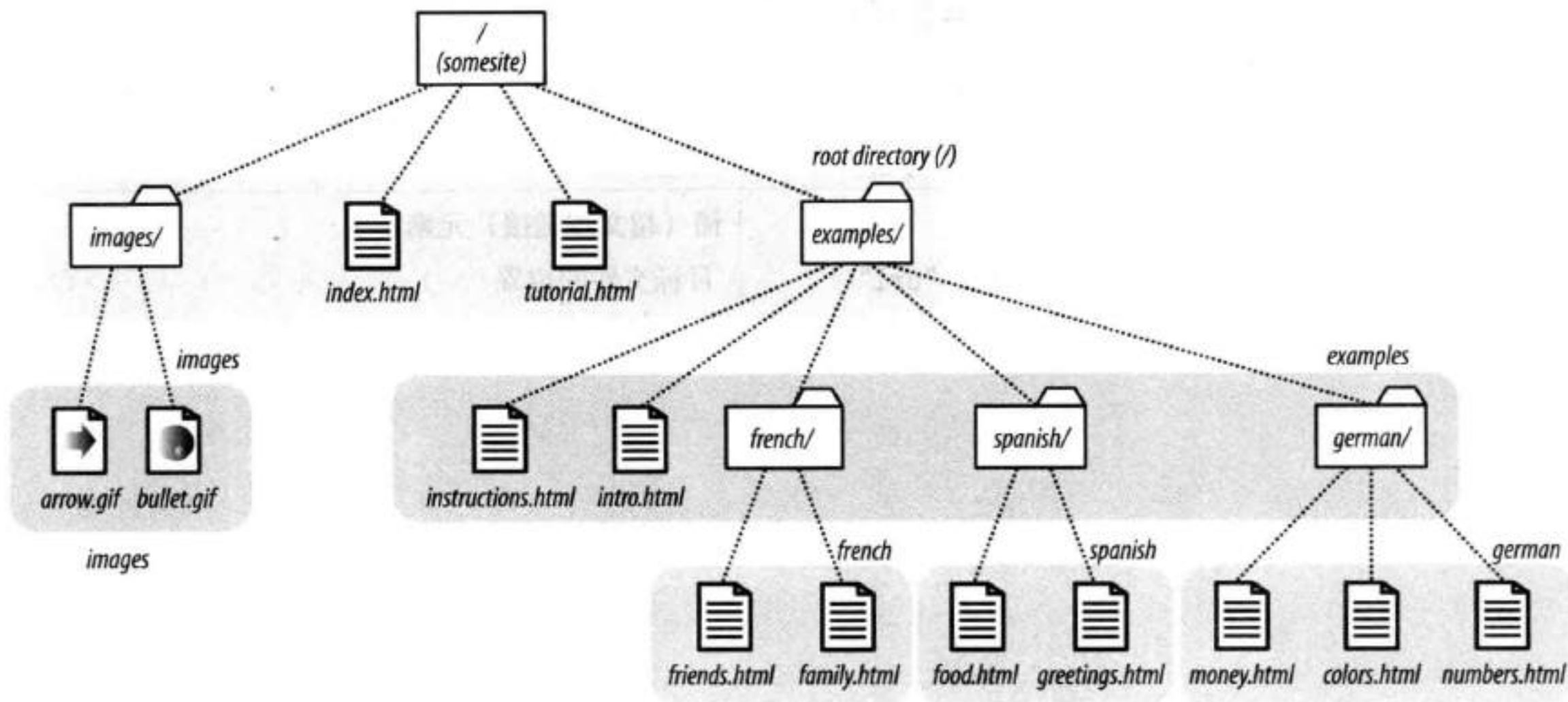
```
<a href="tutorial.html">...</a>
```

2. 在`index.html`中，编写链接到`instructions.html`的锚元素。
3. 创建从网页`tutorial.html`到`family.html`的链接。
4. 创建从`family.html`网页到`numbers.html`的链接，要求以根目录开头。
5. 创建从网页`instructions.html`返回到主页（`index.html`）的链接。

提示

`../`（一个或多个）总是要出现在路径名的开头，千万不要出现在中间。如果你写的路径名中`../`在中间，那你一定犯了什么错。

图6-12：“自我测验”问题的目录结构



6. 在文件*intro.html*中，创建指向本书网站 (*www.learningwebdesign.com/4e/materials*) 的链接。
7. 创建从网页*greetings.html*到*instructions.html*的链接。
8. 创建从*money.html*返回到主页 (*index.html*) 的链接。

我们还没有详细地学习图像 (*img*) 元素，但是你应该能够填写*src*特性后面的URL，来指定这些例子中图像文件的位置。

9. 将图像*arrow.gif*置于网页*index.html*中，标签是：

```

```

10. 将图像*arrow.gif*置于网页*intro.html*中，标签是：

```

```

11. 将图像*bullet.gif*置于*friends.html*网页中，标签是：

```

```

元素回顾：链接

确实只有一个元素与创建超文本链接相关：

元素和特性	描述
<i>a</i> <i>href="url"</i>	锚（超文本链接）元素 目标文件的位置

第7章 添加图片

本章内容

在网页中添加图片
使用src、alt、width、
height等属性

只有文字没有图片的网页会令用户感到乏味。网站的普及一定程度归功于网页上有大量图片。在图片出现之前，因特网是只有文字的荒原。

如今，图片在网页上以两种方式出现：作为内嵌内容的一部分或作为映衬的背景图片。背景图片是使用层叠样式表单实现的，我们将在第13章中讨论。随着标准化的设计不断涌现，仅仅用于装饰目的的层叠样式图片已经不再被人们所青睐。想要了解关于此趋势的更多信息，参见下页侧栏“图片朝着背景图片方向发展”的文章。

本章将重点讨论用img元素将图片嵌入网页。当用图片作内容的时候，使用img元素。比如，产品速拍、图片、广告、阐述等，你会有自己的想法的。

首先，关于图片格式的信息

我们即将开始讲述img元素及标示例子，但首先你必须明确，不是任何图片都是可以插入网页中的。只有GIF，JPEG或是PNG格式的文件可以当作内嵌图片。第21章解释了这些格式及图片形式是最容易处理的。除了拥有适当的格式，图片文件必须以一定的后缀命名——分别为.gif，.jpg（或.jpeg）和.png。这样，才能被浏览器所识别。

如果你的源图片是其他常见格式的，比如，TIFF，BMP或者EPS，在使用之前，你必须将其转换成一种网页认可的形式。如果出于某些原因，你必须保持图形文件原有的形式（例如，一个CAD程序的文件或一个向量格式的图片），你可以使用文件链接，将它作为可用的外部图片，就像：`Get the drawing`一样。

浏览器使用帮助程序来显示他们不能单独处理的媒体。浏览器会将链接中文件的后缀匹配到合适的帮助程序中。外部图片会在一个独立的程序窗口

作为背景的图像

纯粹作为装饰的图像比起文档结构和内容来说对外观的影响更大。因此要使用样式表而不是标记来控制。

使用CSS，可以把一个图像作为页面或者任意文本元素（比如div、h1、li等）的背景，我们将在第13章中介绍。

只用文档结构之外的外部样式表来指定装饰图像有几个好处，它既可以使文档更简洁，更方便访问，也更便于对站点外观进行修改。

如果想了解如何不使用img元素就能制作漂亮的页面，可以查看CSS Zen Garden的文章“Select a Design”中的示例：www.csszengarden.com。

中打开；如果帮助程序嵌入了浏览器中，例如，嵌入到QuickTime中，外部图片也可以在浏览器窗口中打开。浏览器也可能会提示用户保存文件或手动打开一个程序。

下面直接来看看img元素，以及它所需的和推荐的属性。

img元素

加入一个内联图片

img元素告诉浏览器，“这里将插入图片”。你通过第4章和第5章的学习已经初步掌握如何插入横幅图片。使用时，在你想要添加图片的地方，将其放在文本流中，就像以下这个例子一样。图像会在文本流中，而不会引起换行（HTML 5中称为短语元素），如图7-1所示。

```
<p>I had been wanting to go to Tuscany 
for a long time, and I was not disappointed.</p>
```


I had been wanting to go to Tuscany  for a long time, and I was not disappointed.

图7-1：默认情况下，图片将与周围文本的基准线对齐，不会引起换行

当浏览器遇到img元素时，它会向服务器发出一个请求，先检索图片文件再将其显示到网页上。即使对于每个图片文件，浏览器都会发出独立的请求，网络及计算机的速度会让它显得是瞬时发生的。在比较慢的移动终端上，我们可能会意识到等待图片显示的时间有点长。这种情况同样出现于使用拨号上网或其他比较慢的网络环境中，例如，奢侈的旅馆里的昂贵的WiFi环境。

当设计移动网页实例时，通常情况下限制服务器请求的次数是非常明智的，这就意味着要认真地考虑网页中的图片数量。

以上例子中，src和alt属性是必需的。src属性指示浏览器图片文件的地址；当图片无法显示时，alt属性提供显示用的替换文本。在接下来的章节中，将更多地讨论src和alt属性。

关于img元素，还有一些值得注意的地方：

src属性和alt属性在img元素中是必需的。

- 它是一个空元素，本身不包含任何内容。你只需要将它放在需要插入图片的流文本中。
- 如果选择用较严格的XHTML规则，那么就需要用加斜线的方式终止空元素img，如。
- 它是一个内嵌元素，所以表现得像流文本中其他内嵌元素一样。图7-2显示了图片元素的内嵌性质。当调整浏览器窗口的大小时，图片行的流向也会改变，以适应新的窗口宽度。
- img元素以置换元素著称，因为当网页显示时，它会被一个外部文件置换。这与文本元素是不同的，文本元素在其所属源中有自己的内容（因此是非置换的）。
- 默认情况下，图片的底部将与文本的基准线对齐，如图7-1和图7-2那样。采用层叠样式表，你可以将图片移动到右部或左部边缘，以及允许文本环绕在其周围，控制图片的间隔及边界，还可以改变其垂直对齐方式。我们将会在第三部分中讨论这些方式。



图7-2：内联图片是文档流的一部分，当浏览器窗口调整大小时，图片流向会改变

用src提供地址

`src="URL"`

图片的资源（地址）

`src`属性的值是图片文件的URL。大多数情况下，你在网页中使用的图像是存储于你的服务器上的，因此你需要使用相对URL来指出它们。如果你已经阅读了第6章，你现在应该能很好地添加相对URL。简而言之，如果图片与HTML文档在同一个目录下，你就只需要在`src`属性中填写图片的名称：

```

```

开发者往往将一个站点的所有图片都放到同一个目录下，该目录一般命名为`images`、`assets`或`graphics`。甚至可能站点的每部分都分别拥有独立的

提示

利用缓存

下面是使图片更快显示，并且减少服务器流量的一些提示。如果你在你的站点的不同位置使用相同的图片，要确保每个img元素都指向服务器的相同图像文件。当浏览器下载图像文件时，它会把图像储存在硬盘缓存（用于临时存放文件的硬盘空间）。这样，当浏览器需要重新显示页面时，只需要取出本地缓存的源文档和图像文件，而不必再从远程服务器上重新载入。

当你在站点或者页面上分别使用相同的图像时，浏览器只需要下载该图像一次。随后每次需要使用这个图像时，都从本地缓存载入，这样就减少了服务器的流量，而且可以显示得更快。

浏览器通过图像的路径名来识别图像，而不仅仅是通过文件名，如果你想要利用文件缓存，要确保图像的每个应用实例都指向服务器上相同的图像文件，而不是指向同一图像在服务器上不同目录中的多个复制品。

图片目录。如果图片与文档不在同一个目录下，你需要为图片文件提供路径名称。

```

```

当然，你也可以插入从其他网站获取的图片（只是需要确定允许你这么 做），这样只需要使用一个绝对URL，就像：

```

```

用alt提供替换文本

alt="text"

替换文本

每一个img元素也必须包含alt属性，当无法看到图片时，比如，有的用户使用屏幕阅读器、盲文或者小型移动设备，该属性提供了对图片的简要描述。替换文本（也称为alt text）应作为图片内容的替补，它与图片具有相同的目的，表达同样的信息。

```
<p>If you're  and you know it clap  
your hands.</p>
```

屏幕阅读器可能以如下方式显示图片及其alt属性值：

"If you're image happy and you know it clap your hands."

如果图片纯粹用作装饰，对网页的文本内容没有任何意义，那么建议你 将alt属性设为空值，如以下例子及本章其他例子所示（你可能也会想是 否利用CSS将其处理成背景图片更合适，这个问题与当前主题有些偏离 了）。并且注意，在引号之间是没有空格的，正如下面的例子和本章其 他的例子所示：

```

```

然而，不要省略所有的alt属性，因为它会引起文档无效（验证文档在第 3章中已有介绍）。对于网页上的所有内嵌图片，都要考虑替换文档朗读 时听起来是否合适，以及对于读屏的用户，它是增加其体验还仅是强迫 其体验。

如果用户使用图形化浏览器，替换文本也是有其好处的。当用户选择用 浏览器屏蔽图片或图片根本无法下载时，浏览器可以显示替换文本，以 提示用户所缺损的信息。然而，对替换文本的处理在各浏览器中是不一 致的，如图7-3所示。



图7-3：当图片不可用时，大多数浏览器在图片位置显示替换文本（以图标或内联文本的方式），然而Macintosh OS X下的Safari是一个例外

图片可访问性

对使用屏幕阅读器进入Web的用户来说，图片和其他无文本内容是一个挑战。当用户看不到图片时，替换文本允许你为图片内容提供一个简短的描述。但是，有些类型的图像，如数表和图框，要求比实际的ALT属性更长的描述。

对于特别长的描述，要考虑将其写在网页上的其他地方，或者写在一个单独的文档中并在图像附近做一个指示链接。

HTML 4.01包括longdesc(长描述)属性，但是在HTML 5中，由于缺少支持，该属性被丢弃。Longdesc属性指向一个单独的HTML文档，其包含图片描述的长度，如下例所示：

```

```

在HTML 5中，figcaption元素允许图片的长描述放在figure中。

除了在本章中的叙述，关于图像获取仍有很多细节。我建议你在这类资源方面开始研究：

- 在WebAIM上 (webaim.org/techniques/images/longdesc) “创建可获取的图片”，为longdesc属性提供替代文件。
- JoeClark所著的《Building Accessible Websites》书中“第6章The Image Problem” (joelclark.org/book/sashay/serialization/Chaper06.html)。
- W3C中的网页内容获取规则 (WCAG2.0)，包含了提高所有网页内容的可访问性的技术 (www.w3.org/TR/WCAG20-TECHS/)。警告：这个相当难懂。

注意： 一个基于设备大小的img元素的不同图片服务文件是由运行在服务器上的JavaScript或某一程序处理的。这超出了本章的范围，将会在第18章中“自适应图片”侧栏里介绍。

提示

使用浏览器来查看像素大小

当然，你可以通过打开图片的编辑程序来查看图片的像素大小，但是你知道也可以使用网页浏览器吗？

使用Chrome、Firefox或Safari浏览器（但是抱歉，Internet Explorer不能用），只需要打开图片文件，它的像素大小和文件名就会在浏览器的工具栏里显示。这就是我一直以来使用的快捷方式，因为我的浏览器似乎一直开着。

提供宽度与高度尺寸

`width="number"`

图片宽度的像素

`height="number"`

图片高度的像素

`width`和`height`属性以像素来表明图片的大小尺寸。听起来很平常，但这两个属性能加快最终网页的显示速度（以秒来计）。每次一个新的图片出现的时候，浏览器可以使用足够的空间来容纳图片而不是重新构建页面。

如果你用一个固定尺寸的图片来设计页面的版式，这是非常好的。但是，在现在的网页设计中，由相同的图片创造出几种页面版式，并且把小的发送到手机的终端，把大的图片发送到大屏幕的终端是非常普遍的。如果你在设计中需要缩放图像的比例或者传输具有多重尺寸的图片，在标记中不要用`width`和`height`属性。

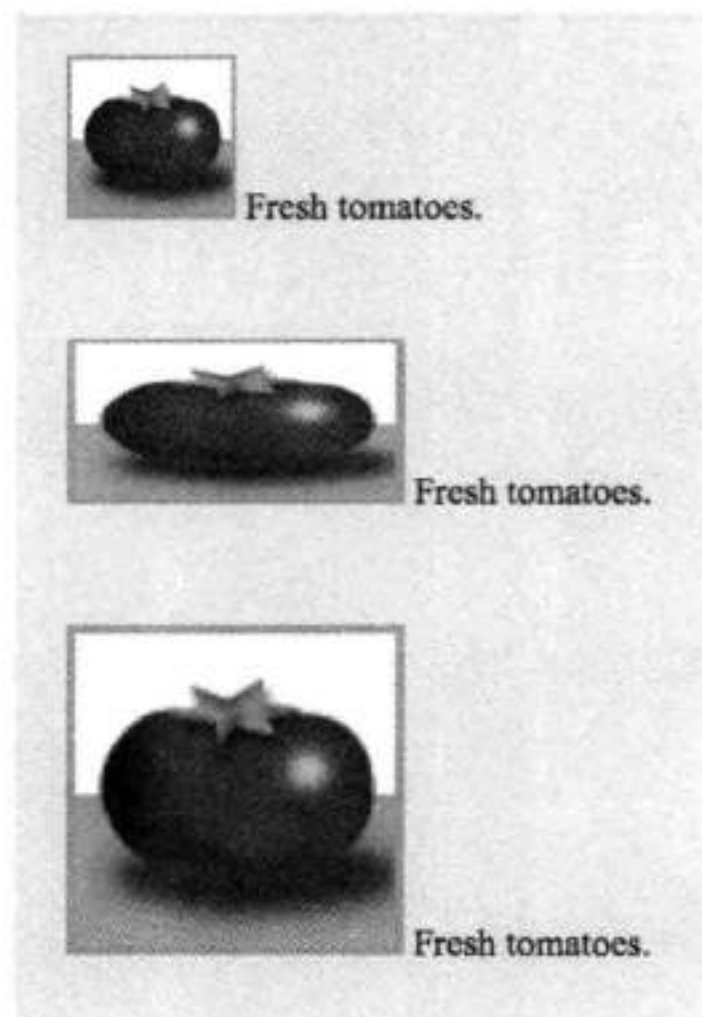
记住这些附加说明，看一下`width`和`height`这两个属性是如何应用于实例中的。

赋予实际像素值

确定你所提供的像素大小为图片实际大小。如果像素值与图片实际尺寸不等，浏览器会改变图片大小来匹配指定值（图7-4）。

虽然这种方式可以改变图片大小，但你应该避免这样做。使用大图片时，即使在页面上显示出来很小，仍需要下载相应的很大的文件。如果你希望网页上出现的只是小图片，就不能强制用户进行这样高负荷的下载。更好的做法是花一定时间用编辑软件来改变图片本身的大小，然后在网页上赋予其实际尺寸。不仅如此，使用属性来改变大小往往会使图片模糊及变形。事实上，当你的图片在浏览器上一直都显得不清晰时，首先应该检查`width`和`height`值是否与图片的实际大小一致。

图7-4：浏览器改变图片的大小来匹配所提供的`width`和`height`值。强烈建议不要以这种方式来改变图片大小



`width="72" height="72"`
(图片的实际大小)

`width="144" height="72"`

`width="144" height="144"`

练习7-1 添加并链接图片

你刚从意大利回来，打算上传一些旅行照片与朋友和家人分享。在本练习中，你需要将缩略图添加到旅行网站中，并将他们链接向更大的照片。

你所需要的缩略图及照片已经创建好了，我还为你提供了基本的HTML文档。这些资源都可以在www.learningwebdesign.com/4e/materials上找到。将tuscany文件夹下载到你的硬盘上，确定让它保持不变的结构。同样地，答案提示在附录A中。

这个小网站由一个首页 (*index.html*) 及包含各个大图片的三个单独XHTML文档构成 (图7-5)。首先，添加缩略图；然后，将实际大小的图片添加到各自页面中；最后，将缩略图链接到那些页面。现在开始吧。打开 *index.html* 文件，将缩略图及附加文字添加到其中。我已经为你做好了第一个：

```
<h2>Pozzarello</h2>
<p>The house we stayed in was called
Pozzarello...
```

我已经将图片放在最开头的段落，开放的<p>标签之后。由于所有的缩略图都放在*thumbnails*目录中，采用URL形式来提供路径名。同时我也为图片添加了描述及Width和height的像素大小。

现在轮到你了。在各节最开始的段落中添加图片 *countryside_100.jpg* (100像素×75像素) 和 *sienna_thrumb.jpg* (75像素×100像素)。确定要包含路径名、替换文本描述以及像素大小。

完成之后，保存文件并把它在浏览器中打开，确定图片可显示而且大小正确。

1. 接下来，将图片添加到其各自的HTML文档中。我已经为你做了 *window.html*：

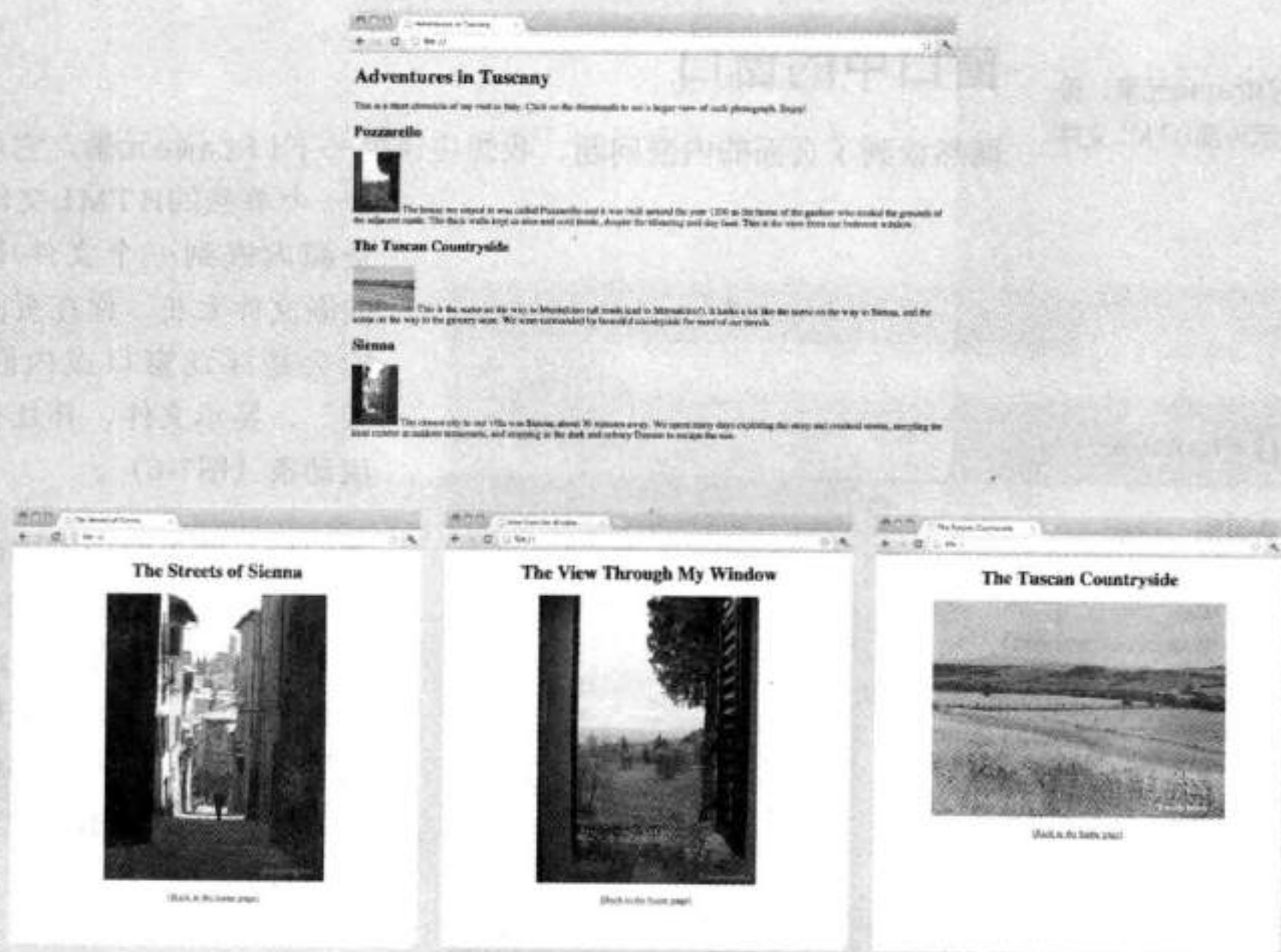


图7-5：旅行照片站点


```
<h1>The View Through My Window</h1>
<p></p>
```

注意原大小的图片是放在`photos`目录下的，因此必须反映出路径名称。

依照例子，将图片分别添加到`countryside.html`、`sienna.html`和接下来的例子中。提示：尽管照片横竖方向不同，但其都是长边为500像素，短边为375像素。

保存每个文件并在浏览器窗口打开以检查它们。

2. 回到`index.html`，将缩略图链接到他们各自的文件。我这里做了第一个：

```
<h2>Pozzarello</h2>
<p><a href="window.html"></a></p>
```

注意，URL与当前的文档（`index.html`）有关，而与图片位置（`thumbnails`目录）无关。

把剩下的缩略图链接到其相关文档。如果所有的图片都可显示，也能链接到每个页面，且之后又能返回首页，那么祝贺你，你成功了！

想要更多的练习？

如果你想要更多的练习，你可以在适当的目录找到三张附加图片（`sweets.jpg`、`cathedral.jpg`和`lavender.jpg`），以及它们的缩略图（`sweets_thumb.jpg`、`cathedral_thumb.jpg`和`lavender_thumb.jpg`）。这次，你需要从零开始，在首页上自己添加描述，并为原大小图片创建HTML文档。

作为更大挑战，在`tuscany`目录下创建一个新的`photopages`目录。将`countryside.html`和`sienna.html`文档移动到新建目录中，然后更新那些页面上所有的URL，使图片重新显示。

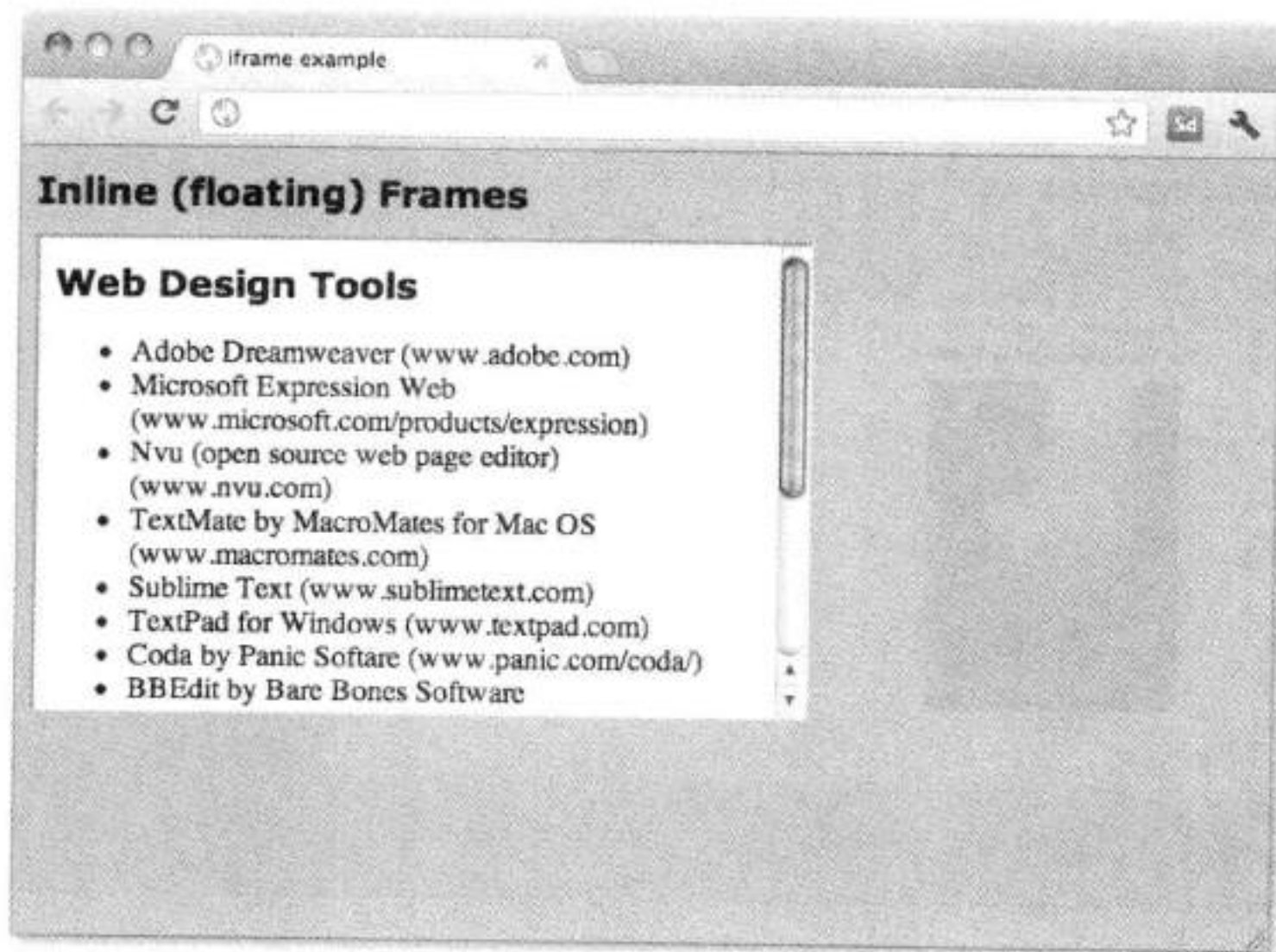
图7-6：内嵌框架（加入了`iframe`元素）像浏览器中的浏览窗口，显示外部HTML文件和资源

窗口中的窗口

既然谈到了页面的内嵌问题，我想应该提一下`iframe`元素，它可以实现

将一个单独的HTML文件或其他资源内嵌到一个文件中。假如内嵌文件太长，你在页面中看到的会是浮动窗口或内嵌的“框架”，显示文件，并且有自己的滚动条（图7-6）。

类似地，将页面中的内嵌框架放在一张图片中，指定它的`src`以及`width`和`height`。`iframe`元素中的内容会自动在不支持该元素的浏览器上显示。本例是在一个内嵌框架中显示一个叫做`list.html`的文件。



```
<h1>Inline (floating) Frame</h1>

<iframe> src=" list.html" width=" 400" height=" 250" >


Your brower does not support inline frames.Read the <a href=" list.
html" >list</a>

</iframe>
```

你很少看到内嵌框架，但是开发者经常要用到它们来隔离像互动广告和其他窗口小部件等第三方内容，使得它们不会影响到页面其他部分中的脚本和内容。

自我测验

图片是Web中重要的组成部分。回答以下问题，看你是否掌握了本章的核心概念。在附录 A中可以找到正确答案。

1. 哪些属性是元素中所必须包括的？
2. 写下添加图片*furry.jpg*的标记，它与当前文本在同一目录下。
3. 为什么需要包含替换文档？回答两个原因。
4. 网页上每个图片都包含width和height属性的好处是什么？什么时候不必考虑它们？
5. 当你在浏览器中查看图片时，如果不能显示，可能是什么原因？有三种可能的解释。

元素回顾：图片

本章只讲了一个元素：

元素与属性	描述
<div>img</div> <div>src="url"</div> <div>alt="text"</div> <div>width="number"</div> <div>height="number"</div> <div>usemap="usemap"</div> <div>title="text"</div>	<div>插入一个内嵌图片</div> <div>图片文件的位置</div> <div>替换文本</div> <div>图形的宽度</div> <div>图形的高度</div> <div>表示客户端图像地图</div> <div>当用户鼠标滑过时，提供提示工具。可以作为图片的补充信息</div>
<div>iframe</div> <div>height="number"</div> <div>src="url"</div> <div>width="number"</div>	<div>插入一个内嵌浏览器（窗口）</div> <div>框架的像素高度</div> <div>显示在框架中的内容</div> <div>框架的像素宽度</div>

第8章 表格标记

在讲述表格标记之前，先回顾下已学过的内容。目前的学习已经涵盖了很大范围：如何建立HTML文档的基本结构，如何标记文本以赋予其意义及结构，如何添加链接，以及如何在网页上添加图片内容。

本章及第9章描述了特殊内容的标记，你也许不会立即用上它们。如果你急不可耐地希望把页面做得漂亮，可直接跳到第三部分玩层叠样式表。当你准备好后，再回来学习表格及表单的章节。

仍然准备继续吗？很好，开始学习表格吧。首先回顾怎样使用表格，然后学习创建HTML表格的元素。记住，这是有关HTML的章节，所以我们把重点放在使用标记将内容结构化地放进表格中，而不关注表格的外观。如同所有的网站内容一样，表格的外观（或者是在网络发展世界中提到的表达）应该由样式表控制。这些你将会在第18章中学到。

如何使用表格

当网页上需要添加表格材料（以行列形式组织的数据）时，便创建HTML表格。表格可以用来组织日历、日程表、统计数据或其他类型的信息，如图8-1所示。注意这里“数据”并不一定指数字。一个表格单元可以包含所有种类的信息，包括数字、文本元素，甚至图片及多媒体对象。

在可视化浏览器中，数据以行或列的方式排列，使用户对数据单元及其各自表头标签有直观的了解。当创建表格时，切记有的用户会使用屏幕阅读器听朗读出的内容或阅读盲文输出。本章稍后会讨论如何让表格内容对无法使用可视表达的用户可用。

本章内容

怎样使用表格

基本表格结构

表头的重要性

合并行或列

单元格填充与间隔

让表格可访问

表的麻烦

大表，如图8-1所示，在小屏幕设备上使用可能是困难的。默认情况下，它们会缩小，以适应屏幕宽度。用户可以放大来读取单元格，但只有少数单元格可能是在有的时候可见，并且它很难解析标题和列组织方式。

实际上，写这篇文章时，我们才刚刚开始弄清楚如何最好地处理在小屏幕上的表格。一种方法是在移动设备上以图形表示，如饼图，更换表。当然，这将仅适用于某些类型的表。简单的两栏或三栏的表，可以考虑使用一个dl列表来表示信息，而不必考虑太多的灵活性。另一种方法是把表的提示（如它的顶部的图像），链接到一个单独的屏幕，便于那些有兴趣的人来查看。Chris Coyier在他的文章“敏感的数据表”（[/css-tricks.com/9096-responsive-data-tables](http://css-tricks.com/9096-responsive-data-tables)）中提出了一个聪明的解决方案，介绍了如何使用CSS将表格式化为一个长的、狭窄的表（以便）适合智能手机的屏幕。另请参阅由Filament Group提出的解决方案（他们是响应式设计的发烧友），见filamentgroup.com/lab/responsive_design_approach_for_complex_multicolumn_data_tables/。

你学习本章的时候可能有新的解决方案，但重要的是在设计任何网页内容时，要始终将移动设备、小屏幕体验牢记在心。

Element	Description	Categories	List of elements		Attributes	Interface
			Parents†	Children		
a	Hyperlink	flow; phrasing*; interactive	phrasing	transparent*	globals: href, idref, rel, rev, hreflang, type	HTMLAnchorElement
abbr	Abbreviation	flow; phrasing	phrasing	phrasing	globals	HTMLAbbrevElement
address	Contact information for a page or section	flow; sectionblock; candidate	flow	flow*	globals	HTMLAddressElement
area	Hyperlink or dead area on an image map	flow; phrasing	phrasing*	empty	globals: alt, coords, shape, href, idref, hreflang, role, rel, rev, target, type	HTMLAreaElement
article	Self-contained syndicable or reusable composition	flow; sectioning; contentblock; candidate	flow	flow	globals	HTMLArticleElement
aside	Sidebar for tangentially related content	flow; sectioning; contentblock; candidate	flow	flow	globals	HTMLAsideElement
audio	Audio player	flow; phrasing; embedded; interactive	phrasing	flow or "transparent"	globals: src, controls, autoplay, loop, muted, volume	HTMLAudioElement
b	Keywords	flow; phrasing	phrasing	phrasing	globals	HTMLStrongElement
base	Base URL and default target browsing context for hyperlinks and forms	metadata	none	empty	globals: href, target	HTMLBaseElement
bdi	Text directionality isolation	flow; phrasing	phrasing	phrasing	globals	HTMLBidiIsolationElement
bdo	Text directionality formatting	flow; phrasing	phrasing	phrasing	globals	HTMLBidiOverrideElement
blockquote	A section quoted from another source	flow; sectioning; root; contentblock; candidate	flow	flow	globals: cite	HTMLQuoteElement

w3c.org

PM	7:30	8:00	8:30	9:00	9:30	10:00	10:30
ABC	<i>The Adventures of Ozzie and Harriet</i>	<i>The Patty Duke Show</i>	<i>Gidget</i>	<i>The Big Valley</i>		<i>Amos Burke — Secret Agent*</i>	
CBS	<i>Lost in Space</i>		<i>The Beverly Hillsbillies</i> #8 25.9 rating	<i>Green Acres</i> #11 24.6 rating	<i>The Dick Van Dyke Show</i> #16 23.6 rating	<i>The Danny Kaye Show</i>	
NBC	<i>The Dick Van Dyke Show</i> #25 23.0 rating			<i>Bob Hope Presents the Chrysler Theatre / Chrysler Presents a Bob Hope Special</i>		<i>I Spy</i>	

wikipedia.org

Providence/Stoughton Line

[Print this Schedule](#)
[Service Alerts](#)

[Providence/Stoughton Line Schedule](#)
[South Station and Back Bay Schedule](#)

[Holiday Schedule Information](#)

Direction: Timing: Redisplay Time:

PROVIDENCE/STOUGHTON LINE INBOUND : Weekday Effective 11/14/11

Train Number	800 AM	822 AM	842 AM	854 AM	858 AM	852 AM	858 AM	906 AM	910 AM	928 AM	912 AM	934 AM	910 AM	914 AM	912 AM	916 AM	918 PM	914 PM	918 PM		
TF Green Airport		05:05				08:13	06:52		07:15				09:23		11:45						
Providence	05:07	05:25		08:07		08:33	07:12		07:35		08:10		09:43		12:05	01:30					
South Attleboro	05:17	05:35		08:15		08:42	07:22		07:45		08:20		09:52		12:15	01:42					
Attleboro	05:27	05:45		08:25		08:52	07:32		07:55		08:30	09:00	10:02		12:25	01:51					
Mattfeld	05:35	05:55		08:35		07:04	07:28	07:44		08:05	08:38	09:05	10:10		12:33	01:58					
Sharon	05:44	06:04		08:45		07:13	07:35		08:14		08:47	09:17	10:19		12:42	02:05					
Braintree		06:29		06:58				07:48	08:28			09:40		10:40			02:20	03:23			
Garden Center		06:35		07:04				07:57	08:35			08:49		10:49			02:27	03:30	0		
Centron Junction	05:51	06:11	06:39		07:08		07:41		08:01	08:24	08:40	08:54	09:24	09:52	10:25	10:50	12:50		0		
Route 128	05:55	06:15	06:44	06:56	07:14	07:34	07:47		08:07	08:30	08:45	08:59	09:25	09:57	10:31	10:57	12:55	02:16	03:36		
Walden Park	05:01	06:21	06:49		07:19		07:52		08:13	08:35	08:49	09:04		10:02	10:35	11:02	01:00		0		
Woburn	06:11	06:31		07:10				08:23		08:14	09:41			10:45	11:12	01:10	02:25				
Back Bay	06:15	06:35	06:59	07:14	07:28	07:40	08:02	08:11	08:27	08:45	08:58	09:18	09:44	10:12	10:50	11:15	01:15	02:33	02:48	03:53	0
South Station	06:20	06:40	07:04	07:19	07:33	07:45	08:07	08:16	08:32	08:51	09:03	09:23	09:49	10:17	10:55	11:20	01:20	02:38	02:54	03:58	0

mbta.org

图8-1：表格用于表格信息，如图表、日历及日程表

在样式表出现之前，表格是创建多层布局或控制对齐与空格的唯一选择。布局表格，特别是复杂嵌套表的安排曾经是标准Web设计的筹码，而我们现在已不再需要它们，并且极不鼓励使用。本章着重讲所鼓励使用的HTML表格。

最小表结构

让我们以一个简单的表格为例看看它是如何构成的。以下是个三行三列的小表格，列出了营养信息。

菜单项	卡路里	脂肪
鸡汤面	120	2
凯撒沙拉	400	26

图8-2根据HTML表格模型揭示了表格的结构。所有的表格内容都放在以行组织的单元格里。单元格即包含了表头信息（列的名称，比如“Calories”），也包含了任何类型内容的数据。

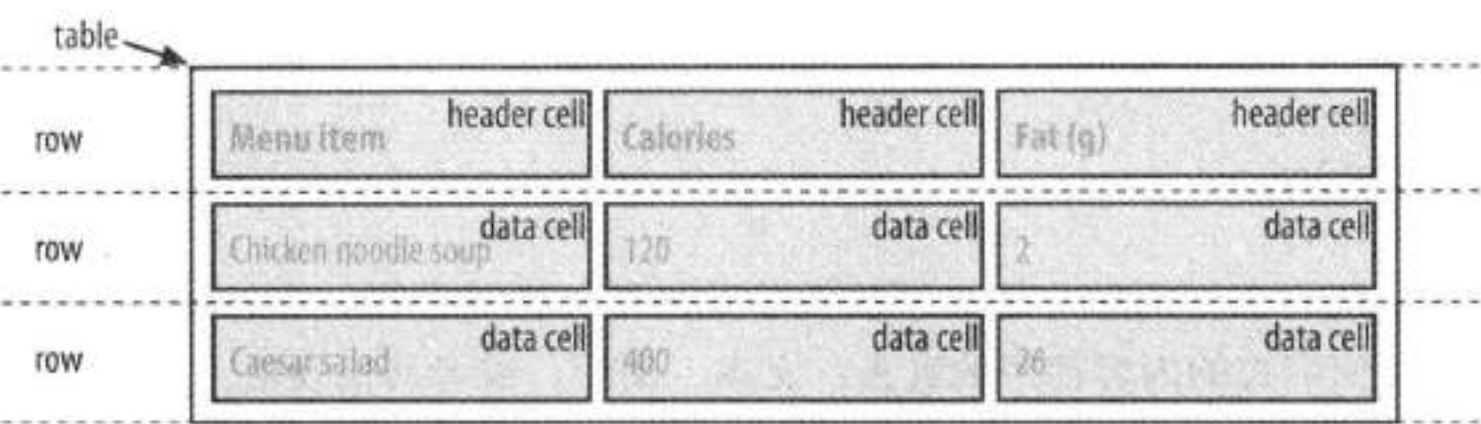


图8-2：表格由包含单元格的行组成。单元格是内容的组成成分

太简单了是吗？现在来看看那些部分如何转换成HTML元素（图8-3）。

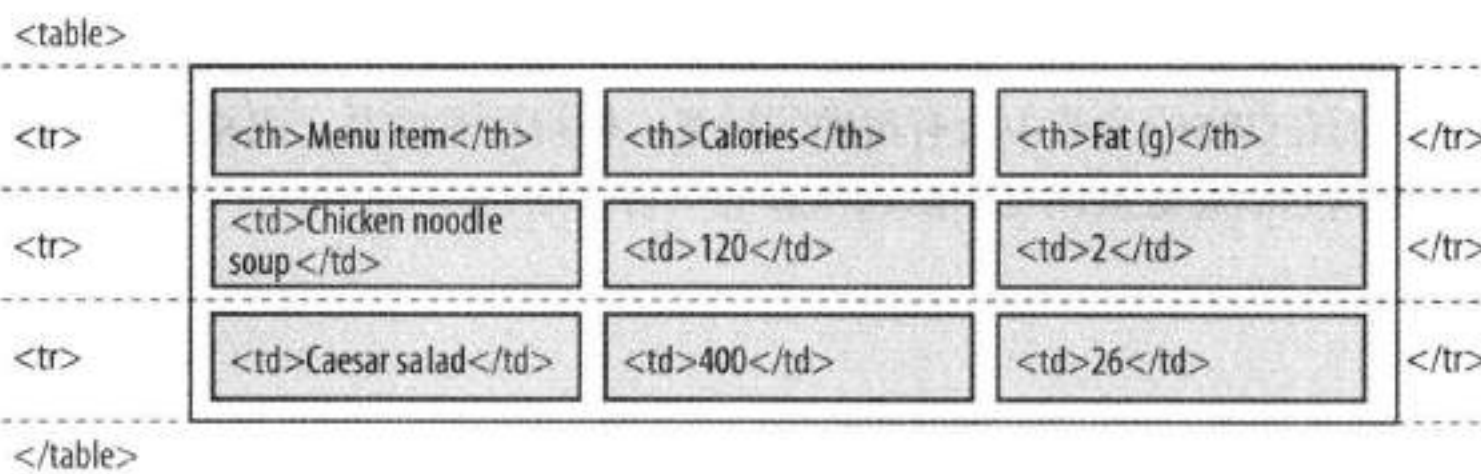


图8-3：创建表格基本结构的元素

图8-3显示了识别表格（table）、行（tr，代表“表格行”），以及单元格（th，代表“表头，” 和 td，代表“表格信息”）的元素。单元格是表格的核心，因为它是真正内容的所在地，而其他元素仅仅是把东西拼到一起。

我们看不到的是列元素（见“注意”）。表格中列的数量取决于每行单元格的数目。这是创建HTML的潜在技巧之一。行是容易的——如果你希望表格有三行，只需要使用三个tr元素。而列不一样。对于一个有四列的表格，你需要让每行都有四个td或th元素，列是默认的。

<table>...</table>

表格的内容（行和列）

<tr>...</tr>

表行

<th>...</th>

表头

<td>...</td>

表格单元格中的数据

注意：在HTML 5中有两个与列有关的元素：`col`用以识别一列，`colgroup`用以建立列群。它们给表格加上一层信息，潜在地加快其显示速度。但它们不属于HTML中以行为中心的表格模型。参看侧栏“高级表格元素”以获取更多信息。

编写源文件时，图8-3中的表格标记看起来与以下的例子很相似。通常都把`th`和`td`元素堆叠起来，便于在源文档中查找。但是这并不影响在浏览器中显示。

```
<table>
  <tr>
    <th>Menu item</th>
    <th>Calories</th>
    <th>Fat (g)</th>
  </tr>
  <tr>
    <td>Chicken noodle soup</td>
    <td>120</td>
    <td>2</td>
  </tr>
  <tr>
    <td>Caesar salad</td>
    <td>400</td>
    <td>26</td>
  </tr>
</table>
```

记住，所有表格的内容都必须放到单元格里，也就是说，在`td`或`th`元素中。任何形式的内容都得放进去：文本、图形，或者其他表格。

起始与结尾的`table`标签用以识别表格的开始与结束。`table`元素中只能直接包含一些`tr`（行）元素。放进`tr`元素中的只能是`td`或`th`元素。也就是说，在不包含`td`或`th`的`table`或`tr`元素中，不能出现文本内容。

最后，图8-4展示了浏览器默认显示出的一个简单页面上的表格，它非常普通，而在CSS的章节中会有很酷的玩意。值得注意的是，表格是块级元素，所以它们总是被浏览器默认从新的一行开始。

高级表格元素

在学习表格机制时，为了达到使其结构清晰的根本目的，没有使用本节例子中的表格。值得注意的是，还有其他表格元素及属性提供了更复杂的语义描述，改善了表格内容的可达性。一个彻底表明版本的表格看上去是这样的：

```
<table>
<caption>Nutritional Information (Calorie and Fat
Content)</caption>

<col span="1" class="itemname">
<colgroup id="data">
  <col span="1" class="calories">
  <col span="1" class="fat">
</colgroup>

<thead>
  <tr>
    <th scope="col">Menu item</th>
    <th scope="col">Calories</th>
    <th scope="column">Fat (g)</th>
  </tr>
</thead>

<tbody>
  <tr>
    <td>Chicken noodle soup</td>
    <td>120</td>
    <td>2</td>
  </tr>
  <tr>
    <td>Caesar salad</td>
    <td>400</td>
    <td>26</td>
  </tr>
</tbody>
```

```
</table>
```

行组元素

你可以将行或行组放到一个表格的表头、注脚或主体中，分别使用元素thead、tfoot和tbody。一些用户代理（也就是浏览器装置）可不断重复多页面表格的表头及注脚。作者也可以使用这些元素为一个表格中的不同区域提供样式。

列组元素

列可以用元素col标记，也可以用colgroup元素放进组中。这对添加列信息的语义背景很有用，也使得计算表的宽度更快。注意，在列元素中没有内容，它只是在真实表格数据开始之上描述列信息。

可达性特征

可达性特征包括如用于提供表格内容描述的标题或摘要，以及用于明确连接表头与其各自内容的scope和headers属性，我们稍候将在本章讨论。

关于高级表格元素的深入探索超出了本书的范围，但是如果你要处理大量数据的表格，可能会希望有更多研究，参见W3C站点（www.w3.org/TR/html5）。

注意： 根据HTML 5规范，一个表可能包含“按这个次序：任选一个标题元素，后跟零个或多个colgroup元素，接着可以跟着thead元素（可选），再接着tfoot元素（可选），然后是零个或多个tbody元素或一个或多个tr元素，最后可跟一个tfoot元素（但只能有一个tfoot子元素）”你明白这一切吗？

定制表格样式

- 一旦你在标记中创建了表格结构，就可以很容易加入一层样式制定其外观。样式表可以也应该用来控制表格的这些视觉外观。我们将在以下章节中学到所有所需的格式化工具。
- 第12章 格式化文本
- 单元格内容的字体设置
 - 单元格的文本颜色
- 第14章 盒子思想
- 表格尺寸（长与宽）
 - 边框
 - 单元格填充（单元格内容的周围空间）
 - 表周围的页边距
- 第13章 颜色和背景
- 背景颜色
 - 铺垫背景图片
- 第18章 CSS技术
- 控制单元格边框与间隔的特殊属性

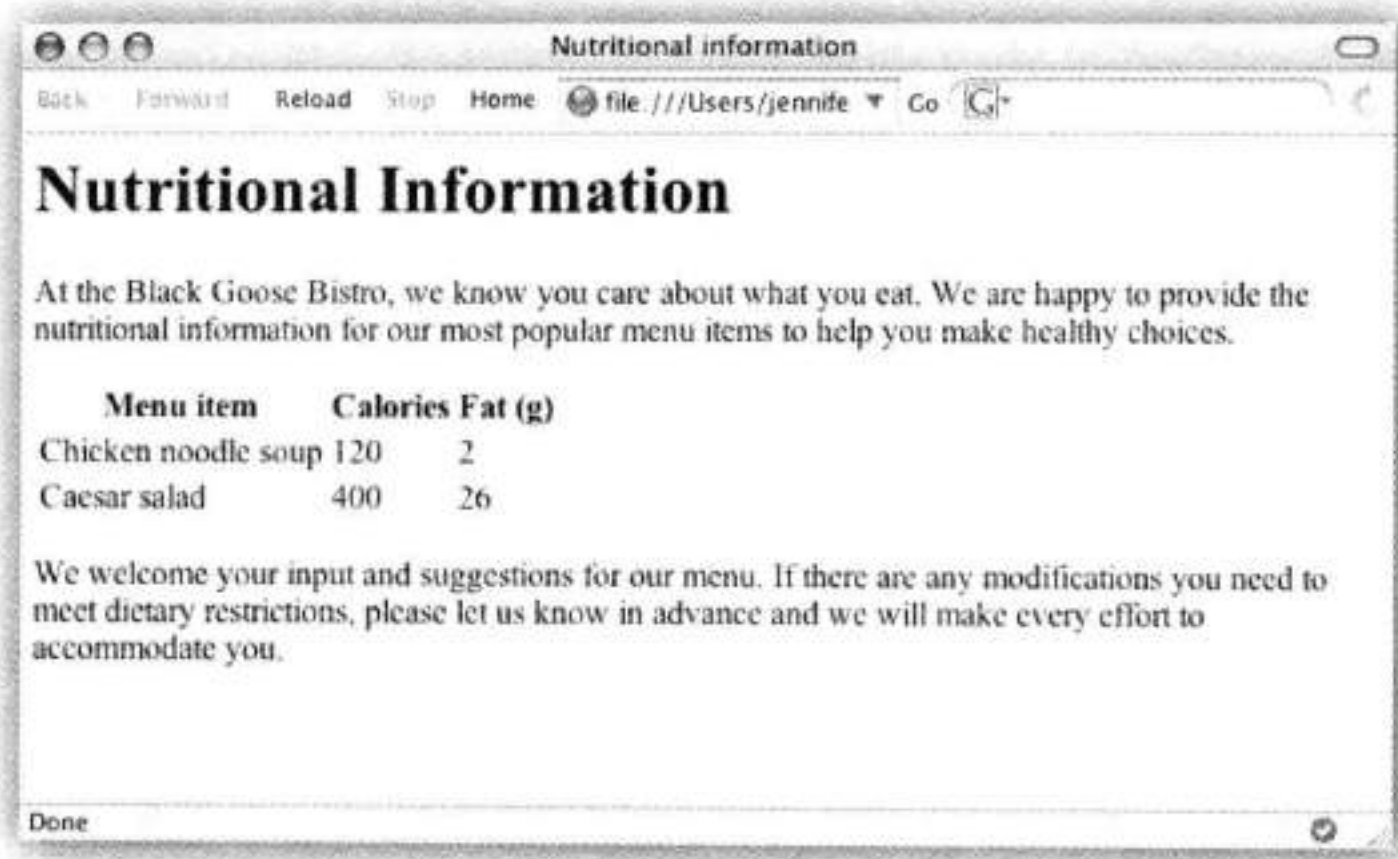


图8-4：在浏览器中默认显示出例子中的表格

这里是另一个表的源码。你能说出在浏览器中显示时它有多少行和列吗？

```
<table>
  <tr>
    <td>Sufjan Stevens</td>
    <td>Illinoise</td>
    <td>Asthmatic Kitty Records</td>
  </tr>
  <tr>
    <td>The Shins</td>
    <td>Oh Inverted World</td>
    <td>Sub-pop Records</td>
  </tr>
</table>
```

如果你说它是一个两个行和三个列的表，你是正确的！两个tr元素创建了两行；三个在每行中的td元素创建了三个列。

表头

正如你在图8-4中看到的，表头的文本标记（th元素）与表格中其他单元格（td元素）显示不同。然而，这种差异并不仅仅用作装饰。表头非常重要，因为它们在之前提供了行或列中单元格的信息及环境。替代浏览器装置可能会采取不同方式对待th元素和td元素。例如，在各数据单元格之前，屏幕阅读器可能会大声地朗读（“Menu item, Caesar salad, Calories, 400, Fat-g, 26”）。

在这种方法中，它们是使得表格内容可达的重要工具。不要采用格式不

同于表格其他部分的td元素来制作假表头。反过来说，不要由于th元素的默认格式（粗体，居中）而避免使用它。符合语义地标识表头，之后再采用样式尺来改变其外观。

以上内容是基础，在我们能做得更棒前，请先做做练习8-1。

练习8-1 创建一个简单的表格

尝试自己写出图8-5中的表格标记。你可以打开一个编辑器，也可以就在纸上书写。附录A中给出了已完成的标记。（注意，我使用样式尺给单元格加了1像素的边框，使其结构清晰。在你的版本中，可以没有它。）

确定关闭所有的表格元素。从技术上讲，你并不需要关闭tr、th和td元素，但我想让你养成整洁编码的习惯，这样在所有浏览设备上都可以有最大的可预见性。如果你选择使用XHTML语法来编写文档，关闭表格元素是必需的，否则文档就是无效的。

Album	Year
Rubber Soul	1968
Revolver	1966
Sgt. Pepper's	1967
The White Album	1968
Abbey Road	1969

图8-5：写出该表格的标记

合并单元格

合并单元格是表格结构的一个基本特征，它是指延伸一个单元格使其覆盖多行或多列。它使你能够创建复杂的表格结构，但也有副作用，即为了维持轨迹使得标记变得有点难度。我们即将讨论到，使用colspan或rowspan属性合并表头或数据单元格。

列合并

使用td或th元素中的colspan属性来进行列合并。它将该单元格向右延伸，跨越随后的列（图8-6）。以下有一个列合并，它使一个表头占有两列。（我给单元格加了边框，以在截图中清晰显示表格结构。）

```
<table>
  <tr>
    <th colspan="2">Fat</th>
  </tr>
  <tr>
    <td>Saturated Fat (g)</td>
```


警告： 注意colspan的值，如果你指定的数值超过了表格中列的数量，大多数浏览器会在现有表格上增加列，这当然会把事情弄乱。

```
<td>Unsaturated Fat (g)</td>
</tr>
</table>
```

Fat	
Saturated Fat (g)	Unsaturated Fat (g)

图8-6: colspan属性使一个单元格向右延伸，跨越正确的列数

注意在第一行 (tr) 只有一个th元素，而第二行有两个td元素。因为th所跨越的列不再出现在源代码中，该含有colspan属性的单元格代表了它。每行单元格数量或等价的colspan值应该相等。例如，有两个td元素，colspan值为2，因此每行所包含列的隐含数量是相等的。

提示：

- 简单起见，本表格全部使用td元素。
- 第二行说明该表格共有三列
- 当合并某单元格后，表格中不再出现td元素。

练习8-2 列合并

尝试写出图8-7中表格的标记。你可以打开一个文本编辑器，也可以就在纸上书写。我在图中添加了样式以揭示表格结构，你的表格不必这样做。在附录A中核对最终标记。

7:00pm	7:30pm	8:00pm
The Sunday Night Movie		
Perry Mason	Candid Camera	What's My Line
Bonanza	The Wackiest Ship in the Army	

图8-7: 写出该表格的标记，以练习合并列

行合并

使用rowspan属性进行行合并，就如列合并一样，只是它使单元格向下延伸跨越几行。在这个例子中，表格中的第一个单元格向下延伸三行 (图8-8) 。

```
<table>
  <tr>
    <th rowspan="3">Serving Size</th>
    <td>Small (8oz.)</td>
  </tr>
  <tr>
    <td>Medium (16oz.)</td>
  </tr>
```

```

<tr>
  <td>Large (24oz.)</td>
</tr>
</table>

```

Serving Size	Small (8oz.)
	Medium (16oz.)
	Large (24oz.)

图8-8: rowspan属性使一个单元格向下延伸, 跨越正确的行数

再次注意, 被延伸取代了的td元素(剩下的行中的第一个单元格)不出现在源代码中。rowspan="3"表示随后的两行单元格, 所以不需要td元素。

练习8-3 行合并

尝试写出图8-9中表格的标记。记住, 合并的单元格并不出现在表格代码中。行总是向下合并, 所以“oranges”单元格也包括第一行, 虽然它的内容看起来是垂直居中的。

如果你用文本编辑器工作, 表格看起来与这里的不完全一致也没关系。可以在附录A中找到答案。

apples	oranges	pears
bananas		pineapple
lychees		

图8-9: 写出该表格的标记, 以练习行合并

提示:

- 行总是向下延伸, 因此单元格“oranges”是第一行的内容。
- 被合并掉的单元格不出现在表格代码中。

单元格填充与间距

默认情况下，单元格的大小仅仅需要与内容相适应，但往往你希望在表格内容周围有些空间（图8-10）。因为间距关系到展示，它是样式表的工作。

单元格填充是单元格内的空间，内容和单元格的边缘之间的空间。要添加单元格填充，应对td或th元素应用CSS padding属性。

单元格间距，单元格之间的区域，比上述更复杂一点。首先，为table和separate设置border-collapse属性，然后使用border-spacing属性来指定边界之间的空间。遗憾的是，这种技术将无法在Internet Explorer 6中使用，但希望在你阅读本书时，IE 6的使用率变得无关紧要。

在过去，单元格边距和间距通过table元素的cellpadding和cellspacing属性控制，但由于其私有本性，它们在HTML 5中已经过时了。

CELL 1	CELL 2
CELL 3	CELL 4

By default, table cells expand just enough to fit the contents.

Cell padding is the space between the edge of the cell and its contents.

CELL 1	CELL 2
CELL 3	CELL 4

Cell spacing is the space between cells.

CELL 1	CELL 2
CELL 3	CELL 4

图8-10：单元格边距和单元格间距

表格可达性

作为一个Web设计师，重要的是，你始终留意你的站点的内容将会如何被盲人访问者使用。特别具有挑战性的是使屏幕阅读器看懂表格，但也有其他一些措施，可以改善体验并使你的内容更容易理解。

描述表的内容

第一步是简单地提供一个表内容的描述，如果可能，包括描述表的结构（如果表格不同寻常）。

使用caption元素给一个表添加一个标题或显示在表格附近的简短说明。你可以用它来描述表的内容或提供它的结构信息。使用时，caption元素必须是table元素中的第一个元素，如在这个例子中所示，为本章前面的营养图表增加了一个标题。

```
<table>
  <caption>Nutritional Information (Calorie and Fat Content)</caption>
  <tr>
    <th>Menu item</th>
    <th>Calories</th>
    <th>Fat (g)</th>
  </tr>

  ...table continues...
</table>
```

如图8-11所示，标题默认显示在表格上方，但你可以使用样式表中的一个属性（caption-side）将它移动到表格下方。

Nutritional Information		
Menu item	Calories	Fat (g)
Chicken noodle soup	120	2
Caesar salad	400	26

图8-11：默认情况下标题显示在表格上方

对于更详细的描述，你可以考虑把表放在figure元素中，并为描述使用figcaption元素。HTML 5规范中有一些提供表格描述的建议，你可以在www.w3.org/TR/html5/tabular-data.html#table-descriptions-techniques找到。

连接单元格和表头

为了提高表内容的可达性，我们已经简短地讨论过表头。但是，有时候人们很难知道哪个header应用于哪个cell。比如，表头可能在左边或者右边而不是在列的上部分。尽管对人们来说，理解表的结构是很容易的。对于那些只能将数据作为文本来听的人来说，整个的结构并不清晰。在HTML 4.01介绍中也给出了一些与可达性有关的附加属性。

scope

scope属性将表头与其用row、column、rowgroup或colgroup表示的行、列、行组或列组（如tbody）明确地关联在一起。以下这个例子使用scope 属性声明将该表头单元格应用于本行。

```
<tr>
```

注意：HTML 4.01包括table元素的summary属性，用于为辅助设备提供长描述，而在可视化浏览器中则隐藏起来。然而，它被HTML 5省略，且使用时会触发验证错误。


```
<th scope="row">Mars</th>
<td>.95</td>
<td>.62</td>
<td>0</td>
</tr>
```

headers

对于非常复杂的表格，使用`scope`来关联表格数据单元格与其表头是不够的（比如，表格涵盖了多个合并单元格的情况），`headers`属性用于 `td`元素内，将其明确地绑定到一个表头的`id`值中。如以下例子所示，表格文本“.38”绑定在表头“Diameter measured in earths”上：

```
<th id="diameter">Diameter measured in earths</th>
...many other cells...
<td headers="diameter">.38</td>
...many other cells...
```

以上部分显然只是可达表格的冰山一角，深入介绍超出了本初级教材的范围。如果你想学习更多，在WebAIM (www.webaim.org/techniques/tables) 上的文章“创建可达表格”是个完美的起点。

小结

本章描述了HTML表格构成的概要。练习8-4提供了关于创建表格的更多练习，涵盖了我们所学习的大部分内容。

经过短短的训练，你可能越来越感到，手工撰写表格标记，虽然不是不可能，但很快就让人感到烦琐和复杂。幸运的是，诸如Dreamweaver这样的Web制作工具都提供了使该过程简化有效的接口。并且，你应该感到高兴，因为你对表格结构和术语，以及改变其外观的首选方法都有牢固的理解。

练习8-4 表格挑战

现在应该开始总结你在本章所学到的编写表格技巧了。你的挑战是写出图8-12中所示表格的源文档。

我将提示你一步步地完成它。

1. 首先在文本编辑器中创建一篇新文档，设置其总体结构（html、head、title和body元素）。在你选择的目录下以table.html保存该文档。
2. 接下来，为了使单元格及表格边框更清晰，以利于结果查看，我将引导你为文档增加一些样式表结构规则。不用完全理解这一步的含义（尽管它相当直观）。只需要在文档的head中添加包含以下内容的style元素。

```
<head>
  <title>Table Challenge</title>
  <style type="text/css">
    td, th { border: 1px solid #CCC; }
    table {border: 1px solid black; }
  </style>
</head>
```

3. 现在开始创建表格了。我通常先设置表格，并添加空行元素来占位，数目与表格最终的行数相等，如下所示（应该非常清楚地看到该表有五行）。

```
<body>
  <table>
    <tr></tr>
    <tr></tr>
    <tr></tr>
    <tr></tr>
    <tr></tr>
  </table>
</body>
```

4. 从最顶行开始，从左往右地依次填入th和td元素，包括所需的行或列合并。我将帮你完成第一行。

第一个单元格（左上角的那个）向下延伸了两行的高度，因此它需要一个rowspan属性。在这里我使用th让它与该行其他单元格保持一致。该单元格中没有内容。

```
<table>
  <tr>
    <th rowspan="2"></th>
    </tr>
```

第一行第二列的单元格横跨了两列，因此它需要一个colspan属性。

```
<table>
  <tr>
    <th rowspan="2"></th>
    <th colspan="2">A common header for two
      subheads</th>
  </tr>
```

第三列的单元格已被合并，所以我们不需要写出其标记。第四列的单元格同样也向下延伸了两行。

```
<table>
  <tr>
    <th rowspan="2"></th>
    <th colspan="2">A common header for two
      subheads</th>
    <th rowspan="2">Header 3</th>
  </tr>
```

5. 现在轮到你了。继续为表格中剩下的四行填写th和td元素。提示：第二行的第一个及最后一个单元格已经被合并了。另外，例子中粗体显示的内容设置成表头。
6. 为了内容的完整性，使用caption元素为表格添加标题。
7. 最后，使用scope属性将表头Thing A、Thing B及Thing C关联到其各自的行中。
8. 保存文件然后在浏览器中打开。表格看起来应该与以上那个相同。如果不一样，返回去检查源标记。如有疑问，可在附录A中找到本练习的最终标记。

Your Content Here			
	A common header for two subheads		Header 3
	Header 1	Header 2	
Thing A	data A1	data A2	data A3
Thing B	data B1	data B2	data B3
Thing C	data C1	data C2	data C3

图8-12：表格挑战

自我测验

以下问题的答案可在附录A中找到。

- 1. HTML的基本部分（元素）是什么？
- 2. 为什么专业Web设计者不再将表格用作布局？
- 3. 你何时会使用col（列）元素？
- 4. 从以下表格标记中找出5个错误。

```
<caption>Primetime Television
  1965</caption>
<table>
  Thursday Night
  <tr></tr>
  <th>7:30</th>
  <th>8:00</th>
  <th>8:30</th>
  <tr>
    <td>Shindig</td>
    <td>Donna Reed Show</td>
    <td>Bewitched</td>
  <tr>
    <colspan>Laredo</colspan>
    <td>Daniel Boone</td>
  </tr>
</table>
```

元素回顾：表格

以下是本章出现的元素概括：

元素与属性	描述
table	创建表格元素
td colspan="number" rowspan="number" headers="header name"	在表格行内创建单元格 该单元格所合并的列数 该单元格所合并的行数 将数据单元格与表头关联起来
th colspan="number" rowspan="number" headers="header name" scope="row col rowgroup colgroup"	与行或列相关联的表头 该单元格所合并的列数 该单元格所合并的行数 将表头与另一个表头相关联 将表头与行、行组、列或列组相关联
tr	在表格内创建一行
caption	为表格创建在浏览器中显示的标题
col	声明一列
colgroup	声明一组列
tbody	标识表的主体行组
tfoot	标识表的表底行组
thead	标识表头行组

第9章 表单

Web最初仅供阅读，但是很快又成为可以在上面完成一些事情的地方——网上购物、订机票、签请愿书、搜索站点、发布tweet……当然（可完成的事情）还不止这些。所有的这些交互都是由表单处理的。

实际上，为了应对从网页到应用程序的转变，HTML 5引入了一些既便于用户填写表单，又易于开发者创建的新的表单控制方式和属性。传统上依赖JavaScript脚本语言的任务可以被标记文本和浏览器自带的功能来完成。HTML 5介绍了许多与表单相关的元素，包括13个新的输入类型和许多新的属性（它们都在本章节末尾的表9-1中列出）。这些功能的某些部分还需要继续完善浏览器才能使用，所以我会注明哪些功能还没有得到普遍支持。

本章介绍的内容包括Web表单、表单如何工作以及创建表单的标记文本。而且我也将简要介绍Web表单设计的重要性。

表单如何工作

工作的表单由两部分组成。第一个部分是你可以在网页上看到的表单本身，它由HTML标记文本所创建。表单是由按钮、文本域和下拉菜单组成的（统称为表单控件），用于收集来自用户的信息。表单还可以包含文本和其他元素。

Web表单的另一个组成部分是服务器端的应用程序或脚本，它们处理由表单收集来的信息，并进行适当的响应。正是它们使表单工作起来的。换句话说，用表单元素展示HTML文档是不够的。Web程序和脚本所需的实际编程知识不在本书的范围之内，但是本章后面的侧栏“让表单工作起来”提供了一些选择，让你能获取需要的脚本。

本章内容

表单如何工作

表单元素

POST与GET的比较

变量和值

表单控制

表单的可访问性功能

略谈编码

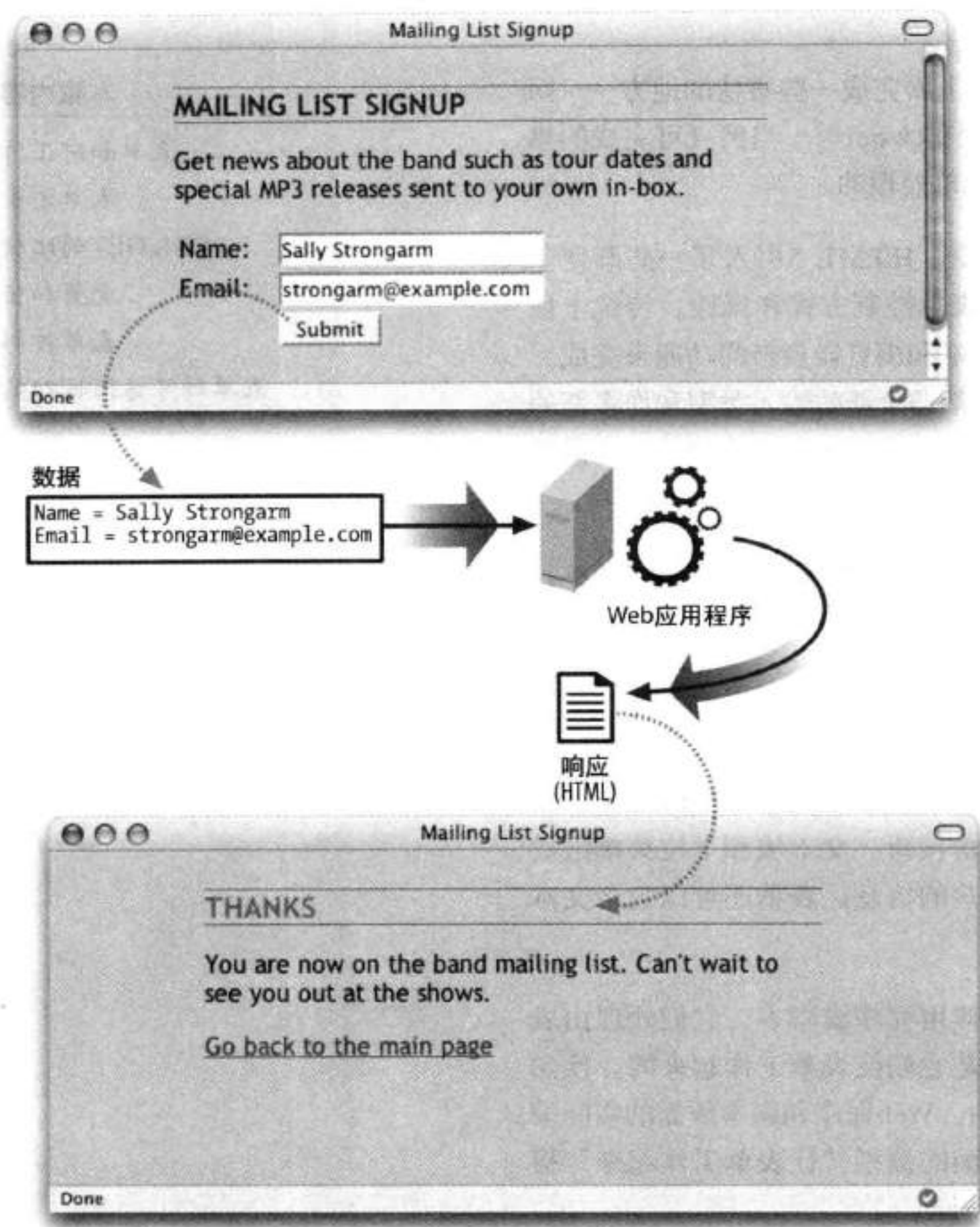
表单数据使用与URL相同的方法编码，其中，空格及其他一些禁用字符将转换为相应的十六进制值。例如，收集来的表单数据中，每个空格字符显示为字符串%20，而斜线(/)则用%2F代替。不用你担心，浏览器会自动处理好。

从数据输入到响应

如果你将要创建Web表单，那么理解后台发生的事是有益的。本例跟踪一个事务的所有步骤，这个事务使用一个简单的表单，为一个邮件列表收集姓名和E-mail地址。虽然很简单，却是表单的典型例子。

1. 访问者（这里可以称为Sally）在浏览器窗口中打开带Web表单的网页。浏览器找到标记文本中的表单控件元素，将其替换为适当的表单控件，包括两个文本输入域和一个提交按钮（如图9-1所示）。
2. Sally想在这个邮件列表中注册，所以她在文本输入域中输入她的名字和电子邮件地址，并单击“提交”按钮提交表单。

图9-1：Web表单提交时的后台行为



3. 浏览器收集她输入的资料，并对其进行编码（见侧栏“略谈编码”），并把它发送给服务器上的Web应用程序。
4. Web程序接收到信息并处理（也就是按编程时设计的处理表单过程进行）。本例中，姓名和E-mail地址被添加到数据库中。
5. Web程序还会返回一个响应。返回响应的种类由内容和表单目的决定。而在本例中，响应是一个简单的包含对申请邮件地址的致谢的网页。其他的程序的响应可能是重新载入HTML表单网页（信息已更新），也可能是将用户转向另一个相关表单网页，还可能是发布错误信息（如果表单填写错误），这里只是举几个例子。
6. 服务器将Web程序的响应发送回浏览器，浏览器显示响应信息。Sally可以看到表单已经工作完毕，而她也就添加到邮件列表中了。

表单元素

`<form>...</form>`

交互式表单

使用form元素来将表单添加到网页中（不要对此感到奇怪）。form元素是所有表单内容的容器，它包括很多表单控件，比如文本输入域和按钮。它可能还包含块元素（比如h1、p和列表），但是，它不能包含其他的form元素。

下面的示例源码文档中包含一个表单，这个表单类似于图9-1中的表单。

```
<!DOCTYPE html>
<html>
<head>
  <title>Mailing List Signup</title>
  <meta charset="utf-8">
</head>
<body>
  <h1>Mailing List Signup</h1>

  <form action="/mailinglist.php" method="post">
    <fieldset>
      <legend>Join our email list</legend>
      <p>Get news about the band such as tour dates and special MP3
releases sent to your own in-box.</p>
      <ol>
        <li><label for="firstlast">Name:</label>
          <input type="text" name="username" id="firstlast"></li>
        <li><label for="email">Email:</label>
          <input type="text" name="email" id="email"></li>
      </ol>
      <input type="submit" value="Submit">
    </fieldset>
  </form>

</body>
</html>
```

除了是表单控件元素的容器，form元素还有一些特性，这些特性在与服务器上表单处理程序交互时很有必要。现在来看看每个特性。

action特性

用于处理表单的程序或脚本（有时称作action页）的位置（URL）由action特性提供。本例中的action特性将数据发送给称作mailinglist.php的脚本。

```
<form action="/mailinglist.php" method="post">...</form>
```

提示

注意不要嵌套form元素，也不要让它们重叠。一个form元素必须在下一个打开前关闭。

注意： 将表单控件包含在HTML语义元素（如列表或者div）中，是目前的最好的用法。正如例子中显示的有序列表，是一种流行的解决方案，但要知道的是，它易于在后面使用样式表来格式化表单，特别是在移动浏览器上。

让表单工作起来

如果你不是一个程序员，不要烦恼。还有别的办法增加表单的交互性。

使用虚拟主机

许多网站的虚拟主机包括访问简单功能的脚本，如留言本、邮件列表等。更高级的虚拟主机甚至可能提供你需要的所有东西，将一个完整的购物车系统添加到你的站点，这些都包含在你每月的主机费用中。还可以获得帮助文档和技术支持人员，帮助你使用虚拟主机。

雇一个程序员

如果你需要制订方案，可能就需要雇一个具备服务器端编程技术的程序员。告诉程序员你想要用表单实现的功能，然后他会提出一个解决方案。然后，你需要确认：在当前的虚拟主机中，你有权在服务器安装脚本，而且服务器支持你所选择的语言。

.php后缀表示表单由PHP脚本语言编写的脚本处理，但网页表单可能使用以下某种技术处理：

- PHP (.php) 是一种开源的脚本语言，最常用于Apache Web服务器。
- 微软的ASP.NET (Active Server Pages, 活动服务器网页) (.asp) 是一种编程环境，在微软互联网信息服务器 (IIS) 上运行。
- Ruby on Rails。Ruby是编程语言，用于Rails平台。许多流行的Web应用程序都用它创建。
- Java服务器网页 (.jsp) 是一种基于Java的技术，与ASP类似。
- Python是一个用于Web和服务器的流行的脚本语言。

还有其他形式处理方式，可能有自己的后缀，或根本没有（这正是Rails平台上Ruby的情况）。请与你的程序员、服务器管理员或由action属性所提供的程序脚本文件的名称和位置确认。

有时，在HTML文件中有嵌入的PHP表单处理代码。在这种情况下，如果将action设为为空，那么表单将自己展示在网页上。

method特性

method特性指定信息将如何发送到服务器。用图9-1中的示例表单采集来的数据作为例子。

```
username = Sally Strongarm
email = strongarm@example.com
```

当浏览器对这些信息进行编码，以传送到服务器时，这些字符串将是这样的（如果你需要复习“编码”知识，请看前面的侧栏）：

```
username=Sally%20Strongarm&email=strongarm%40example.com
```

将编码数据发送到服务器，只有两种方法：POST或GET，可以用form元素中的method特性来指定。方法是可选的，如果省略，将默认为GET。下面将给出这两种方法之间的区别。本例中使用POST方法，如下所示：

```
<form action="/cgi-bin/maillinglist.pl" method="post">...</form>
```

POST 方法

当form元素的方法设置为POST时，浏览器发送独立的服务器请求，请求包括一些特殊的报头和后面的数据。只有服务器才能看到请求的内容，因此，POST是发送保护信息（比如信用卡或其他个人信息）的最好方法。

发送大量数据（比如很长的文本输入），POST方法也是更好的，因为没有GET中的字符长度限制问题。

GET方法

使用GET方法，编码后的表单数据附加到URL中，并发送到服务器。问号标记字符将URL和后面的数据分开，如下所示：

```
get http://www.bandname.com/cgi-bin/maillinglist.pl?name=Sally%20Strongarm&email=strongarm%40example.com
```

如果你想让表单提交的结果（比如一组搜索结果）成为书签，那GET方法就正合适。因为表单的内容很容易看到，所以GET方法不适用于包含个人信息或财务信息的表单。另外，在表单用于上传文件时，GET方法不能使用。

本章中，我们将一直使用更流行的POST方法。既然已经完成了form元素的技术方面的学习，那我们就继续学习表单的另一项实质内容：表单控件。

变量和内容

Web表单使用各种允许用户输入信息或选择选项的控件。控件类型包括各种文本输入域、按钮、菜单和几个特殊功能控件。使用一系列表单控件元素，可以将控件添加到文档中，这些元素将在“重要表单控件综述”中进行分析。

作为一个Web设计师，熟悉控件选项很重要，这样你的表单直截了当而且容易使用。如果知道表单控件在后台做什么，那么也很有帮助。

name特性

表单控件的任务是从用户那儿收集一点信息。在前面几页的表单例子中，文本输入域收集访问者的姓名和E-mail地址。用技术术语来说，“用户名”和“email”是由表单收集的两个“变量（variable）”。而由用户输入的数据（“Sally Strongarm”和“strongarm@example.com”）是变量的值或内容。

name特性表示控件的变量名。在本例中，由textarea元素采取的文本表示为“comment”变量：

```
<textarea name="comment" rows="4" cols="45" placeholder="Leave us a comment."></textarea>
```

当用户在文本域输入一条评论时（“This is the best band ever!”）时，信

注意：POST和GET都是不区分大小写的，通常依照习惯全部大写。然而，在XHTML文档中，method特性的值（*post*或*get*）必须是全部小写字母。

息将以名/值（变量/内容）对的形式传递到服务器，如下所示：

```
comment=This%20is%20the%20best%20band%20ever%21
```

所有的表单控件元素必须包含一个name特性，这样表单处理程序可以将信息分门别类。可以在name特性中包括提交（submit）按钮和重设（reset）按钮元素，但是它们不是必需的因为它们有特殊的功能（提交或重设表单），这些功能与数据收集无关。

命名变量

你不能只是给控件胡乱命名。处理数据的Web程序被设计来寻找特殊的变量名。如果你设计一个表单，与一个先前存在的程序或脚本共同工作，那就需要想出一个特殊的变量名用于表单，这样它们就有共同语言了。你可以从共事的开发者、系统管理员那里得到变量名，或者从服务器上备用脚本提供的用法说明中获得变量名。

如果脚本或程序稍后创建，那么就要保证简单而又描述性的命名变量。另外，每个变量的名称必须是唯一的，也就是说，相同的名称不能用于两个变量。你还应该避免在变量名中放置空格，可以用下划短线或句点代替空格。

我们已经学习了form元素的基础知识，以及变量如何命名。现在，我们可以学习表单标记的实质内容：控件。

重要表单控件综述

这部分很有趣——与添加控件到网页的标记一起使用。本节介绍用于创建下列元素的知识：

- 文本输入控件
- 专门的文本输入控件
- 提交和重置按钮
- 单选按钮和复选框
- 下拉菜单和滚动菜单
- 文件选择和上传控件
- 隐藏域控件
- 日期和时间（HTML 5）
- 数值控件（HTML 5）
- 颜色选择器控件（HTML 5）

我们会在这些控件中暂停一会儿，让你尝试创建图9-2中的调查问卷表格。

如你所见，多数控件是用元素添加到表单中的。元素的功能和外观因type特性而异。在HTML 5中，共有23种不同的元素控件。在这里将会一一介绍。

注意： 在本章结尾表9-1列出了与每个输入类型关联的属性。



The screenshot shows a web browser window with the title "Contest Entry Form". The address bar shows a file path: "file:///Users/jen/Dropbox/LWD4/EXERCISES/LWD4e_chapter09/contest_entry_final.html". The browser has several bookmarks: "popup with tags", "my pinboard", "Account Settings", and "Other Bookmarks".

The form itself is titled **"Pimp My Shoes" Contest Entry Form**. Below the title is a paragraph: "Want to trade in your old sneakers for a custom pair of Forcefields? Make a case for why your shoes have got to go and you may be one of ten lucky winners."

The form is divided into two main sections:

- Contest Entry Information**: This section contains four input fields: "Name:", "Email Address:", "Telephone Number:", and "My shoes are SO old...". The "My shoes are SO old..." field has a placeholder text "No more than 300 characters long".
- Design your custom Forcefields:**: This section contains three sub-sections:
 - Custom Shoe Design**: This section has a label "Color (choose one):" followed by four radio buttons: "Red", "Blue", "Black", and "Silver".
 - Features (Choose as many as you want)**: This section has four checkboxes: "Sparkley laces", "Metallic logo" (which is checked), "Light-up heels", and "MP3-enabled".
 - Size**: This section has a label "Sizes reflect standard men's sizes:" followed by a text input field containing the number "5".

At the bottom of the form are two buttons: "Pimp My Shoes!" and "Reset".

图9-2：本章练习中我们将要创建的内容输入表单

文本输入控件

Web表单中最重要的任务之一是键入文本信息。你用哪种元素取决于需要用户输入单行文本（input）还是多行文本（textarea）。

注意： 本节的标记例子（包括标签label元素）是用来提高可访问性的。本章将在表单可访问性功能中更详细地介绍标签（label），但在此之前，我希望你能习惯当前的表单标记。

单行文本域

```
<input type="text" />  
单行文本输入控件
```

最简单的表单输入类型之一就是文本输入域，它用于输入一个词或一行文字。实际上它是默认输入类型，也就是说，如果你忘记包含type属性，或者使用了不可识别的值，就会自动使用默认类型。使用input元素，并将其type特性设置为text，就可以将单行文本域添加到表单了，如图9-3所示：

```
<li><label>City <input type="text" name="city" id="form-city"  
value="Your Hometown" maxlength="50"></label></li>
```

在这里，我想指出几个属性。

name

name特性是用来识别变量名的。

value

value特性是表单被载入的时候，在输入域中出现的默认文本。当你重设表单时，输入域重现这个值。

```
<textarea>...</textarea>  
多行文本输入控件
```

maxlength

默认情况下，不管输入域的尺寸是多少，用户可以在其中输入有限数量的字符（如果文本超过输入框的字符宽，输入框就会向右滑动再显示）。如果你正在使用的表单处理程序需要限制最大字符数，可以使用maxlength特性来设置。

多行文本输入域

有时，你会希望你的用户能够输入超过一行文字。在这种情况下，可以使用textarea元素，在浏览器显示时被替换为一个多行、可滚动的文本输入框（图9-3）。

不同于空的input元素，你可以在textarea元素的开始和结束标签之间放置内容。当表单在浏览器上显示的时候，textarea元素的内容将会显示在文本框中。当表单被提交时，它也将被发送到服务器，因此需要仔

注意： 表单控件的具体显示风格因操作系统和浏览器版本而异。

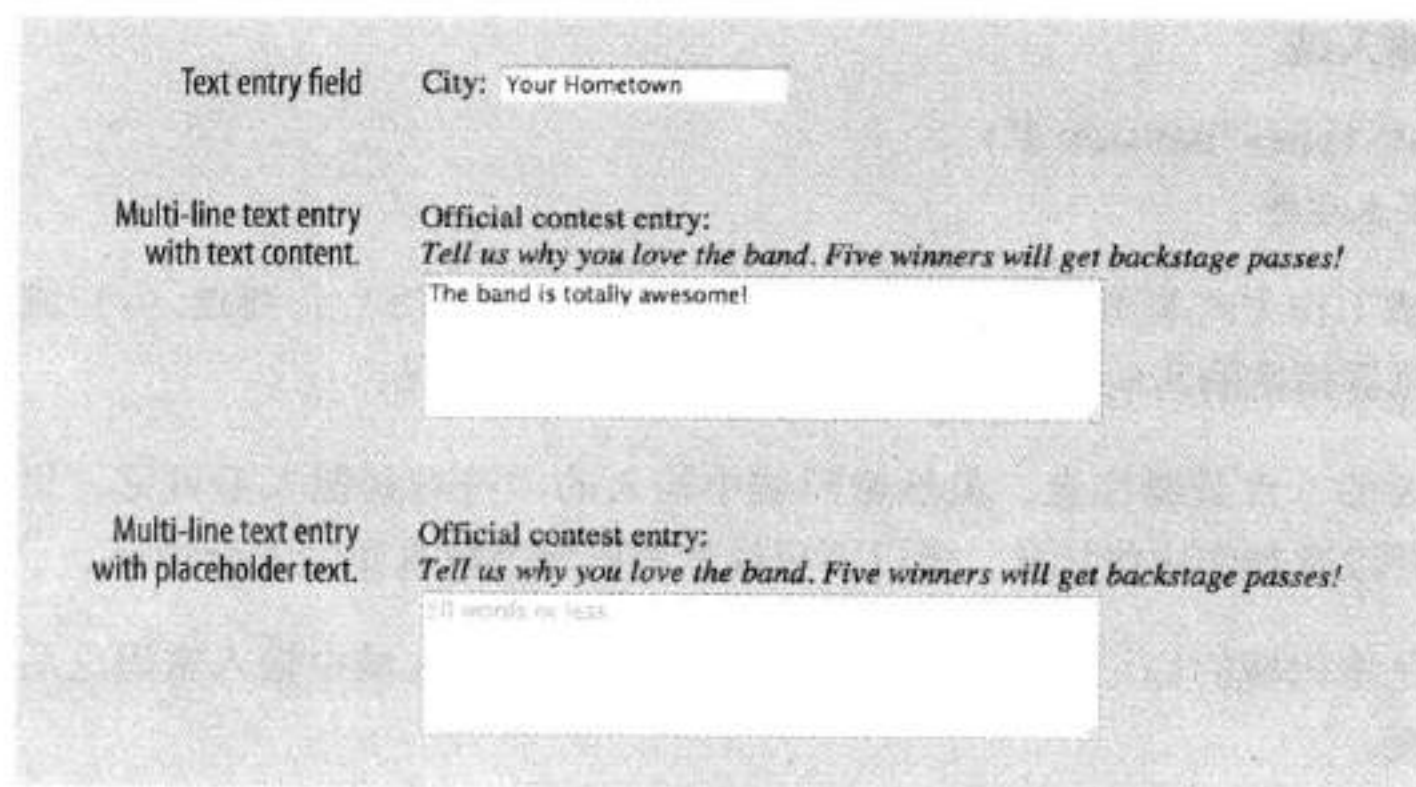


图9-3: Web表单输入文本控件的例子

细考虑到底提交了什么。对于开发人员来说,很少会在开始和结束标签之间什么都不添加,可以通过`title`和`placeholder`属性来提示应该填写什么。新的HTML 5 `placeholder`属性可以用于`textarea`和其他基于文本的`input`类型,并可以用于提供一个简短的提示。在写本书的时候,Android系统,旧版本的Firefox(版本早于3.6)和IE都不支持该属性。

```
<p><label>Official contest entry <br>
<em>Tell us why you love the band. Five winners will get backstage
passes!</em><br>
<textarea name="contest_entry" rows="5" cols="50">The band is totally
awesome!</textarea></label></p>

<p>Official contest entry:<br>
<em>Tell us why you love the band. Five winners will get backstage
passes!</em><br>
<textarea name="contest_entry" placeholder="50 words or less">
</textarea>
</p>
```

`rows`和`cols`属性是一种用于限定`textarea`大小的方法,但更常用的方法是使用CSS。`rows`限定文本区域显示的行数,`cols`指定字符数的宽度。如果用户输入的文本超出所分配的空间,那么就会出现滚动条。

在本例中还有几个属性。`wrap`属性指定提交的文本是否应保留换行。`soft`(默认值)不保留换行符,而`hard`会保留。`maxlength`属性(新HTML 5)限制了可以键入到字段中的字符数量。

专用的文本输入域

除了通用单行文本输入,还有一些输入类型输入特定类型的信息,如密码、搜索条件、电子邮件地址、电话号码和URL。

disabled和readonly

`disabled`和`readonly`特性可以添加到任何表单控件元素中,来阻止用户选中这些控件。当一个表单元素被设置为`disabled`后,这个元素就不可选了。可视化浏览器可能默认将这个控件显示为灰色淡出(当然,你可以使用CSS更改)。`disabled`状态只能用脚本改变。这是一个有用的特性,可以限制对表单前面的一些基于数据输入的表单域的访问。

`readonly`特性可以阻止用户改变表单控件的值(虽然可以被选中)。它让开发者可以根据表单前面其他的数据输入,来使用脚本设置控件的值。`readonly`输入在显示外观上与其他输入是不同的,这样用户就不会试图修改它们的值。

警告: 在撰写本文时,iOS忽视`option`元素上的`disabled`特性(iOS 5以及更早版本)。

密码输入域

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

密码文本控件

密码域工作方式类似于文本输入域，除了使用星号（*）、圆点（.）或者浏览器指定的其他字符代替所有字符，使字符不可见。

很重要的一点是要注意，虽然密码域中输入的字符对其他人不可见，但表单并没有加密这些信息，所以不应该把密码域当做真正的安全措施。

下面是密码域的标记的例子。图9-4显示了用户在输入域中输入密码之后的外观。

```
<li><label for="form-pswd">Log in:</label><br>
<input type="password" name="pswd" maxlength="8" id="form-pswd" ></li>
```

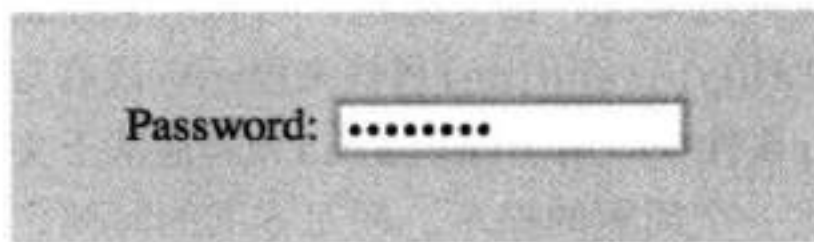


图9-4：密码被转换为在浏览器中显示的圆点

HTML 5文本输入

```
<input type="search">
```

搜索域

```
<input type="email">
```

电子邮件地址

```
<input type="tel">
```

电话号码

```
<input type="url">
```

地址 (URL)

在HTML 5之前，收集电子邮件地址、电话号码、网址或搜索项的唯一方法是插入一个通用的文本输入域。在HTML 5中，email、tel、url和search输入类型使浏览器能够知道输入的到底是什么信息。这些新的输入类型使用前面所述的通用文本输入类型相同的属性（name、maxlength、size和value），当然还有一些新的HTML 5特性。

所有的这些输入类型通常显示为单行文本输入。但是支持它们的浏览器可以基于其他的语义信息来做一些有趣的事情。例如，iOS上的Safari对输入类型提供了一个配套的键盘，如为search输入类型提供了“搜索”按钮，为url类型提供了“.com”的按钮（图9-5）。浏览器通常会在搜索域中添加一个一键“清空”图标（通常样子看起来是一个小X）。一个支持的浏览器可以检查用户的输入，判断是否有效，如确保email中输入的文字是标准的电子邮件地址结构（在过去，你需要使用JavaScript来验证）。例如，如果输入不匹配预期的格式，Opera（图9-6）和Chrome浏览器会显示一个警告。

并非所有的浏览器都支持新的HTML 5输入类型，即便支持也并非以同样的方式，但好消息是，如果无法识别某类型，就会显示默认的工作良好的通用文本输入。既然这是一个进步，我们没有理由不立即开始使用它们。



图9-5: iOS上的Safari在输入类型的基础上提供了自定义的键盘

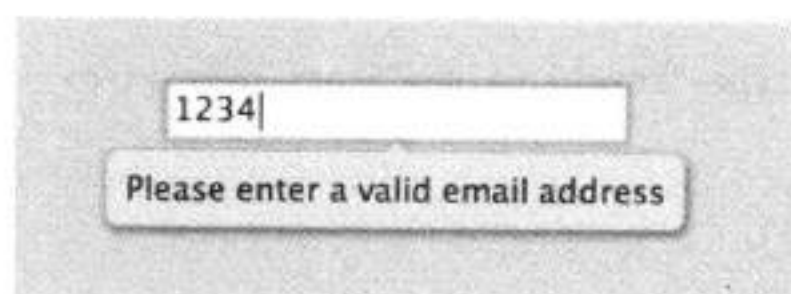


图9-6: 当输入不匹配期望的电子邮件地址格式时, Opera浏览器会验证并显示一个警告

datalist元素

新的HTML 5中的datalist元素允许创作者为任意的文本输入类型提供一些包含建议值的下拉菜单。它为用户提供了一些可选择的快捷键, 但如果没有被选中, 用户可以输入自己的文本。在datalist元素中, 建议值被标记为选项元素。使用input元素的list属性与其各自的datalist控件的id相连。

在下面的例子中(图9-7), datalist为文本输入提出了几种教育水平选项。

```
<p>Education completed: <input type="text"
  list="edulevel" name="education">

<datalist id="edulevel">
  <option value="High School">
  <option value="Bachelors Degree">
  <option value="Masters Degree">
  <option value="PhD">
```

```
</datalist>
```

在撰写本文时, 只有Opera浏览器已实现了datalist元素, 其他浏览器会忽略它, 并提供一个简单的文本输入, 这是完全可以接受的。你也可以使用JavaScript来创建datalist控件功能(比如polyfill)。

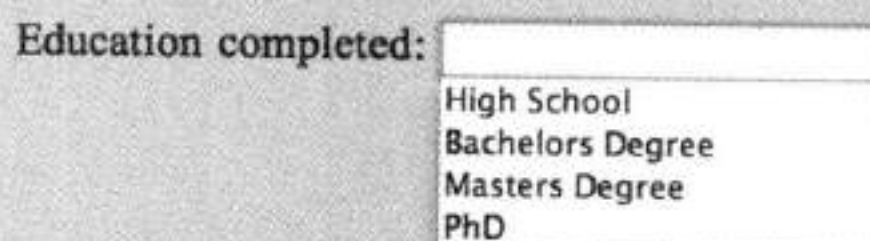


图9-7: datalist为文本输入域创建一个建议值的弹出式菜单


```
<input type="submit">
```

提交表单数据到服务器

```
<input type="reset">
```

重置表单控件为其默认设置

更多按钮

还有少数自定义按钮元素，初学者在学习过程中很少遇到它们。但为了讲述得更完整，我们把它们放到侧栏里介绍。

图像按钮

```
<input type="image">
```

input控制的这种类型允许用你选择的图像替换提交按钮。图像将平面显示，而不像三维按钮。不幸的是，这种类型的按钮有访问性的问题，所以必须要包含一个alt值。

自定义输入按钮

```
<input type="button">
```

设置input元素的类型为“button”，创建可以用脚本语言（比如JavaScript）自定义的按钮。不像提交按钮和重置按钮有预定的功能，它的功能需要后期设定。

button元素

```
<button>...</button>
```

button元素是一个灵活的元素，用于创建类似于input元素创建的自定义按钮。button元素的内容（文本或/和图像）显示在按钮上。

如果想了解使用button元素的更多信息，请阅读Aaron Gustafson在digital-web.com/articles/push_my_button发布文章“Push My Button”。

提交和重置按钮

Web表单中可以添加很多不同种类的按钮。最基本的是提交按钮。当单击时，提交按钮立即将收集到的表单数据发送给服务器处理。重置按钮将所有表单控件返回到它们刚载入时的状态。换句话说，复位表单并不简单地清除所有的域。

提交和重置按钮用input元素添加。正如前面提到的，因为这些按钮具有不包括输入数据的特点，所以它们是唯一不需要name属性的表单控件元素，当然如果你需要，添加一个也是可以的。

提交和重置按钮可以直接使用。只要将它们放置在适当的地方，在大多数情况下是在页面的最末尾。默认情况下，提交按钮的标签为“Submit”或“Submit Query”，重置按钮标记为“Reset”。可以使用value属性来更改按钮上的文字，正如在这个例子中显示的复位按钮（图9-8）中所示：

```
<p><input type="submit"> <input type="reset" value="Start over"></p>
```

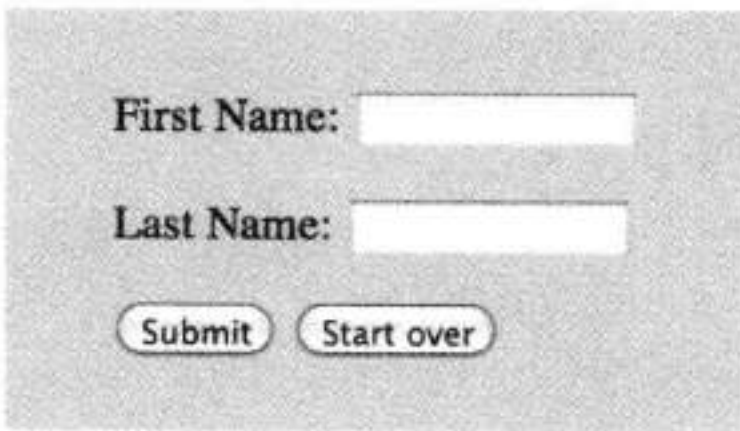


图9-8：“提交”和“重置”按钮

表中的重置按钮不如以前用得那么频繁了。这是因为随着技术的发展，我们使用JavaScript来检查表格输入的有效性。所以，用户可以在进行中实时获得反馈。由于设计的周全，几乎没有用户需要浏览到表格的最下面来重置所有的东西。

现在你应该已经足够了解表单标记了，可以开始建立如图9-2所示的调查问卷了。练习9-1将引导你完成第一步。

练习9-1 开始比赛表单

下面是场景描述。你是一个Web设计师，负责为Forcefield Sneakers的“Pimp My Shoes”比赛创建输入表单。复制编辑器已经交给你一个表单内容的草图（图9-9），并且完成了一些控件的工作方式的注释。程序员在处理你需要用到的脚本和变量名信息时，会留下一些固定的注释。

你所面临的挑战是要将草图变成功能完整的在线表单。我将给你开个头，创建一个只包含文本内容和最少的标记和样式的非常精简的文档。这个文档 `contest_entry.html` 可以从 www.learningwebdesign.com/4e/materials 在线获取。如果你想检查你的工作，可以从附录A中找到整个完成后的表单的源码。

“Pimp My Shoes” Contest Entry Form

Want to trade in your old sneakers for a custom pair of Forcefields?
Make a case for why your shoes have got to go and you may be
one of ten lucky winners.

Contest Entry Information

Name:
Email:
Phone:

这个表单会通过POST方法发送到
`http://www.learningwebdesign.com/
contest.php`
分别把这些文本域命名为“Name”、
“email”、“Phone”和“Story”。

My shoes are SO old...

No more than 300 characters long

添加占位符文本

Design your custom Forcefields:

Custom shoe design

Color (choose one):

- ☐ Red
- ☐ Blue
- ☐ Black
- ☐ Silver

把本节的控件分别命名为“Color”、
“features[]”和“size”。注意“features”
后面的方括号[]是为了便于脚本正确处理

Features (choose as many as you want):

- ☐ Sparkley laces
- ☒ Metallic logo 确保默认选中 Metallic logo
- ☐ Light-up heels
- ☐ M P3-enabled

Size

(Sizes reflect standard men's sizing):

5

从下拉菜单的13个数值
中选5

Pimp My Shoes!

Reset

改变提交按钮的文本

图9-9：比赛输入表单的草图

1. 在文本编辑器中打开 `contest_entry.html`。
2. 我们要做的第一件事是，将 `intro` 段后面的所有内容放到一个 `form` 元素中。程序员留下注释，指定用于这个表单中的 `action` 和 `method` 特性。`form` 元素结果如下：

```
<form action="http://www.learningwebdesign.com/contest.php"
method="post">
...
</form>
```

3. 本练习中，我们将操作表单的“Contest Entry Information”部分。现在我们准备添加头三个短的文本输入表格控件。下面是第一个，另两个你自己插入。

```
<li>Name: <input type="text" name="username" ></li>
```

提示：为每个输入域选择最合适的输入类型。确保以程序员注意事项中指定的方式命名输入元素。

4. 现在，在新段里，给鞋子介绍添加一个多行文本区域。由于我们并没有为这个表单提供样式表，所以需要使用标记，使它有4行长60字符宽（在实际使用中，推荐使用CSS，因为CSS功能更为全面）。

```
<li>My shoes are SO old...<br>
<textarea name="story" rows="4" cols="60"
maxlength="300" placeholder="No more than 300
characters long"></textarea></li>
```

5. 我们将暂时跳过表单的剩余部分，直到我们再掌握几个控件。但最后我们还能添加提交和重设按钮，它们就在 `</form>` 标签前面。注意，我们需要改变提交按钮上的文字。

```
<p><input type="submit" value="Pimp
my shoes!">
<input type="reset"></p>
</form>
```

6. 现在，保存文档并在浏览器中打开。完成后将产生图9-3的效果。如果没有出现这样的结果，就说明你还有一些工作要做。

看起来不错吧！你可以输入一些信息并提交表单，实际运行一把。你将收到如图9-10中的响应（是的，`contact.php`居然工作了，但对不起，内容是编造的。）

图9-10：如果你的表单正常运转，你应该看到一个像这样的响应页面

THANK YOU

Thank you for entering the Forcefield Sneaker "Pimp My Shoe" contest. We have received the following information with your entry:

About you:

Name: Jennifer Robbins
Email Address: jen@oreilly.com
Telephone Number: 555.555.1212
Sad shoe story: My shoes have no soul.

Your shoe design (if you win)

Sorry, we did not receive your information.

单选和复选框按钮

复选框和单选按钮都使访问者的选择过程变得简单。在某些功能方面它们是类似的，比如，用户可以选的切换开关，以及使用input元素。但是它们提供不同的功能。

组合中只允许一个被选中时，或者换句话说，当选项互斥（比如是或否、男或女）时，适合使用由一组单选按钮组成的表单控件。当一个单选按钮是“开”的时候，其他所有按钮都必须“关”，有几分像以前收音机按钮的工作方式：按下一个按钮，其他的就会弹出。

然而，当复选框组合在一起时，组合中你想选多少就可以选多少。在可以选择多个选项列表中，复选框是正确的选择。

单选按钮

使用input元素，并将type特性设置为radio，就可以将单选按钮添加到表单中了。下面是最精简的单选按钮的语法：

```
<input type="radio" name="variable" value="value">
```

Name特性是必须的而且在绑定多个单选输入中扮演了很重要的角色。当你给了一些单选按钮一样的name值时（下面例子中的age），它们就形成了一群互斥的选项。

本例中，单选按钮用作用户输入年龄组的界面（一个人不能属于多个年龄组，所以单选按钮是正确的选择）。图9-11展示了单选按钮是如何显示在浏览器中的。

```
<p>How old are you?</p>
<ol>
  <li><input type="radio" name="age" value="under24" checked> under
  24</li>
  <li><input type="radio" name="age" value="25-34"> 25 to 34</li>
  <li><input type="radio" name="age" value="35-44"> 35 to 44</li>
  <li><input type="radio" name="age" value="over45"> 45+</li>
</ol>
```

注意，所有的input元素变量名都相同（都是“age”），但它们的值不同。因为这些都是单选按钮，一次只有一个按钮会被选中，因此，当表单提交时只有一个值会被发送给服务器处理。

注意：在单选按钮、复选框以及菜单的代码例子中，为了使得标记结构尽可能简洁明了，我省去了fieldset和label元素。在即将讲述的《表单访问性》中，你将会了解为什么在所有的表单元素标记中包含这些元素很重要。

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

单选按钮

注意：在XHTML文档中，checked特性的值一定要显式地设置为checked，如例子中所示。

```
<input type="radio" name="foo"
checked="checked" >
```

在HTML文档中，你不需要写出checked特性的值。它可以简化，如下：

```
<input type="radio" name="foo"
checked>
```

练习9-2 添加单选按钮和复选框

比赛入口表单的两个问题是，使用单选按钮和复选框来进行选择。打开contest_entry.html并按下面的步骤来做。

1. 在定制鞋的设计部分，有颜色和功能选项列表。因为鞋子只可能是一种颜色，所以Color选项应该为单选按钮。把单选按钮插入每个选项之前。下面的例子是剩下的Color选项。

```
<li><input type="radio"
name="color" value="red">
red</li>
```

2. 同对Color所做的一样，你标记Feature的选项。然而这次，type特性值为checkbox。确保每个变量名都为[features]，同时如同草图中注释的，metallic logo被预先选中。

3. 保存文档，在浏览器中打开，检查并确认你的作品外观正常，然后提交表单，确保功能也正常。

Radio buttons	Checkbox buttons
How old are you?	What type of music do you listen to?
<input checked="" type="radio"/> under 24 <input type="radio"/> 25 to 34 <input type="radio"/> 35 to 44 <input type="radio"/> 45+	<input checked="" type="checkbox"/> Punk rock <input checked="" type="checkbox"/> Indie rock <input type="checkbox"/> Hip Hop <input type="checkbox"/> Rockabilly

图9-11：单选按钮（左）适用于只允许一个选择的情况。当用户选择任意多个选项时，从不选到全选，复选框（右）是最好的选择

表单载入后，给input元素添加checked特性，你可以决定哪个按钮被选中。本例中，“under 24”的按钮将会被默认选中（参见注意）。

复选框

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

复选框

使用input元素，设置其类型为checkbox，就可以添加复选框了。同单选按钮一样，给一些复选框分配相同的name特性值，你就创建了一个复选框组。如我们前面所提醒的，不同之处在于，一次可以选中多个复选框。在表单提交的时候，每个选中的值都会发送到服务器。下面是一个复选框组的例子，用来确定音乐方面的兴趣。图9-11显示了它们在浏览器中的外观。

```
<p>What type of music do you listen to?</p>
<ul>
  <li><input type="checkbox" name="genre" value="punk" checked> Punk
  rock</li>
  <li><input type="checkbox" name="genre" value="indie" checked> Indie
  rock</li>
  <li><input type="checkbox" name="genre" value="hiphop"> Hip Hop</li>
  <li><input type="checkbox" name="genre" value="rockabilly">
  Rockabilly</li>
</ul>
```

当然，复选框不一定必须用在组中。本例中，一个复选框用于允许访问者参加特殊促销活动。如果用户选中了这个复选框，将只有这个控件的值传递到服务器。

```
<p><input type="checkbox" name="optin" value="yes"> Yes, send me news
and special promotions by email.</p>
```

在练习9-2中，你将有机会将单选按钮和复选框添加到比赛入口表单中。

菜单

提供一组选择还有一个替代方案，把它们放到下拉菜单或滚动菜单中。菜单往往比一组单选按钮或复选框更简洁。

使用select元素将下拉菜单和滚动菜单添加到表单中。菜单是下拉还是滚动，取决于你指定的size特性，以及你是否允许选中多个选项。

下面来看这两种菜单。

下拉菜单

当没有指定size特性，或者size特性设置为1时，select元素默认显示为下拉菜单。在下拉菜单中，只能选择一项。下面举例说明（如图9-12所示）：

```
<p>What is your favorite 80s band?
<select name="EightiesFave">
  <option>The Cure</option>
  <option>Cocteau Twins</option>
  <option>Tears for Fears</option>
  <option>Thompson Twins</option>
  <option value="EBTG">Everything But the Girl</option>
  <option>Depeche Mode</option>
  <option>The Smiths</option>
  <option>New Order</option>
</select>
</p>
```

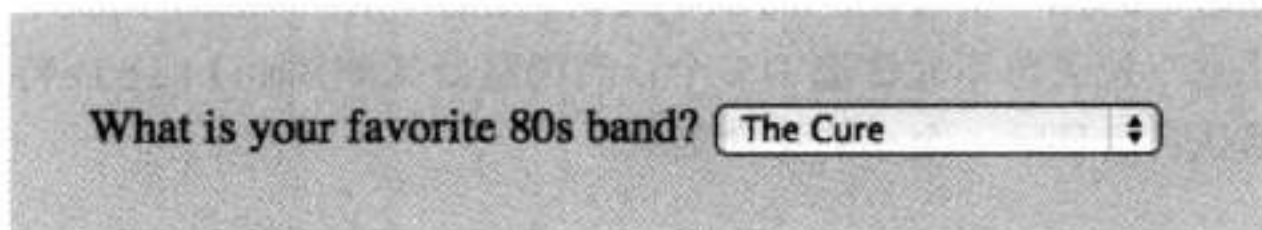


图9-12：当用户单击箭头或横条的时候，下拉菜单会弹出

你可以看到，select元素只是很多option元素的容器。当表单提交时，被选中的option元素的值将被传递到Web程序。如果由于某种原因，你想发送一个没在菜单中显示的不同的值，那就使用value特性来提供一个覆盖值。例如，如果某人从示例菜单中选择了“Everything But the Girl”，那么表单将为“EightiesFave”变量提交值“EBTG”。对于其他人，在选项标签之间的内容将作为值被发送。

你可以做一个菜单，使它像练习9-3中选择鞋子尺码的那个菜单一样。

<select>...</select>

菜单控件

<option>...</option>

菜单中的选项

<optgroup>...</optgroup>

菜单中的逻辑选项组

滚动菜单

使用`size`特性，指定你想看到的行数，就可以使用菜单显示为滚动列表。本例中的菜单与前一个菜单的选项相同，只是它被设置显示为滚动列表，而且是6行高（图9-13）。

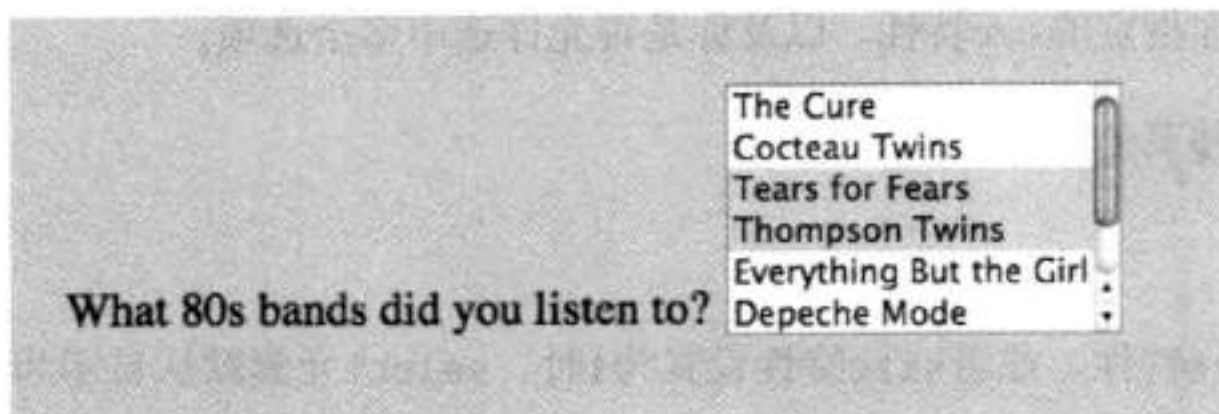


图9-13：一个滚动菜单选择多个选项

```
<p>What 80s bands did you listen to?
<select name="EightiesBands" size="6" multiple>
  <option>The Cure</option>
  <option>Cocteau Twins</option>
  <option selected>Tears for Fears</option>
  <option selected>Thompson Twins</option>
  <option value="EBTG">Everything But the Girl</option>
  <option>Depeche Mode</option>
  <option>The Smiths</option>
  <option>New Order</option>
</select>
</p>
```

你可能注意到这里的几个新特性。`multiple`特性允许用户从滚动列表中选中多项。注意下拉菜单不允许选中多个；当浏览器探测到`multiple`特性时，就会默认自动显示一个小的滚动菜单。

在一个`option`元素中使用`selected`特性，使这个选项成为菜单控件的默认值。表单载入时，被选中的选项会高亮显示。`selected`特性也可用于下拉菜单。

组合菜单选项

你可以使用`optgroup`元素，创建概念上的选项组。`optgroup`元素中必须用`label`特性为选项组提供标题。图9-14展示了选项组在现代浏览器中的显示。

```
<select name="icecream" size="7" multiple>
  <optgroup label="traditional">
    <option>vanilla</option>
    <option>chocolate</option>
  </optgroup>
  <optgroup label="fancy">
    <option>Super praline</option>
    <option>Nut surprise</option>
```

注意： `option`元素中的`label`属性不同于用于改善易用性的`label`元素（本章后面将讨论）。

```

    <option>Candy corn</option>
  </optgroup>
</select>

```



图9-14: 呈现在一个现代浏览器中的选项组

练习9-3 添加菜单

添加到比赛入口表单中另外一个的控件是下拉菜单，它用于选择鞋子尺码。

1. 将一个带有鞋子尺码（5~13）的选择菜单元素插入表单中。

```

<p>Size (sizes reflect men's sizing):
  <select name="size" size="1">
    <option>5</option>
    ...insert more options here...
  </select>
</p>

```

2. 保存文档并在浏览器中查看。你还可以提交表单，确保表单正常工作。你应该会获得一个致谢回复网页，上面列出了你在表单中输入的所有信息。

恭喜！你已经创建了第一个能工作的Web表单。在练习9-4中，我们将添加标记，使得在辅助设备上更易于访问。但首先，我们要介绍更多的控件类型。

文件选择控件

Web表单不仅能收集数据，还能用于传送来自用户硬盘的外部文档。例如，印刷公司可以利用Web表单接收关于名片定单的美术作品。杂志社也可以在他们的网站上使用表单，收集照片比赛用的数码照片。

文件选择控件使用户能够选择硬盘中的文档，并随数据一起提交。使用input元素，设置type特性为file，就可以添加到表单中了。

下面的示例代码和图9-15展示了用于照片的文件选择控件。

```

<form action="/client.php" method="POST" enctype="multipart/form-data">
  <label>Send a photo to be used as your online icon
  <em>(optional)</em><br>
  <input type="file" name="photo" size="28"></label>
</form>

```

```

<input type="file" />
文件选择域

```


文件上传控件因浏览器和操作系统的不同而不同。它可能是一个有按钮的文本字段可以浏览硬盘驱动器，如Firefox（图9-15，顶部）或它可能只是Chrome浏览器展示的一个按钮（底部）。

注意到这点很重要：如果表单包含一个文件选择输入元素，那么你必须指定表单的编码方式（`enctype`）为`multipart/form-data`，而且使用POST方法。如果浏览器显示，那么例子中的`size`属性是用来设置文本域的字符宽的特性的（尽管它也可以被CSS规则控制）。

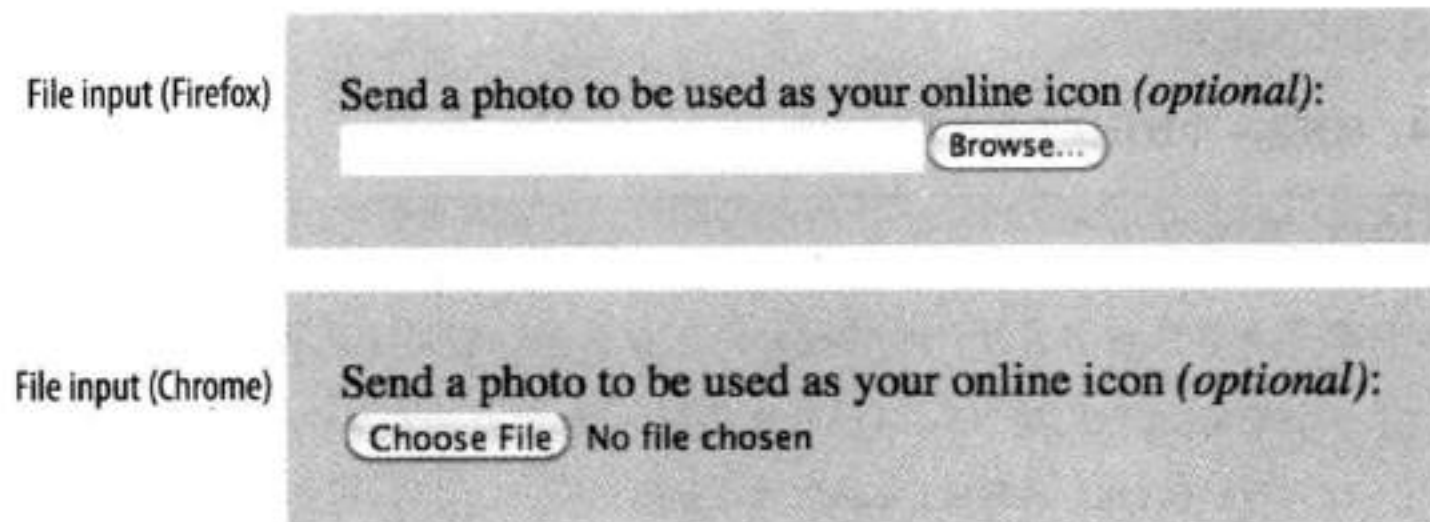


图9-15：一个文件选择表单域

隐藏控件

`<input type="hidden">`
隐藏控制域

可能有时，你需要将并非来自用户的信息，发送给表单处理程序。在这种情况下，你可以使用一个隐藏表单控件，在表单提交时它发送数据，但是当表单显示在浏览器上时，它是不可见的。

隐藏控件是由元素添加的，使用type设置为hidden。当表被提交时，它的唯一目的是传递名称/值给用户。在这个例子中，当交易完成时，一个隐藏表格元素被用来提供文件thank-you的位置。

警告：用户可以访问和操纵隐藏的表单控件。如果你要成为一个专业Web开发人员，你需要学习如何编程以避免这样的事情。

```
<input type="hidden" name="success-link" value="http://www.example.com/littlechair_thankyou.html">
```

我曾经在制定出用户最终填的表之前和有许多隐藏控件打过交道。这就是你从程序员、系统管理员或者其他处获得的信息。如果你有一稿数用的脚本，一定要记得检查是否所有的隐藏控件都是必需的。

日期和时间控制（HTML 5）

如果你曾经预订酒店或机票，你绝对使用过一个小日历选择日期控件。小日历很可能使用JavaScript创建。HTML 5引进了6个新的输入类型，日期和时间选择控件是其中一部分，使用浏览器标准的显示（就像他们可以显示复选框、弹出菜单和其他部件）。在写本书时，只有少数浏览器可以实现日期和时间选择器，如Opera浏览器，图9-16所示，但在不支持的浏览器上，对于日期和时间输入类型，只是使用文本输入域来替代。

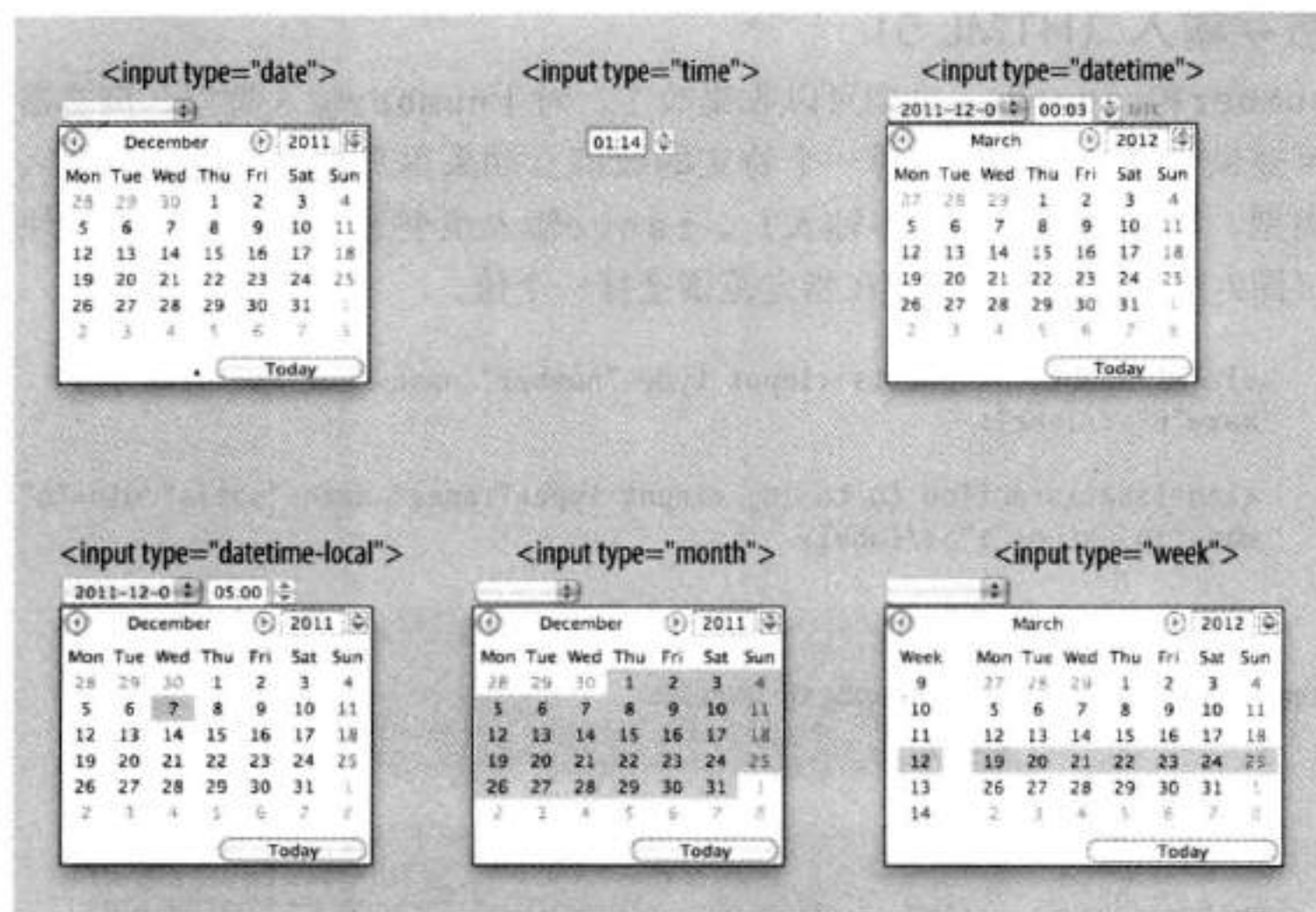


图9-16：在Mac OS 中的opera11日期和时间选择器输入

新的日期和时间相关的输入类型如下：

`<input type="date" name="name" value="2004-01-14">`

创建日期输入控件，如弹出式日历，用于指定日期（年，月，日），必须在ISO日期格式提供（YYYY-MM-DD）其初始值。

`<input type="time" name="name" value="03:13:00">`

创建时间输入控件以指定时间（小时，分，秒，小数部分），没有时区。设置该值为hh:mm:ss的格式。

`<input type="datetime" name="name" value="2004-01-14T03:13:00-5:00">`

创建包括时区信息的日期/时间输入控件。该值是一个ISO格式的与GMT相关的日期和时间，就像我们在第5章所看到的time元素一样（YYYY-MM-DDThh:mm:ssTZD）。

`<input type="datetime-local" name="name" value="2004-01-14T03:13:00">`

创建没有时区信息的日期/时间输入控件（YYYY-MM-

`<input type="date">`

日期输入控件

`<input type="time">`

时间输入控件

`<input type="datetime">`

与时区有关的日期/时间控制

`<input type="datetime-local">`

没有时区的日期/时间控制

`<input type="month">`

指定一年的某一个月

`<input type="week">`

指定一年的某个星期

DDThh:mm:ss)。

```
<input type="month" name="name" value="2004-01">
```

创建日期输入控件指定一年中特定的一个月 (YYYY-MM)。

```
<input type="week" name="name" value="2004-W2">
```

创建日期输入控件指定一年的某一周，使用ISO的周格式 (YYYY-W#)。

```
<input type="number">
```

数字输入

```
<input type="range">
```

滑动输入

数字输入 (HTML 5)

`number`和`range`输入类型可以收集数字。对于`number`输入类型，浏览器可提供一个小控件来选择一个特定的数值（如果用户端不支持这种输入类型，可以显示一个文本输入）。`range`输入类型通常显示为一个滑块（图9-17），它允许用户在指定范围选择一个值。

```
<label>Number of guests <input type="number" name="guests" min="1"
max="6"></label>
```

```
<label>Satisfaction (0 to 10) <input type="range" name="satis" min="0"
max="10" step="1"></label>
```

```
<input type="number">
```

Number of guests:

```
<input type="range">
```

Satisfaction (from 0 to 10):

图9-17: HTML 5输入类型`number`和`range` (Mac OS X中Opera11的显示)

`number`和`range`的输入类型接受`min`和`max`属性来指定最小和最大允许值（浏览器可以检查用户输入是否符合约束）。`min`和`max`是可选的，你也可以只设置它们中的一个。

`step`属性允许开发人员指定数字输入时可接受的增量。默认值是1。如果`step`值设置为0.5将允许值1、1.5、2、2.5等出现；值设置为100将允许出现100、200、300等。你还可以将`step`属性设置为可以显示的接受任何数值增量。

同样地，如果浏览不支持这些新的输入类型，将显示一个标准的文本输入域，这种妥协也是个办法。

颜色选择器（HTML 5）

这样做的目的是创建一个弹出拾色器，可以直观地选择一个颜色值，就像操作系统或图像编辑程序中使用的那样。值是十六进制RGB值（#RRGGBB）。图9-18显示了在Opera11中的颜色选择器窗口。不支持该控件的浏览器会显示默认的文本输入。

```
<label>Your favorite color <input type="color" name="favorite"></label>
```

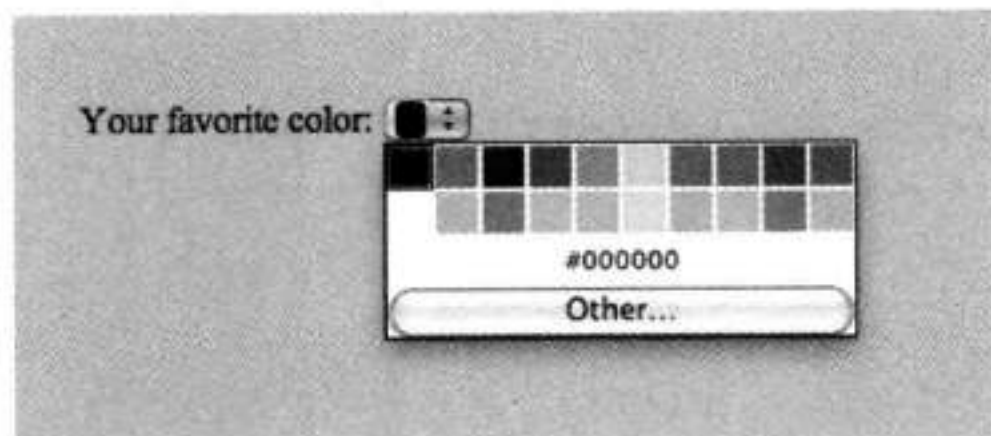


图9-18：颜色输入类型（在Mac OS X的Opera11中）

这包涵了表单控制综述。学习如何插入表单控件是表单制作的一部分，任何称职的Web开发人员都会花时间，以确保表单是否可以访问。幸运的是，我们可以做一些事情来描述表单的结构。

`<input type="color">`

颜色选择器

更多关于HTML 5表单元素的信息

为了完整起见，让我们来看看在HTML 5中的其余表单新元素。撰写本文时，由于难以言说的原因，它们都缺乏浏览器的支持，所以你要等待一段时间以将其添加到你的HTML工具箱。我们已经讲述了为文本输入提供建议值的datalist元素。HTML 5还引入了以下元素：

progress

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

表示状态的一个持续的过程

progress元素为用户提供了持续状态的进展信息，如文件下载。它可以有一个特定的最终值（max属性提供）或者只是表示某些事情正在发生（如等待服务器响应）。

Percent downloaded: `<progress max="100" name="fave">0 </progress>`

meter

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

表示状态的一个持续的过程

meter类似于progress，但它总是代表在一个已知的测量值范围内（也称为测量仪表）。它有一些属性：min和max，表示范围内的最低值和最高值，low和high的可以在不希望的水平触发警告，opimum属性指定首选值。在过程中，这些值很可能被JavaScript动态更新。

`<meter min="0" max="100" name="download">50%</meter>`

output

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

计算输出值

简单地说，output元素提供了一种方法来表示与输入相关的脚本或程序的计算结果，并且与影响计算的输入相关联。

keygen

`<keygen>`

密钥对生成器

keygen元素代表了一个密钥对（用于确保隐私）控制。当表单被提交时，私钥存储在本机，公钥被打包和发送到服务器。不要担心，我对这意味着什么也有些迷茫。你可以阅读有关的公共密钥加密（en.wikipedia.org/wiki/Public-key_cryptography），并在你弄清楚时解释给我听。

表格的可访问性

考虑无法查看可视浏览器的用户如何理解和浏览你的网页表单是一个很重要的问题。label、fieldset和legend等表单元素可以通过将表单中的各个部分和语义联系起来以提供可访问性。这样标记不仅是语义丰富，而且有更多的元素可以与样式表“挂钩”。这是对大家都好处的！

label

虽然我们在浏览器的地址文本输入栏旁边可以看到“地址”标签，但是在源代码中，标签和域输入需要分开。label元素与它的相应表单域是联系在一起的，这为使用基于语音的浏览器的用户提供了重要环境。

每个label元素都与某一个表单控件有联系。有两种方法来使用它。一个方法，称为隐式关联，在一个label元素中嵌套表单控件和它的描述。在下面的例子中，label被分配到单个复选框以及与其相关的文字说明。（顺便说一下，这是一种标记单选按钮和复选框的方法，你无法将一个标签指定到整个组。）

```
<ul>
  <li><label><input type="checkbox" name="genre" value="punk"> Punk
  rock</label></li>
  <li><label><input type="checkbox" name="genre" value="indie"> Indie
  rock</label></li>
  <li><label><input type="checkbox" name="genre" value="hiphop"> Hip
  Hop</label></li>
  <li><label><input type="checkbox" name="genre" value="rockabilly">
  Rockabilly</label></li>
</ul>
```

另一种方法，称为显式关联，将标签与控件的id引用相匹配。for属性指明标签属于哪个控件。在源代码中，当控件与它的描述性文本并非直接相连时，这个方法很有用。而且这种方法还有其他的优势，比如可以使标签和控件作为两个不同的元素，当使用样式表对其元素时，它的作用会非常突出。

```
<label for="form-login-username">Login account</label>
<input type="text" name="login" id="form-login-username">

<label for="form-login-password">Password</label>
<input type="password" name="password" id="form-login-password">
```

使用标签的另一个好处是用户可以单击它们来选中表单元素。使用触摸设备的用户也可以方便地选中目标。

警告：撰写本文时，iOS设备中不可单击隐式标签，因此需要用JavaScript。我知道我们还没有做任何JavaScript，但如果你想知道，完整代码看起来像这样：

```
document.getElementsByTagName
('label').setAttribute
('onclick','');
```

提示

为了使表单相关的id不同于页面上的其他id，可以考虑在id前加上前缀“form-”，如例子中所示。

另一个可以将表单组织起来的技巧是给form元素一个ID名字，并且将其包含在控件所属ID的前缀中，如下面的例子：

```
<form id="form-login">

  <input id="form-login-
  username">

  <input id="form-login-
  password">
```


警告： 当Fieldset和legend涉及样式的时候，往往会有一些难题。例如，fieldset中背景颜色的处理在不同浏览器中都是不同的。在文本不换行时，legend是独一无二的。解决的办法是把一个span或b元素放在它们中，控制所包含元素的外观而不失可访问性。如果你给这些表单元素设置风格，那么请务必做大量的测试。

fieldset和legend

fieldset元素表示表单控件的逻辑组。一个fieldset字段可能还包括legend元素，用其提供一些说明。

图9-19显示了下面例子的默认显示，但你可以使用样式表来改变fieldset和legend的显示方式。

```
<fieldset>
  <legend>Mailing List Sign-up</legend>
  <ul>
    <li><label>Add me to your mailing list <input type="radio"
      name="list" value="yes" checked="checked"></label></li>
    <li><label>No thanks <input type="radio" name="list" value="no">
      </label></li>
  </ul>
</fieldset>

<fieldset>
  <legend>Customer Information</legend>
  <ol>
    <li><label>Full name: <input type="text" name="username"></label></li>
    <li><label>Email: <input type="text" name="email"></label></li>
    <li><label>State: <input type="text" name="state"></label></li>
  </ol>
</fieldset>
```

图9-19: fieldset和legend的默认显示

练习9-4 label 和fieldset

我们的比赛表单已经可以工作了，但我们需要添加相应的标签并建立一些fieldset使它更可用于辅助设备。请再一次打开contest_entry.html文档，按照下列步骤操作。

我喜欢逐步深入讲解，所以这个练习我们先将表单控件放在fieldset中，然后我们再做所有的标签。当然你也可以用其他方式做，只要完成标记标签和fieldset即可。

1. 表单顶部的“大赛报名信息”是紧密关联的信息，因此将其包裹在fieldset元素中。将fieldset中的标记部分的标题从段落 (p) 更改为legend。

```
<fieldset>
  <legend>Contest Entry Information</legend>
  <ul>
    <li>Name: <input type="text"
      name="username"></li>
    ...
  </ul>
</fieldset>
```

2. 接下来，将颜色、功能和尺寸等问题使用legend元素“Custom Shoe Design”组织在一个fieldset中，（文本已经存在，你只需要将它从p改成legend）。

```
<h2>Design your custom Forcefields:</h2>
<fieldset>
  <legend>Custom Shoe Design</legend>
  Color...
  Features...
  Size...
</fieldset>
```

3. 为颜色选项创建另一个fieldset，再将一个段落中的描述改在legend中。为特征和尺寸做同样的事。最后，在较大的“Custom Shoe Design”fieldset中将有三个fieldset。当你完成后，保存你的文件，并在浏览器中打开它。现在看起来应该非常接近图9-2所示的最终形式，当然由于浏览器的不同可能有一些差异。

```
<fieldset>
  <legend>Color <em>(choose one)</em>:</legend>
  <ul>...</ul>
</fieldset>
```

4. 好了，现在让我们来完成一些标签。在大赛报名信息fieldset中，明确地使用for/id标签方法将标签和文本输入联系起来。我已经给你做了第一个，你来做其他三个。

```
<li><label for="form-name">Name:</label> <input
  type="text" name="username" id="form-name"></li>
```

5. 对于单选和复选框按钮，将label元素包在input和它的特征值标签中。通过这种方式，当用户单击或者在标签元素中使用tab键时，按钮将会被选中。这是第一个，你做其他七个。

```
<li><label><input type="radio" name="color"
  value="red"> Red</label></li>
```

保存文件，你就大功告成了！默认情况下，标签对表单外观并不产生任何影响，但你可以对你已经添加的语义值感觉良好，而且可以在其他时候使用样式表。

表单布局和设计

在结束本章前，我得再讲讲表单设计，虽然本章只是关于标记，而非外观。

有用的表单

设计不当的表单会破坏你的站点上的用户体验，并给你的业务目标带来负面影响。糟糕的设计意味着失去客户，所以满足用户在桌面和小屏幕设备的特殊需求非常重要。你一定希望用户能顺利地完成下单购买等动作。

良好Web表单设计的主题是非常丰富的，为这个主题写一本书都不为过。事实上，确实有这样一本书：《Web Form Design》（Rosenfeld Media, 2008）由Web表单专家Luke Wroblewski编写，我高度推荐它。Luke的另外一本书，《Mobile First》（A Book Apart 2011），讲述了在移动环境下，如何格式化表单。你可以在他的站点上浏览几百篇文章：www.lukew.com/ff?tag=forms。

在这里，我会从《Web Form Design》摘取一些建议，但最好去看看这本书。

避免不必要问题。

为用户提供尽可能简单的表单，不必包括对手头任务不是绝对必要的问题。除了使速度变慢，这些额外的不必要的问题也可能使用户难以理解你的动机。如果你有另一种方法来获取信息（例如，信用卡的类型可由账户前四个数字确定），那就用这个方法，不要让用户有负担。如果有一些信息虽然有用，但并不是必需的，可以考虑要求在以后的时间（等你与用户建立良好的关系后）再获取。

考虑标签位置的影响。

与输入相关的标签的位置会影响填写表单所花费的时间。用户越能一眼了解页面的内容，表单就会完成得越快。把每个域的标签排成直线，以便于更快地扫描和填写，特别是当要求获取熟悉的信息时（用户名、地址等）。顶部位置的标签也可以容纳不同长度的标签，在小屏幕设备上也可以工作良好。即便是在长表单中也有用，因此，如果垂直空间是一个问题，你可以在输入的左侧放置标签。标签左对齐会使输入很慢，但如果你希望用户能够减缓输入、仔细查看和考虑各类信息，左对齐是适当的。

仔细选择输入类型。

正如你在本章看到，有相当多的输入可供选择，有时很难决定使用哪一个。例如，一个可选列表可以表示一个下拉菜单或复选框。仔细权衡每种控件类型的利弊，并跟进用户测试。

相关输入的分组。

如果表单中的域、菜单和按钮以相关的主题可视地组织起来，就很容易进行解析。例如，一个用户的联系人信息就可以作为一个组，这样即便是五六个输入都可以被视为一个单元。通常情况下，你需要做的只是添加一个精巧的指示，如细水平线和一些额外的空间。当然不要做过头。

分清主要和次要动作。

表单末尾的主要动作通常是某种形式的提交按钮（“买”、“注

册”等），表示完成表单并且准备进入下一步。你一定希望该按钮在视觉上凸显，很容易找到（沿表单主轴线对齐也可以）。次要动作是使你退后一步，如清除或复位表单。如果必须包括一个次要动作，请确保它的样式看起来不同，并且看上去不如主要动作重要。提供一个撤消操作的机会也是一个好主意。

表单样式

本章我们已经看到，表单标记的默认渲染并没有达到当今最专业的Web表单的效果。至于其他元素，你可以使用样式表创建一个整洁的布局以及改变大多数表单控件的外观。为了改善用户的印象，简单美好的事情也有很长的路需要走，比如对你的站点的控件采用统一的方式排列。

记住，表单部件由浏览器绘制并遵循操作系统约定。但是，你仍然可以对表单元素（比如文字输入、选择菜单、文本域、`fieldset`、标签和`legend`）应用尺寸、边距、字体、颜色、边框和背景效果等。只要确保在各种浏览器进行测试检查，以避免意外即可。第三部分的第18章中列出了一些具体技术，一旦你有更多的CSS经验，你就可以使用那些技术。如需更多帮助，在网上搜索“表单CSS”会出现大量教程。

自我测验

为这个测验准备好测验你所掌握的Web表单的知识了吗？下列是一些确认你已经入门的问题。

1. 在下列表单中，选择使用GET方法还是POST方法发送。
 - a. 访问网上银行账户的表单。
 - b. 给印刷商发送t-shirt美术设计的表单。
 - c. 搜索归档文章的表单。
 - d. 收集长篇论文的表单。
2. 哪种表单控件元素适合下列任务？当答案是“input”的时候，必须包括type特性。一些任务可能有不止一个正确答案。
 - a. 从12个星座中选出你属于的星座。
 - b. 确认你是否有心脏病史（是或否）。
 - c. 写书评。
 - d. 从8种口味的列表中选出你最喜欢的冰激凌口味。
 - e. 从25种口味的列表中选出你最喜欢的冰激凌口味。

3. 下列示例代码都包含一个错误，请指出。
- a. <input name="country" value="Your country here.">

b. <checkbox name="color" value="teal">

c. <select name="popsicle">
 <option value="orange">
 <option value="grape">
 <option value="cherry">
 </select>

d. <input type="password">

e. <textarea name="essay" height="6" width="100">Your story.</textarea>

元素回顾：表单

本章讨论了令人印象深刻的表单的元素和属性。（HTML 5）标记的元素是新的HTML 5规范引入的元素。

元素和特性	描述
button name="text" type="submit reset button" value="text"	普通输入按钮 提供控件的唯一变量名 自定义类型的按钮 指定发送到服务器的值
datalist [HTML 5]	为文本输入提供一个选项列表
fieldset	与控件和标签相关的组合
form action="url" method="get post" enctype="content type"	form元素 表单处理程序的位置（必需的） 提交表单数据的方法 编码方法，通常是application/x-www-form-urlencoded（默认）或multipart/form-data
input autofocus type="submit reset button text password checkbox radio image file hidden email tel search url date time datetime dateti me-local month week number rang- e color "	根据不同的type特性值，创建各种控件 当文档载入时，表示控件已经准备好了输入 输入的类型

元素和特性	描述
disabled form="form id alue" 阅读表9-1来获取每个输入类型相关属性的完整列表	与特定表单相关的控件 使输入失效，这样它就无法被选中
keygen [HTML 5] autofocus challenge="challenge string" disabled form="form id value" keytype="keyword" name="text"	为安全交易证书生成密钥 当文档载入时，表示控件将被高亮显示，并且可以输入 提供一个挑战字符串，以便使用key来提交 使控件失效，这样它就无法被选中 与特定表单相关的控件 指定需要生成的key的类型（比如rsa或ec） 给控件提供标识名字
label for="text" form="form id value"	附加信息到控件 通过id引用识别关联控件 与特定表单相关的控件
legend	指定字段集fieldset的标题
meter [HTML 5] form="form id value" high="number" low="number" max="number" min="number" optimum="number" value="number"	在已知的范围表示一个分数值 与特定表单相关的控件 表示尺寸“high”的范围 表示尺寸“low”的范围 指定范围的最高值 指定范围的最低值 表示“optimum”的数值 表示真实的或测量的值
optgroup disabled label=" text"	定义选项组 使optgroup失效，从而不能被选中 为选项组提供标签
option disabled label="text" selected value="text"	选择菜单控件的选项 使option不可用，从而不能被选择 为选项组提供替代标签 预先选中该选项 为选项提供可选用的值

元素和特性	描述
<code>output [HTML 5]</code> <code>for="text"</code> <code>form="form id value"</code> <code>name="text"</code>	表示计算结果 创建输出和另一个元素的关系 与特定表单相关的控件 为控件提供一个独特的变量名
<code>progress [HTML 5]</code> <code>form = "form id value"</code> <code>max="number"</code> <code>value="number"</code>	表示完成任务进度（即使不知道任务的最大值，也可以使用） 与特定表单相关的控件 指定任务的总值或最终大小 指明多少任务被完成
<code>select</code> <code>autofocus</code> <code>disabled</code> <code>form="form id value"</code> <code>multiple</code> <code>name="text"</code> <code>readonly</code> <code>required</code> <code>size="number"</code>	下拉菜单或滚动列表 当文档载入时，表示控件将被高亮显示，并且可以输入 表示控件无功能，可以用脚本触发 与特定表单相关的控件 允许在滚动菜单中多选 为控件提供唯一的变量名 使控件不能被用户修改 表示用户输入对这个空间是必需的 滚动列表的高度（以文本行计）
<code>textarea</code> <code>autofocus</code> <code>cols="number"</code> <code>dirname="text"</code> <code>disabled</code> <code>form="form id value"</code> <code>maxlength="text"</code> <code>name="text"</code> <code>placeholder="text"</code> <code>readonly</code> <code>required</code> <code>rows="number"</code> <code>wrap="hard sort"</code>	多行文本输入域 当文档载入时，表示控件将被高亮显示，并且可以输入 文本的宽度（以字符计） 允许定义文本方向 使控件不可用，从而不能选择该控件 与特定表单相关的控件 指定用户可以输入的最大字符数 为控件提供唯一变量名 提供一个简短的提示来帮助用户输入正确的数据 使控件不能被用户修改 表示用户输入对这个控件是必需的 文本区域的高度（以文本行计） 控制文本输入中的换行是否在数据中返回。“hard”保留换行； “soft”不保留

表9-1：每种输入类型可用属性

	submit	reset	button	text	password	checkbox	radio	image	file	hidden
accept									•	
alt								•		
checked						•	•			
disabled	•	•	•	•	•	•	•	•	•	•
maxlength				•	•				•	
name	•	•	•	•	•	•	•	•	•	•
readonly				•	•	•	•		•	
size				•	•				•	
src								•		
value	•	•	•	•	•	•	•		•	•
仅HTML 5										
autocomplete				•	•					
autofocus	•	•	•	•	•	•	•	•	•	
form	•	•	•	•	•	•	•	•	•	•
formaction	•							•		
formenctype	•							•		
formmethod	•							•		
formnovalidate	•							•		
formtarget	•							•		
height				•						
list				•						
max										
min										
multiple									•	
pattern				•	•					
placeholder				•	•					
required				•	•	•	•		•	
step										
width								•		

	email	telephone, search, url	number	range	date, time, datetime, datetime-local, month, week	color
accept						
alt						
checked						
disabled	•	•	•	•	•	•
maxlength	•	•				
name	•	•	•	•	•	•
readonly	•	•	•		•	
size	•	•				
src						
value	•	•	•	•	•	•
仅HTML 5						
autocomplete	•	•	•	•	•	•
autofocus	•	•	•	•	•	•
form	•	•	•	•	•	•
formaction						
formenctype						
formmethod						
formnovalidate						
formtarget						
height						
list	•	•	•	•	•	•
max			•	•	•	
min			•	•	•	
multiple	•					
pattern	•	•				
placeholder	•	•				
required	•	•	•		•	
step			•	•	•	
width						

第10章 HTML 5

在前几章里，我们已经使用过HTML 5元素了，但HTML 5远远不只是新的标记。HTML 5实际上是一种方法，它可以完成以前需要特殊编程才能完成的任务。它提供了标准的、开源的方式来展示声音，录像，网页上的互动元素，以及用于存储数据，脱机工作，利用位置信息的能力。通过HTML 5来完成普通任务，开发者可以依赖嵌入式浏览器的能力，而不再需要徒劳地挨个完成每个功能。

HTML 5提供了许多可能。实际上，它已经变成了一个流行术语。当市场营销人员和记者用HTML 5这一术语时，他们通常是指CSS3技术或者其他新的非Flash的网页技术。在本章，你会学到HTML 5规约实际包含了什么。当HTML 5符号被错误的应用时，我们会感到很烦恼。你也可以加入我们。最重要的是，网页标准的主流意识能够在与客户沟通中使我们的工作变得更加简单。

当然，随着规约（或标准）的发展，浏览器支持并不总是最好的。其中一些特征在当前可以得到很好的运用，但是另一些则不行。但是现在没有必要等待规约“全部完成”，浏览器可以逐一实现每个功能，然后鼓励开发者去使用它（参见侧栏“追踪浏览器支持”）。我还应该提到HTML 5进化得非常快，其中一些特性在你阅读本文的时候可能已经变了。我会尽自己最大的努力给你一个大概的了解，这样你就能自己决定研究哪个特性或功能。

HTML 5中许多新的东西都要求高超的网页开发技术，所以，你并不能一下子就运用它，但是我认为对它的运作有一个基本的了解是有益处的。

“基本的了解”就是我写这一章的目的。如果你想了解更为尖端的关于HTML 5的知识，我推荐你阅读下面的书：

本章内容

HTML 5是什么和不是什么

HTML 简史

新元素和属性

HTML 5 API

添加视频和音频

canvas 元素

追踪浏览器支持

有一些资源可以帮助你了解哪些HTML 5的特性可以使用。大多数特性都为CSS和选择器提供支持。

- 何时可用 (caniuse.com)
- HTML 5 Please (html5please.com)
- 维基百科“HTML 5布局引擎比较” ([en.wikipedia.org/wiki/Comparison_of_layout_engines_\(HTML_5\)](http://en.wikipedia.org/wiki/Comparison_of_layout_engines_(HTML_5)))

注意：对于万维网和HTML的初学者而言，想要了解详细的历史，可以阅读David Raggett的书“Raggett on HTML4” (Addison-Wesley, 1998)，可以在W3C站点访问 (www.w3.org/People/Raggett/Book4/ch02.html)。

- Mark Pilgrim 写的《HTML 5, Up and Running》(O'Reilly Media和Google出版)
- Bruce Lawson和Remy Sharp 写的《Introducing HTML 5》(New Riders出版)

不得不告诉你，本章是本书的精华。虽然学会它可能并不会使你感觉愉悦，但是绝对对你有益处。初级的网页设计师应该理解其美好的前景和应用的方式。

在通往XHTML 2的路上发生的有趣的事

为了理解为目标所做的准备，需要快速地了解历史，我们就从这里开始。

别眨眼，否则你会错过了了解HTML的机会

HTML的故事是迷人而又无序的，从Tim Berners-Lee在1991年创造HTML一直延续到现在标准化的HTML 5。为了发布一个可行的版本，早期的HTML版本（1994年的HTML+和1995年的HTML 2.0）建立在Tim的工作之上。但是当万维网像旋风一样在世界刮过时，尤其是MosaicNetscape和之后的Microsoft Internet Explorer浏览器，浏览器开发者来不及等待任何标准的制定。他们给人们所想要的东西，创造了一系列的特定浏览器的元素，改善网页在他们各自的浏览器上的外观。这已经称为浏览器大战。因此，它成为在20世纪90年代后期，为一个网站创建两个不同的版本，以适应这两大浏览器版本，就显得很寻常了。

新成立的W3C一锤定音，于1996年发布了第一份规范：HTML 3.2。HTML 3.2包含了当时使用的所有的HTML元素，并且包含了HTML的外观拓展，这些拓展的出现也来自于Netscape和IE的斗争，以及没有可替代的样式表。HTML 4.0（1998）和HTML 4.01（1999年的轻微修订后的替代版）旨在使HTML回到正轨，它们强调结构与表现的分离，以及提高可访问性。表现层的事都交给了新指定的层叠样式表标准。

XHTML登场

就在HTML 4.01被开发出来的同一时期，W3C的工作人员意识到一个有局限的标记语言并不能完美地描述所有的在Web上显示的信息（化学符号、数学方程、多媒体演示和财务信息等）。其解决之道就是XML（可扩展标记语言），一个创造标记的语言。XML是一简单化的SGML（标准通用标记语言），Tim Berners-Lee曾经用这个来创建其最初的HTML应用。但是SGML已经证明它比网站所要求的更复杂。

W3C希望有一个拥有许多标记语言的基于XML版本的Web。当然，抛开这不说，许多人都喜欢细致的标记文件，遵守XML句法以减少潜在的混乱。

他们的第一步就是根据XML句法来重写HTML以与其他句法兼容。其结果就是XHTML（可扩展HTML）。它的第一个版本XHTML 1.0基本上和HTML 4.01完全一样，拥有相同的元素和属性，但是对于标记该如何完成则有更为严格的要求。

HTML 4.01和严格基于XML的兄弟XHTML 1.0成为网页标准运动的奠基石（见侧栏“网页标准项目”）。在写本书的时候，它们还是支持最彻底和最连贯的标准。

但是W3C并没有停在这。为了实现基于XML的Web，他们开始研发XHTML 2.0，这是为了使工作良好而做的比HTML 4.01更为大胆的尝试。问题是，它不能反向兼容旧有的标准和浏览器行为。标准的撰写和批准过程拖了好几年，而没有浏览器实现。由于没有浏览器的实现，XHTML 2.0就被卡住了。

你好，HTML 5！

在2004年，Apple、Mozilla和Opera的成员从W3C中分离出来，组成了网页超文本应用技术工作组（WHATWG，whatwg.org）。其目的是推动HTML的发展使之能够和现实生活的引用和浏览器相一致。他们刚开始的文件Web Applications 1.0 和Web Forms 1.0一起组成了HTML 5，当前仍然在Ian Hickson（当前在Google）的领导下开发。

W3C最后基于WHATWG所完成的工作上建立了它自己的HTML 5工作组（也是由Hickson领导的）。在写本书的时候，HTML 5规约的工作是在两个组织的协作，甚至有时是冲突的状态下进行的。在写本书的时候，该标准还不是正式的推荐，但是这并不能阻止浏览器逐渐地实现。

XHTML标记需求

- 元素和属性名必须为小写。在HTML中，元素和属性名不区分大小写。
- 所有元素必须被关闭（终止）。右尖括号前加上斜线表示空元素（如
）。
- 属性值必须在引号中。单引号或双引号都是可以接受的，只要使用保持一致。此外，在引号内，不应该有多余的空格（字符空格或回车）。
- 所有的属性都必须有显式的属性值。XML（也包括XHTML）并不支持属性最小化，在SGML实践中，某些属性可以缩减到只有属性值。所以在HTML中，你可以用checked指示当表单载入时，表单按钮被选中，而在XHTML中，你需要显式地写checked="checked"。
- 元素的适当嵌套需要严格执行。一些元素有新的嵌套约定。
- 特殊字符必须用字符实体来表示（如在&中的&字符）。
- 使用id而不是name作为标识符。
- 脚本必须在CDATA部分中，这样它们才会被当做简单文本字符，而不被解析为XML标记。下面是该语法的一个例子：

```
<script type="type/javascript">
  // <![CDATA[
    ... JavaScript goes here...
  // ]]>
</script>
```

HTML 5的目标是使HTML更有助于创建Web应用程序。

网页标准项目

在1998年，浏览器之争最激烈的时候，被称为网页标准项目（简称WaSP）的草根联盟开始给浏览器创建者施加压力（当时主要是Netscape和Microsoft），以使他们开放标准，并由W3C文档化。而且不仅如此，他们对Web开发社区进行培训，说明了标准开发的好处。他们的努力改变了网页开发和支持的方式。当今的浏览器（甚至是Microsoft）不仅对标准进行支持，而且进行持续的创新。你可以在WaSP站点阅读他们的任务说明、历史和当前的努力（webstandards.org）。

注意：WHATWG在www.whatwg.org维护着HTML的“最新标准”（最新意味着没有版本号）。它与HTML 5接近，但是它包含了一些W3C不打算采用的额外的元素和特性，除此之外，它还有一组略微不同的API。

XHTML 2.0呢？在2009年年末，W3C正式将工作组的资源和努力引入HTML 5。

这也是我们怎么会在这里，这也是进入本章实质内容前的前奏，本章的实质内容是HTML 5提供的新特征。我也建议你阅读侧栏“HTML 5趣事”，可以获取规约的许多趣事。我将给你介绍HTML 5的新内容，包括：

- 新的DOCTYPE
- 新的元素和特性
- 废弃的4.01元素
- API

HTML 5趣事

HTML 5是在之前的HTML版本上建立的，而且引入了许多重大的新起点。下面是HTML 5规约的一些趣事。

- HTML 5基于严格的HTML 4.01，HTML的版本并不包括任何基于演示的或其他过时的元素和属性。这意味着HTML 5绝大部分内容是由已经使用多年的相同的元素组成的，浏览器也知道如何处理这些元素。
- HTML 5不使用DTD（文档类型定义），DTD是一个文档，它定义标记语言中的所有元素和属性。这也是你使用SGML的方式，如果你还记得，HTML是根据SGML规则制作的草案。HTML4.01由三个独立的DTD定义：过渡（包括遗留的被标记为“过时的”或即将淘汰的元素），严格（不建议使用的功能剥离出来，如前所述）和框架（将文档分解成单独滚动帧，现在被认为是过时的技术）。

- HTML 5是HTML第一个规范，其中包括详细的说明，包括浏览器应该如何处理畸形和遗留的标记。当浏览器遇到不正确或非标准的标记时，浏览器制造商可以遵循一个标准的约定来处理。
- HTML 5也可以根据严格的XML语法（称为HTML 5的XML序列化）来编写。一些开发商都喜欢整洁良好的XHTML（小写元素名称、引号中的属性值、关闭所有的元素等），这样的写作方式仍是一个选项，并不是必需的。在特殊的情况下，为了与其他的XML应用程序协作，可能需要一个HTML 5文档来作为一个XML，它可以使用XML语法来工作。
- 除了标记，HTML 5定义了大量的API（应用编程接口）。API使得与基于Web的应用程序更容易沟通。他们还可以将一些常用的程序（如音频和视频播放器）移到本地的浏览器上。

标记部分

首先看下HTML 5的标记部分，之后介绍API。

最小的DOCTYPE

正如在第4章看到的一样，HTML文档应该由DOCTYPE声明开始，它能辨别出HTML文档的版本类型。HTML声明很短也很简单：

```
<!DOCTYPE html>
```

把它与严格HTML 4.01作比较：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/HTML4.01/strict.dtd">
```

为什么会这么复杂？在HTML 4.01和XHTML 1.0、1.1中，声明必须指向公有DTD（文档类型定义），DTD是一个在标记语言中用来定义所有元素和规则的文件。HTML 4.01由三个独立的DTD定义：过渡（包括遗留元素，比如字体；特性，比如不赞成使用的align，也许很快会被淘汰），严格（不建议使用的功能剥离出来）和框架（将文档分解成单独滚动帧，现在被认为是过时的技术）。HTML 5并没有DTD，这就是需要简单的DOCTYPE声明的原因。

DTD是SGML剩余的东西，并且已经被证明在Web中不如想象的那么有帮助。所以HTML 5的编写者就不会采用它了。

验证器：检查所有的文件中的标记是否正确的软件，它用DOCTYPE声明来确认文件遵守了其特定的规则。侧栏“HTML DOCTYPE”列出了通常使用HTML 4.01或者XHTML 1.0的声明。

注意：为了检测你的文档是否有效，可以使用W3C的验证器（validator.w3.org）。html5.validator.nu也有一个HTML 5规范的验证器。在Adobe Dreamweaver也有一个内置的验证器，可以使用它来验证你的文档。

HTML DOCTYPE

下面列出了所有通常使用的文档声明类型。

HTML 5

```
<!DOCTYPE html>
```

HTML 4.01 Transitional

过渡DTD包括废弃的元素和属性：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/HTML4.01/loose.dtd">
```

HTML 4.01 Strict

严格DTD定义省略所有过时的元素和属性：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/HTML4.01/strict.dtd">
```

HTML 4.01 Frameset

如果你的文件包含框架，也就是说，它使用frameset而不是body作为内容，那么可以将框架标识为DTD：


```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/HTML4.01/frameset.dtd">
```

XHTML 1.0 Strict
和HTML 4.0严格一样，但重新根据XML语法规则形成：

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

XHTML 1.0 Transitional
和HTML 4.01过渡一样，但重新根据XML语法规则形成：

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

XHTML 1.0 Frameset
和HTML 4.01框架一样，但重新根据XML语法规则形成：

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

元素与属性

HTML 5引入了许多新的元素。你会发现它们散见于这本书中。但是表10-1把它们都列了出来。

注意：HTML 5不同于HTML 4.01的详细列表，可以看看W3C的官方文档www.w3.org/TR/html5-diff/。

表10-1：HTML 5中的新元素

article	datalist	header	output	source
aside	details	hgroup	progress	summary
audio	embed	keygen	rp	time
bdi	figcaption	mark	rt	track
canvas	figure	meter	ruby	video
command	footer	nav	section	wbr

新的表单输入类型

我们在第9章中包括了新的表单输入控件类型。但是在这里，我们仍会再看下，他们是：color、date、datetime、datetime-local、email、month、number、range、search、tel、time、url和week。

新全局属性

全局属性是那些可以应用于任何元素的属性。全局属性的数目在HTML 5中得到了扩展，并且许多全局属性都是新的（见表10-2）。在写本书的时候，W3C仍然在添加和删减属性，因此有必要查看最新的标准（dev.w3.org/html5/spec/global-attributes.html#global-attributes）。

表10-2：HTML 5中的全局属性

属性	值	描述
accesskey	简单的文本字符	给链接指派一个访问键（快捷键命令）。访问键可以用于表单域。用户可以通过单击Alt键（PC机）或者Ctrl键(Mac)来访问元素
aria-*	WAI-ARIA中的标准状态或属性关键字	WAI-ARIA（可访问富互联网应用）定义了一种方式，以便于使用辅助设备的用户更容易访问网页内容和应用。HTML 5允许ARIA的任何属性和角色都能被添加到元素。比如，div用于弹出式菜单，可以包含aria-haspopup属性，以便于使用非可视浏览器的用户。也可以看看相关的role全局属性
class	文本字符串	将一个或更多分类名指派给元素
contenteditable	true false	表示用户可以编辑元素。这个属性在当前的浏览器已经得到很好的支持
contextmenu	menu元素的id	给某个元素指定右键菜单。右键菜单必须是用户请求，比如，右击鼠标
data-*	文本字符串或数字	使作者可以创建通用的数值相关的属性（符号“*”意思是“任意”），比如，data-length、data-duration和data-speed等。所以数值可以被应用程序或脚本使用
dir	ltr rtl	指定元素的方向（左到右或者右到左）
draggable	true false	true值表示元素可以被拖拽，也就是说可以通过单击按住，然后拖拽移动到窗口的新位置
dropzone	copy link move s:text/plain f:file-type, 比如f:image/jpg	表示元素可以拖拽和放开文本或文件数据。值是空格分开的列表，其中包含它可以访问的数据类型（s:text/plain用于文本字符串；f:file-type用于文件类型），还包含一个关键字，说明当拖拽内容放开的时候如何处理：copy会复制拖拽的数据；move会移动数据到新位置；link会生成指向初始数据的链接
hidden	在XHTML中，HTML文档没有值，可以设置值hidden="hidden"	防止元素及其后代呈现在用户设备上（浏览器）。任何隐藏的脚本或表单控件中部分仍然会执行，但不会呈现给用户
id	文本字符串（不能用数字开始）	给元素指派一个独一无二的标识名

表10-2：HTML 5中的全局属性（续）

属性	值	描述
lang	两字符的语言代码（可以看看 www.loc.gov/standards/iso639-2/php/code_list.php ）	将语言代码指定给元素
role	WAI-ARIA中的标准角色关键字（可以看看 www.w3.org/TR/wai-aria/roles ）	将一个标准化的WAI-ARIA角色分配给一个元素，使残疾用户使用得更清晰。例如，一个div的内容，将在可视浏览器显示一个弹出式菜单，它可以标记为role="menu"，以便于屏幕阅读器使用
spellcheck	true false	表明元素会检查拼写和语法
style	分号分隔的样式规则列表（property:value对）	与元素相关的样式信息。比如，<h1 style="color: red; border: 1px solid">Heading</h1>
tabindex	数字	指定当前元素在当前文档中的Tab键顺序中的位置。该值必须是0~32 767之间。它用于在页面上的链接或表单中的字段中使用tab键切换，是非常有用的辅助浏览设备。值-1是允许从跳位流中移除元素，并且它们可仅由JavaScript来获取焦点
title	文本字符串	提供与元素有关的一个标题或咨询信息，典型地显示为工具提示（tooltip）

淘汰了的HTML 4.01标记

HTML 5还宣布许多HTML 4.01中的元素被淘汰了，因为它们不再被支持（表10-3）。如果你用它们，浏览器还会支持它们，但是我强烈推荐把它们扔到九天云外。

表10-3：HTML 5中已经过时的HTML 4元素

acronym	dir	noframes
applet	font	strike
basefont	frame	tt
big	frameset	
center	isindex	

还在跟着我吗？我知道这些东西很枯燥。这就是插入了图10-1的原因。它和HTML 5没关系，但是我认为，在继续API之前，看到它能换换心情。



图10-1：这只可爱的浣熊与HTML 5无关（Tara Menne摄）

遇到API

HTML 5之前的HTML规范只包括元素的文档、属性和语言允许的值。对于简单的文本文件没什么问题，但HTML 5的创作者想要更容易地创建基于Web的应用程序，需要使用脚本和编程。出于这个原因，HTML 5还定义了一些新的API，使它更容易与程序沟通与应用。

API（应用编程接口）是一个文档化的命令集、数据名称等，以便于一个应用程序与另一个应用程序通信。比如，Twitter的开发者将每个数据类型，以及访问数据类型的方法都编写在API文档中（*dev.twitter.com/docs*），以便于其他开发者使用这些方法和元素。因此，有很多Twitter程序和widget。Amazon.com也通过API开放了其产品的数据。事实上，其他发布者也认识到通过API开放自己内容的影响力。可以说API现在非常流行。

但是，让我们回到HTML 5，它有一些API，这些API可以完成传统上需要专有插件（如动画）或自定义编程来完成的任务。这个想法是，如果浏览器可以原生地提供这些功能（遵循一定的规范），开发者就可以做各种有趣的事并且在所有浏览器都有效，正如我们今天可以在页面上嵌入图片一样。当然，我们离这些还有很长一段路，但是我们正在稳步前行。一些API有标记组件，如使用新的HTML 5 `video`和`audio`元素来嵌入多媒体。其他的API可以在后台调用JavaScript或服务器端组件，比如可以

注意：如果你想了解所有的API，可以阅读Erik Wilde的文章“HTML Landscape Overview” (dret.typepad.com/dretblog/html5-api-overview.html)。W3G在www.w3.org/TR/tr-title-all中列出了他们维护的所有文档，其中有许多是API。

创建Web应用程序，即便在没有互联网链接的时候也可以工作（离线Web应用程序API）。

W3G 和WHATWG为网络应用程序提供了许多API，这些API已经实现或正在实现。大多数API有自己的规范，这些规范不同于HTML 5标准，但一般也把它们归为网络应用程序广泛使用的HTML 5。HTML 5包含以下API规范：

媒体播放器API

用于控制嵌入网页的音频和视频播放器，使用新的video和audio元素。在本章后面将仔细讲解video和audio元素。

会话历史API

用于更好地控制后退按钮，显示浏览器历史记录。

离线网络应用API

它使一个应用程序在没有互联网连接的情况下仍然可以工作。它包括一个所有应该被浏览器下载下来的缓存文件和资源的清单。当连接可用时，它会检查文件是否已经改变，然后更新这些文件。

编辑API

提供一个命令集，可以用来创建基于浏览器的文本编辑器，允许用户插入和删除文本，格式化文本，比如粗体、斜体，或作一个超文本链接。此外，有一个新的contenteditable属性允许任何内容元素在页面上可编辑。

拖放API

拖动选择的文本或文件到本页面或另一个页面的目标区域。draggable属性表示元素可被选择和拖动。dropzone属性用于目标区域，它可以定义什么类型的内容可以接受（文本或文件类型）以及如何处理这些内容（copy, link, move）。

以下一些是由W3C开发的，并且有自己的标准和规范的API（HTML 5以外的）：

Canvas API

canvas元素把一个动态的、二维的空间添加到一个页面上。在这一章的结尾会看到它。

网页储存API

允许数据可以被存储到浏览器的缓存，这样应用程序就可以使用它。传统上，它已经由cookies完成了，但网络存储的应用程序API允许更多的数据存储。它还控制了数据是否限制到一个session（sessionStorage:当窗口关闭，数据就清除）或者基于一个域名

(localStorage:所有指向该域名的窗口都可以访问数据)。

定位API

允许用户分享他们的地理位置(经纬度)，这样在网络应用中就可作用于脚本。这允许应用程序提供位置感知等功能，比如，找出在附近的餐馆或在你的地区找到其他用户。

网络工作者API

提供了一种在后台运行计算复杂的脚本的方法。这使得浏览器保持网页界面对用户操作的快速响应。网络工作者API是HTML 5的一部分，但是在W3C中，它已经独立出来了。

网络套接字API

创建一个“套接字(socket)”，它是一个浏览器客户端和服务端之间的开放连接。这使得信息在客户端和服务端之间实时流动，对于传统的HTTP请求没有时滞。它对于多人游戏、聊天或不断更新的数据流，比如体育、股票信息或社交媒体流，是有用的。

一些API有相关的HTML元素，比如，用于在页面上嵌入媒体播放器的audio和video元素，以及用于添加动态图像区域的canvas元素。在下面的章节中，我们将简单看看如何使用这些元素。

视频与音频

在万维网早期，就可以把一个MIDI文件添加到网页中(比如，早期视频游戏中的声音)。没过多久，就有了更好的选择，它可以使用RealMedia和Windows Media，将各种格式的音频和视频文件嵌入网页中。到最后，动画等多媒体文件也可以嵌入网页之中，这得感谢YouTube和类似的视频服务提供商。

所有这些技术的共同点是他们需要下载、安装第三方专有的插件以便播放媒体文件。浏览器没有可以处理声音或视频的内置功能，所以插件可以填补这个空白。随着网络作为一个开放的标准平台不断地发展，对多媒体的支持成为浏览器必须具备的功能之一。因此，新的audio和video元素以及各自的API就出现了。

好消息和坏消息

好消息是，当今的浏览器对audio和video元素提供了很好的支持，包括IE 9+、Safari 3+、Chrome、Opera、Firefox3.5+等桌面浏览器，以及IOS Safari 4+、Android 2.3+和Opera Mobile等移动设备浏览器(但是不包括Opera Mini)。

注意： 你可以将网络socket看做浏览器和服务端之间的电话通话，这与传统的浏览器/服务器通信的单工模式，即步话机模式不同。

告别Flash?

苹果公司宣布在iOS设备上将不再支持Flash，转而隆重推荐HTML 5，导致Adobe停止了它的移动设备Flash产品的开发。不久，微软也宣布停止开发Silverlight媒体播放器，而使用HTML替代。在写本书的时候，HTML 5还未能达到Flash的效果和性能，但是它毕竟在不断前进。这也意味着在今后的几年中，我们依然会看到桌面设备上的Flash和Silverlight，但趋势必然是远离插件，向着Web标准技术发展。

但是你可能会以为这是一个完美的世界，所有的浏览器都完美地支持音频和视频，实际上恐怕这并不如你想的那么简单。虽然他们都可以用标记语言和脚本以嵌入和控制媒体播放器，可惜他们没有就所支持的格式达成一致。让我们简单地介绍媒体文件格式。如果你想添加音频或视频到你的网页，这些东西都很重要，必须理解。

媒体的格式是如何起作用的

当你准备以音频或视频格式进行网页传输时，你需要决定两种格式。第一个格式是媒体如何编码（编码是一种算法，可以将源文件转化为0、1序列，并且压缩）。用于编码的方法就是编解码器（codec）。编解码器有很多，其中一些听起来很熟悉，比如MP3，其他可能听起来还很新鲜，比如H.264、Vorbis、Theora、VP8和AAC。幸运的是，只有少数解码器是适用于Web的，我们很快就会讲到。

其次，你需要为媒体选择一个容器格式……你可以认为它是一个把压缩媒体及其元数据绑在一起的ZIP文件。通常一个容器格式可容纳多个编解码器，详细内容是复杂的。因为空间有限，所以在这一章中，我只能开门见山地介绍最常见的网络容器/解码器组合。如果你要添加视频或音频到你的网站，我鼓励你详细了解所有这些格式。侧栏“进一步阅读：HTML 5 Media”是一个开始。

了解视频格式

对于视频来说，最普遍的选择是：

- Ogg container + Theora video codec + Vorbis audio codec。这通常称为“Ogg Theora”，这种文件应该有一个`.ogv`的后缀。这个选项中的所有编解码器和容器是开放源代码的，不受专利或版权的限制，这使它们适用于网络发布，但有些人说其质量不如其他选项。
- MPEG-4 container + H.264 video codec + AAC audio codec。这种组合是一般称为“MPEG - 4”，并以`.mp4`或者`.m4v`为后缀。H.264是一个高品质的、灵活的视频编解码器，但它是有专利权的，必须缴费获得许可。版权许可使得浏览器拒绝支持。
- WebM container + VP8 video codec + Vorbis audio codec。
“WebM”是一个新的容器格式，用`.webm`作为后缀。它是专门为VP8和Vorbis设计的，而且它的优势在于它是开放源代码和免费的。

当然，我前面提到的问题是，浏览器制造商不同意单独支持一种格式。一些浏览器支持开放源码，版权免费的选择，像Ogg Theora或WebM。其他浏览器尽管其有使用费的要求，仍坚持使用高质量的H.264。那就意味

进一步阅读：HTML 5 Media

如果你准备好了解更多的HTML 5 Media，我推荐这些书：

- Shelley Powers写的《HTML 5 Media》（O'Reilly Media）
- Mark Pilgrim写的《HTML 5 UP and Running》（O'Reilly Media），有一章节讲述了HTML 5视频
- Sylvia Pfeiffer写的《The Definitive Guide to HTML 5 Video》（Apress）

着网站开发者需要作出多个版本的视频来保证支持所有的浏览器。表10-4列出了浏览器都支持的多种视频选项。

表10-4：当前浏览器的视频支持（2012年年中）

Format	Type	IE	Chrome	Firefox	Safari	Opera Mobile	Mobile Safari	Android
Ogg Theora	video/ogg	–	5.0+	3.5+	–	10.5+	–	–
MP4/H.264	video/mp4	9.0+	–	–	3.1+	–	3.0+	2.0+
WebM	video/webm	9.0+	6.0+	4.0+	–	11+	–	2.3.3+

了解音频格式

尽管音频格式看起来有类似的多种选择，但没有一种格式是被所有的浏览器支持的（表10-5）。

- **MP3**。该文件格式是一个编解码器，扩展名为`.mp3`。它已成为很普遍的音乐下载的文件格式。MP3（MPEG-1 Audio Layer 3的简写）已经授予专利，使用它需要支付的费用也已由硬件和软件公司（而非媒体制作商）支付。
- **WAV**。WAV格式同样也是一个编解码器（`.wav`）。
- **Ogg container + Vorbis audio codec**。它通常指“Ogg Vorbis”，保存的时候扩展名是`.ogg`或者`.oga`。
- **MPEG 4 container + AAC audio codec**。它不如MP3使用得那么广泛。
- **WebM container + Vorbis audio codec**。WebM（`.webm`）格式也包括音频。

表10-5：当今浏览器支持的音频（2012年）

Format	Type	IE	Chrome	Firefox	Safari	Opera Mobile	Mobile Safari	Android
MP3	audio/mpeg	9.0+	5.0+	–	4+	–	3.0+	2.0+
WAV	audio/wav或 audio/wave	–	5.0+	3.5+	4+	10.5+	3.0+	2.0+
Ogg Vorbis	audio/ogg	–	5.0+	3.5+	–	10.5+	–	2.0+
MPEG-4/AAC	audio/mp4	9.0+	5.0+	–	4+	–	3.0+	2.0+
WebM	audio/webm	9.0+	6.0+	4.0+	–	11+	–	2.3.3+

视频和音频编码工具

编辑和编码视频、音频文件有很多方式，在这里我无法一一讲述，下面是一些能完成这些工作的免费工具。

视频转换

- Miro Video Converter (www.mirovideoconverter.com) 是一个免费的工具，可以将任何视频转换为为移动设备和桌面设备优化的H.264、Ogg Theora或者WebM格式，并且提供了简单的拖放界面。它可用于OS X和Windows。
- Handbrake (handbrake.fr) 是一个流行的开源工具，可以更好地控制H.264设置。它可用于Windows、OS X和Linux。
- Firefogg (firefogg.org) 是Firefox的一个扩展，它可以将视频转换为Ogg Theora格式。在Firefox3.5+

上安装Firefogg扩展，然后访问Firefogg扩展，并且使用它们的在线界面转换视频。

音频转换

- MP3 / WMA / Ogg 转化器 (www.freemp3wmaconverter.com) 是一个免费的工具，可以转换的音频格式包括：MP3、WAV、WMA、OGG、AAC等。Mac用户无法使用，只能用于Windows。
- 在Mac上，可以使用Max，这是一个开源的音频转换器，它可以从sbooth.org/Max/下载。虽然Audacity (audacity.sourceforge.net) 是一个录制工具，但是它也提供基本的转换工具。

把视频添加到网页中

`<video>...</video>`
给页面添加一个视频播放器

我想是时候讲解添加视频到网页的标记了（毕竟本书是关于HTML的）。让我们从一个例子开始，假设你正在做一个设计，你确切地知道你的用户将使用什么浏览器。在这种情况下，你可以使用video标签的src属性（就像img元素的用法一样），并且只提供一个视频格式。图10-2显示了在浏览器中使用默认播放器播放的电影。了解了这个例子，再来看看其他属性。

```
<video src="highlight_reel.mp4" width="640" height="480"
poster="highlight_still.jpg" controls autoplay>
</video>
```



图10-2：使用视频元素嵌入电影（使用的是Mac上的Chrome）。

在例子中有一些有趣的属性，值得详细看看。

`width="pixel measurement"`

`height="pixel measurement"`

指定嵌入式媒体播放器占据屏幕框的大小。通常，最好能使维度完全匹配电影的像素维数。这部电影将会调整大小以匹配这里设置的维度。

`poster="url of image"`

在视频播放之前，提供一张静止的图像。

Controls

添加controls属性，可以让浏览器显示内置的媒体的控件，一般就是一个播放/暂停按钮，一个“导引”可以让你转到视频的一个位置，以及音量控制。如果你想要更多的跨浏览器的一致性，你可使用CSS和JavaScript创建自己的自定义播放器界面。本章没有讲述如何做到完成这个功能，但列出的侧栏资源“进一步阅读：HTML 5 Media”可以解释。在许多情况下，默认设置就足够。

autoplay

当视频下载足够时，视频就自动开始播放。在一般情况下，使用autoplay应避免让用户自行决定何时播放视频。

此外，video（和audio）元素可以使用loop属性使视频循环播放，静音播放视频元素（muted）的视频没有音频轨道，mediagroup可以将一个video元素成为一组相关的媒体元素的一部分（如视频和同步的语言翻译），preload属性可以建议浏览器是否在页面加载时就下载获取视频数据（preload="auto"）或等待，直到用户按下播放按钮（preload="none"）。设置preload="metadata"可以加载媒体文件的信息，但不是媒体本身。一台设备可以决定如何最好地处理auto设置；例如，在智能手机的浏览器，可以设置为不预先载入媒体，以保护用户的数据使用量，即便当它被设置为auto。

所有的视频

但是，且慢！我们已经知道，一种视频格式在现实世界中是不会自动裁剪的。最起码，你需要做两个版本的视频：Ogg Theora格式和MPEG-4（H.264视频）。一些开发人员喜欢WebM而不是Ogg，因为浏览器对这两种格式支持的几乎一样好，但是WebM文件较小。对于使用不支持HTML 5视频的浏览器的用户，你可以在页面上嵌入Flash播放器，或使用像YouTube或Vimeo的服务，在这种情况下，让它们来处理转换，而你只要复制嵌入代码即可。

警告： iOS3设备无法播放带有poster属性的视频，所以如果你需要支持iPhone和iPad就不要使用它。

object和embed

object元素是在页面中嵌入媒体，如电影、Flash movie、applet甚至是图片的一般方法，它包含很多的param（参数）元素，用于提供object需要播放的资源 and 说明。你也可以在object元素中添加备用的内容，以便在媒体无法支持的时候，使用备用内容。这个属性和参数根据object类型而改变，有些时候由第三方播放媒体的插件决定。

object的类似元素embed，也可以把媒体插入到网页。它已经成为一个非标准，但是广泛支持的元素，而且最终出现在HTML 5中。一些媒体需要使用embed元素，它常在一个object元素中用于备用内容，以便在所有浏览器都适用。

你可以下一页看到“每个人的视频”中使用object和param元素的一个代码例子。

在标记中，`video`元素中的一系列`source`元素指向每个视频文件。浏览器在列表中向下寻找，直到它们找到一个支持的版本并下载该版本。Flash可以用传统的`object`及`embed`元素来添加，因此，如果浏览器无法理解`video`和`source`，那么很可能使用Flash播放。最后，为了确保可访问性，强烈建议添加一些简单的链接以供下载视频，使以上方法出现问题的时候，能够用任何其他可用的媒体播放器来播放。

不多废话，这里是一个非常完整的代码示例，它可为所有用户提供服务，包括移动设备的用户。你可以选择不提供所有这些格式，而是因地制宜地使用相应的格式。

下面的例子是基于Kroc Camen的文章“每个人的视频”（camendesign.com/code/video_for_everybody）中的代码。强烈建议检查该页面的更新、修改代码指示以及更多的技术细节。看看该例子的各部分。

```
<video id="yourmovieid" width="640" height="360" poster="yourmovie_
still.jpg" controls preload="auto">
  <source src="yourmovie-baseline.mp4" type='video/mp4;
codecs="avc1.42E01E, mp4a.40.2"'>
  <source src="yourmovie.webm" type='video/webm; codecs="VP8,
vorbis"'>
  <source src="yourmovie.ogv" type='video/ogg; codecs="theora,
vorbis"'>
  <!--Flash fallback -->
  <object width="640" height="360" type="application/x-shockwave-
flash" data="your_flash_player.swf">
    <param name="movie" value="your_flash_player.swf">
    <param name="flashvars" value="controlbar=over&image=poster.
jpg&file=yourmovie-main.mp4">
    
  </object>
</video>
<p>Download the Highlights Reel:</p>
<ul>
  <li><a href="yourmovie.mp4">MPEG-4 format</a></li>
  <li><a href="yourmovie.ogv">Ogg Theora format</a></li>
</ul>
```

注意：如果你仔细比较，会发现在`source`元素的`type`属性中，字符串都用单引号括起来。这是因为`codecs`必须用双引号括起来，所以整个属性就需要不同的引号类型以避免混淆。

注意：在这个例子中，为了在iOS3中播放，MPEG-4视频在“基线”中提供。如果你在阅读本书的时候，iOS3已经过时，你就可以提供高质量的“main”配置文件来替代：

```
<source src="yourmovie-
main.mp4" type='video/mp4;
codecs="avc1.4D401E, mp4a.40.2"'>
```

每个`source`元素都包含媒体文件（`src`）的位置和文件类型（`type`）的信息。除了列出文件容器的MIME类型（例如，`video/ogg`），还列出了使用的编解码器（见注意）。对于MPEG-4视频来说，这是特别重要的，因为H.264编解码器具有不同的配置文件，如`baseline`（用于移动设备）、`main`（用于桌面Safari和IE 9+）、`extended`和`high`（这两个一般不用于网络视频）。每个配置文件都有其自己的ID，就像你所看到第一个`source`元素的例子中那样。

从技术上讲，`source`元素的顺序并不重要，但在早期的ipad上有一个

bug，它要求baseline最好将MPEG-4设置为第一。如果它排在后面，运行iOS3的ipad就不会找到它，而且把它放在第一并不会影响任何其他浏览器。

source元素之后，一个object元素用于嵌入一个Flash播放器，以便为有Flash插件的浏览器播放MPEG-4视频。有很多的Flash播放器，但Kroc Camen的（“每个人的视频”）推荐JW播放器，这是很容易安装的（只是把一个JavaScript的.js文件和Flash的.swf文件放在你的服务器上）。从www.longtailvideo.com/players/jw-flv-player/下载JW播放器和说明，并进行安装和配置。如果你使用JW播放器，将例子中的your_flash_player.swf替换为player.swf。

很需要注意的是，Flash备用内容是用于无法识别video元素的浏览器的。如果浏览器支持video，但只是不支持某种媒体文件格式，它就不会播放Flash版本。它不显示任何内容。这就是为什么在video元素外提供视频的一个直接链接是一个好主意，以便最方便地访问。

最后，如果你希望视频自动开始播放，给video元素添加autoplay属性，并且给Flash param元素添加autostart= true，就像下面这样：

```
<video src="movie.mp4" width="640" height="480" autoplay>
  <param name="flashvars" value="autostart=true&controlbar=over&
    image=poster.jpg&file=yourmovie-main.mp4">
```

请记住，视频不会在iOS设备上自动播放，即便你在代码中进行了设置。苹果在其移动设备上禁用autoplay以防止意外的数据传输。

把音频添加到网页中

如果你已经熟知视频标记的例子，你肯定已经知道如何添加音频到一个页面。audio元素使用与video元素相同的属性，除了width、height和poster（因为没有东西可以显示）。就像视频元素，你可以使用source元素来提供很多音频格式选项，如这里的例子所示。

```
<audio id="soundtrack" controls preload="auto">
  <source src="soundtrack.mp3" type="audio/mp3">
  <source src="soundtrack.ogg" type="audio/ogg">
  <source src="soundtrack.webm" type="audio/webm">
</audio>
<p>Download the Soundtrack song:</p>
<ul>
  <li><a href="soundtrack.mp3">MP3</a></li>
  <li><a href="soundtrack.ogg">Ogg Vorbis</a></li>
</ul>
```

如果你想成为邪恶的人，你可以在一个页面中嵌入音频，将其设置为自

警告：如果你的服务器无法进行适当的配置来报告你的视频和音频文件的视频类型（即它的MIME类型），一些浏览器将不会播放它们。每种格式的MIME类型都在表10-4和表10-5的Type列列出。所以如果你打算提供媒体文件，请确保通知你的服务器管理员或者托管公司的技术人员，并且将MIME类型设置准确。

```
<audio>...</audio>
```

给网页添加一个音频文件

动播放和循环，并且不提供任何阻止它的控制方法，就像这样：

```
<audio src="soundtrack.mp3" autoplay loop></audio>
```

警告：Firefox版本7 及之前的版本不支持loop属性。

但是，你永远不会做类似的东西，对不对？当然，你肯定不会这么做。

canvas

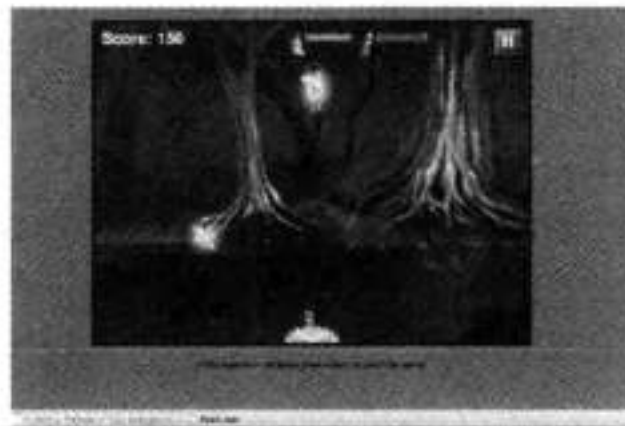
HTML 5中的另一个很酷、而且无需插件的是canvas元素和相关的API。canvas元素在网页上创建一个区域，你可以使用一组JavaScript函数进行绘制，比如创建线条、形状、填充、文字、动画等。你可以使用它来显示一个图示，但canvas元素之所以如此有潜力是因为它都是脚本产生的。这意味着它是动态的，可以绘制图像，并响应用户的输入。这使其成为一个极好的平台，可以创建动画、游戏，甚至是整个应用程序。所有这些都可以使用浏览器自有的行为，并且无需像Flash一样的专有的插件。

好消息是，写这篇文章的时候，当前的浏览器都支持canvas元素，除了Internet Explorer 8及更早的版本。幸运的是，FlashCanvas的JavaScript库（*flashcanvas.net*）可以使用Flash的绘图API给这些浏览器增加对canvas的支持。因此，Canvas已经整装待发。

图10-3给出了几个例子，都是使用canvas元素来创建游戏、画图程序、一个互动的分子结构工具以及小行星的动画。你可以在*Canvasdemos.com*找到更多的例子。



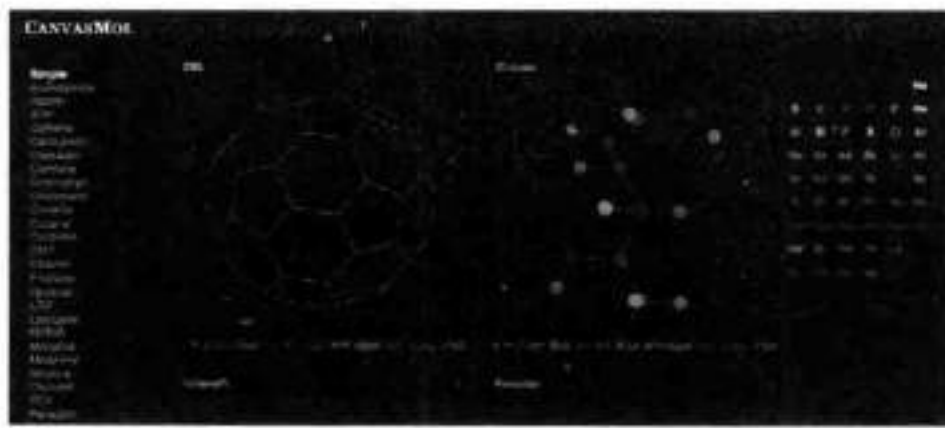
ie.microsoft.com/testdrive/Performance/AsteroidBelt/#



www.refind.com/game/magician.html



muro.deviantart.com



alteredqualia.com/canvasmol/

图10-3：使用canvas元素来做游戏、动画和应用程序的几个例子

要想掌握canvas元素，本书是不够的，尤其是我们还没有任何JavaScript的经验，但我会给你一个机会，让你试着使用JavaScript来绘画。这会让你了解它是如何工作的，也会理解这些例子的复杂度。

canvas元素

你可以使用canvas元素来给页面添加一个画布空间，并指定width和height属性。这些加起来标记才完整。对于不支持canvas元素的浏览器，你可以给标签提供一些备用内容（信息、图像或任何适当的信息）。

```
<canvas width="600" height="400" id="my_first_canvas">
  Your browser does not support HTML 5 canvas. Try using Chrome,
  Firefox, Safari or Internet Explorer 9.
</canvas>
```

该标记将在绘图的地方开辟出空间。

用Javascript绘图

canvas API包括创建基本形状的函数（如绘制矩形轮廓的strokeRect()和用于启动画线的beginPath()）和用于移动的函数（如rotate()和scale()），以及应用样式属性（例如，lineWidth、strokeStyle、fillStyle和font）。

下面的例子是从O'Reilly Media的同事Sanders Kleinfeld的书《HTML 5 for Publishers》（O'Reilly出版）中摘录的。他非常好，让我在这里使用它。

图10-4显示了简单的笑脸，我们将使用Canvas API来完成。

这里是创建它的脚本。不要担心，你不需要了解任何JavaScript。你只需通过脚本引擎执行并适当留意。在末尾，我还将介绍一些应用功能。我认为你只需要懂得它的要点就足够了。

```
<script type="text/javascript">
window.addEventListener('load', eventWindowLoaded, false);
function eventWindowLoaded() {
  canvasApp();
}

function canvasApp(){
var theCanvas = document.getElementById('my_first_canvas');
var my_canvas = theCanvas.getContext('2d');
my_canvas.strokeRect(0,0,200,225)
  // to start, draw a border around the canvas

  //draw face
my_canvas.beginPath();
my_canvas.arc(100, 100, 75, (Math.PI/180)*0, (Math.PI/180)*360, false);
  // circle dimensions
```

<canvas>...</canvas>

添加一个2-D的动态绘画区域



图10-4：完成后的“你好，Canvas”例子，可以在examples.oreilly.com/0636920022473/my_first_canvas/my_first_canvas.html中获取


```

my_canvas.strokeStyle = "black"; // circle outline is black
my_canvas.lineWidth = 3; // outline is three pixels wide
my_canvas.fillStyle = "yellow"; // fill circle with yellow
my_canvas.stroke(); // draw circle
my_canvas.fill(); // fill in circle
my_canvas.closePath();

    // now, draw left eye
my_canvas.fillStyle = "black"; // switch to black for the fill
my_canvas.beginPath();
my_canvas.arc(65, 70, 10, (Math.PI/180)*0, (Math.PI/180)*360, false);
    // circle dimensions
my_canvas.stroke(); // draw circle
my_canvas.fill(); // fill in circle
my_canvas.closePath();

    // now, draw right eye
my_canvas.beginPath();
my_canvas.arc(135, 70, 10, (Math.PI/180)*0, (Math.PI/180)*360, false);
    // circle dimensions
my_canvas.stroke(); // draw circle
my_canvas.fill(); // fill in circle
my_canvas.closePath();

    // draw smile
my_canvas.lineWidth = 6; // switch to six pixels wide for outline
my_canvas.beginPath();
my_canvas.arc(99, 120, 35, (Math.PI/180)*0, (Math.PI/180)*-180, false);
    // semicircle dimensions
my_canvas.stroke();
my_canvas.closePath();

    // Smiley Speaks!
my_canvas.fillStyle = "black"; // switch to black for text fill
my_canvas.font = '20px _sans'; // use 20 pixel sans serif font
my_canvas.fillText ("Hello Canvas!", 45, 200); // write text
}
</script>

```

最后，下面是本例所使用的Canvas API函数的更多信息：

strokeRect(x1, y1, x2, y2)

从点 (x1, y1) 到 (x2, y2) 绘制矩形轮廓。默认情况下，(0,0) 点是画布左上角的点，x和y表示的坐标分别向右和向下计算。

beginPath()

开始画线。

closePath()

如果开始时使用beginPath()来画线，就要使用它来结束画线。

arc(x, y, arc_radius, angle_radians_beg, angle_radians_end)

以 (x, y) 为圆心画一个圆弧，arc_radius是圆的半径长度，angle_radians_beg和_end表示的圆弧角度的开头和结尾。

stroke()

画线定义的路径。如果没有用这个函数，路径将不会出现在画布上。

fill()

填充beginPath()和endPath()指定的路径。

fillText(your_text, x1, y1)

在画布 (x, y) 坐标添加指定的文本。

此外，下面的属性用来指定颜色和样式：

lineWidth

路径边界的宽度。

strokeStyle

边框的颜色。

fillStyle

创建路径所形成形状的填充颜色（内部）。

Font

文本的字体和大小。

当然，`canvas` API包含了很多其他的函数和属性，远超过我们在这里所使用的。有关完整列表，请参阅W3C的HTML 5的Canvas2D环境规格[dev.w3.org/HTML 5/2dcontext/](http://dev.w3.org/HTML5/2dcontext/)。在网络上搜索也可以找到大量的Canvas教程，你应该准备好进一步学习。此外，我还推荐这些资源：

- Steve Fulton和Jeff Fulton (O'Reilly Media) 的书《HTML 5 Canvas》。
- 或观看视频，如果你想更快地学习，尝试这些资源：Client-side Graphics with HTML 5 Canvases: An O'Reilly Breakdown (shop.oreilly.com/product/0636920016502.do)。

小结

现在你关于HTML 5应该有一个好的了解。我们已经了解了新的文档语义元素。你已经了解了各种能给浏览器赋予有用的功能的API。你学会了如何使用视频和音频元素，在页面上嵌入媒体（加上一个引物媒体格式）。最后，你还了解了`canvas`元素。

这本书在接下来的第三部分，你将学习如何写自定义页面外观的样式表，包括文本样式、颜色、背景，甚至页面布局。届时就可以对浏览器的默认风格说再见了！

自我检测

让我们来看看你是否仔细学习本章内容。这些问题可以测试你是否明白本章的重要知识点。祝你好运！与前面一样，答案在附录A中。

1. HTML和XHTML之间的区别是什么？
2. 使用侧栏“XHTML标记需求”为指导，将这些HTML元素改写为XHTML语法。
 - a. `<H1> ... </H1>`
 - b. ``
 - c. `<input type="radio" checked>`
 - d. `<hr>`
 - e. `<title>Sifl & Olly</title>`
 - f. ``
 `popcorn`
 `butter`
 `salt`
 ``
3. 什么是DTD？
4. 至少以三种方式命名：HTML 5是独一无二的规范。
5. 什么是“全局属性”？
6. 用以下函数匹配API：
 网络工作者_____ a. 使经度和纬度信息可见
 编辑API_____ b. 保持服务器和浏览器之间的通信开放

- API定位_____ c. 让Web应用程序起作用，即使没有互联网连接
- 网络接口_____ d. 在后台运行处理器密集型脚本
- 离线应用_____ e. 提供的一组命令用于复制、粘贴和文本格式

7. 识别如下的容器格式、视频编解码器或音频编解码器

- Ogg _____
- H.264 _____
- VP8 _____
- Vorbis _____
- WebM _____
- Theora _____
- AAC _____
- MPEG-4 _____

8. 列出两个画布API函数来绘制一个矩形并填充为红色。注意，你不需要编写整个脚本。

表现层的CSS

第三部分

本部分内容

第11章

CSS入门

第12章

格式化文本（使用更多选择器）

第13章

颜色和背景（附加更多选择器和外部样式表）

第14章

盒子思想（填充、边框和空白边）

第15章

浮动与定位

第16章

使用CSS进行网页布局

第17章

过渡、变换和动画

第18章

CSS技术

第11章 CSS入门

你已经看到样式表被多次提及，现在，我们开始将它付诸实践，下面开始给我们的网页添加一些必需的样式。层叠样式表（Cascading Style Sheet，CSS）是用来定义用HTML，事实上，任何XML）语言编写的文档的表现层的W3C标准。表现层与文档显示或呈现给用户的方法相关，不论是显示在计算机屏幕、移动电话显示屏上，印刷在纸上，还是由屏幕阅读器大声朗读。用样式表处理表现层，HTML可以按预想的方式来定义文档结构和大意。

CSS是一种独立的语言，有自己的语法。本章内容涵盖了CSS术语和基本概念，这些将帮助你为后续的章节做准备，在后续你将学习：如何使用CSS来改变文本和字体样式、添加颜色和背景、进行基础的网页布局。我的目标就是，到第三部分结束的时候，为你们能够自己进行深入的阅读以及大量的实践打下扎实的基础。

CSS的益处

下面不是告诉你样式表为什么是正确的，而是告诉你使用样式表的好处。

- 精准的样式和对布局的控制。通过使用CSS，你可以达到如同印刷一般的精准效果。甚至有一部分内容是专门用于打印页面的（但是在本书中不会涉及）。
- 更少的工作。你可以通过改变一个样式来改变整个网站的显示效果。
- 更易用的网站。当表现层的所有事务都由CSS处理之后，你可以更有意义地标记内容，使它对非可视化或移动设备更易用。
- 可靠的浏览器支持。当前几乎每个浏览器都支持CSS二级的全部标准和CSS三级的大部分标准（CSS中“级”的意思参见侧栏“认识标准”）。

本章内容

层叠样式表（CSS）
的好处和能力

如何用HTML标记
创建文档结构

编写CSS样式规则

将样式附加到HTML文档

CSS的大概念：继承、层叠、
特性、规则顺序和盒子模型

回头想想，使用样式表的确没有任何缺点。虽然后续有一些浏览器一致性的争论，但如果你知道哪里去找这些问题，就都可以避开或解决。

CSS的力量

我们不是在谈论小的视觉调整，比如，改变标题颜色或添加文本缩进。当发挥它的全部潜能时，CSS是一个强大的设计工具。《CSS Zen Garden》（www.csszengarden.com）中设计的多样和丰富让我眼界大开。

在过往的岁月中,当开发人员对是否该放弃他们CSS的表格布局仍犹豫不决时,David Shea的CSS禅意花园网站就准确地展示了到底是什么可以独自完成使用CSS。David通告了HTML文档的相关内容,并邀请设计师贡献他们自己的样式表，这些样式表给HTML文档带来了视觉设计效果。图11-1显示了我最喜欢的几个设计。这些设计采用的都是完全相同的HTML源码文档。



CSS Zen Dragon
by Matthew Buchanan



Shaolin Yokobue
by Javier Cabrera



By the Pier
by Peter OngKelmScott



Organica Creativa
by Eduardo Cesario

图11-1：CSS Zen Garden中的这些网页使用相同的XHTML源码文档，但使用专门的CSS来改变设计（CSS Zen Garden和设计师许可使用本图）

不仅如此，它还不包含单独的元素（所有的图像都用做背景）。但看看这些网页看起来多么不一样——多么奇妙。这都是用样式表做到的。在保持CSS独立于HTML以及表现层独立于结构的方面，这是个有力的证明。

CSS禅意花园不再进行更新，它现在被认为是一个具有转折点意义的历史文档，该转折点主要体现在采用Web标准上。尽管它年代久远，但我仍然觉得在准确演示CSS的功能方面，禅意花园是一个不错的一站式教程。

我承认，需要很多实践才能创建如图11-1所示的CSS布局。杀手级图像设计技术也有帮助（不过本书中不会学习）。因为我想让你知道基于CSS的设计的潜力，特别是因为入门书中的例子往往比较简单和直观，所以我把这个例子给你看。你需要多花时间来学习，并且永远关注重点。

样式表如何工作

样式表的工作就像1-2-3一样容易！

1. 以一个用HTML标记的文档开始。
2. 按照你想象中某些元素的样子来编写样式规则。
3. 将样式规则附加到文档。当浏览器显示文档时，它遵从你的规则来显示元素（除非用户应用了某些强制样式，但是这个我们后面再讲）。

当然，除了这些之外，还有一些。我们接下来考虑每一步的细节。

1. 标记文档

从前面的章节中你可以知道很多标记内容的方法。例如，选择准确描述内容意思的元素很重要。你也听我说过，标记创建了文档的结构，有时称为结构层，表现层可以应用在结构层之上。

在本章及后续章节中，你将看到，理解文档结构和元素关系是作为样式表作者所完成工作的重中之重。

想体验一下用样式表改变文档外观多么简单，只需要在练习11-1中亲自动手。为了方便练习，我已经准备好一些你要用的XHTML文档了。

练习11-1 你的第一个样式表

本练习中，我们将给一个小文章添加一些简单的样式。*twentis.html*文档和相关图像*twenty_20s.jpg*可以在www.learningwebdesign.com/4e/materials/找到。首先，在浏览器中打开文档，查看它的默认外观（有些如图11-2所示）。你还可以在文本编辑器中打开文档，以准备在接下来的练习中学习操作。

The Back of the New \$20

Have you seen the "Series 2004 \$20 Notes"? The U.S. Treasury has rolled out yet another revamp of the U.S. twenty dollar bill in an effort to stop those pesky counterfeiters once and for all. It features high-tech fake-busting elements like a watermark, a security thread, and color-shifting ink. It also features crappy design.

I'm not going to concern myself here with a critique of the front of the bill (my friend Jeff says "it looks like something got spilled on it."). It's the back of the note that's driving me crazy.

Too Many 20s



In particular, it's all those little 20s haphazardly sprinkled in the white space. They are nails-on-a-chalkboard to my visual design senses.

Are they supposed to be another security feature? ("They'll NEVER be able to duplicate this \$20... look at those 20s... they're all OVER the place!") Did they let a summer intern at the Bureau of Engraving and Printing design it? ("Hey, let Jimmy try it!") Were they concerned the \$20 bill might be confused with a \$10? ("What this 20 needs is a LOT more 20s.")

Connect-the-Dots

There must be more to it. My theory: the new 20s contain subliminal connect-the-dots messages, like tiny constellations. So, perhaps the 20s connect to form a secret message designed to stimulate the economy ("SPEND MORE") or boost patriotism ("WE'RE NO.1").

I'm not sure I've successfully cracked the code, so I'm asking for your help. I encourage you all to get a new \$20 bill, connect the dots to find the message on the back (pencil is best), and mail it to me for review. Together, we can get to the bottom of this.

图11-2：没有任何样式表指令时文章的外观。虽然我们并不准备把它变漂亮，但你可以感觉到样式是如何工作的

2. 编写规则

样式表由一个或多个样式指令（又叫做规则）组成，这些指令描述了元素或元素组将如何显示。学习CSS的第一步是熟悉部分规则。如你所见，它们相当需要直觉。每条规则选择一个元素并声明它的外观。

下面的例子包含两条规则。第一条使文档中所有的h1元素变绿；第二条指定段落的字体为小的sans-serif字体。

```
h1 { color: green; }
p { font-size: small; font-family: sans-serif; }
```

在CSS术语中，规则包括两个主要的部分，一个是用来标识受影响的一个或多个元素的选择器（selector），另一个是提供显示指令的声明（declaration）。声明依次由属性（如color）和它的值（如green）组成，由冒号和空白隔开。一个或多个声明都置于花括号内，如图11-3所示。

注意： Sans-serif字体在线条端点没有小的衬线（serif），往往看起来更圆滑、更现代。我们将在第12章中详细讨论。

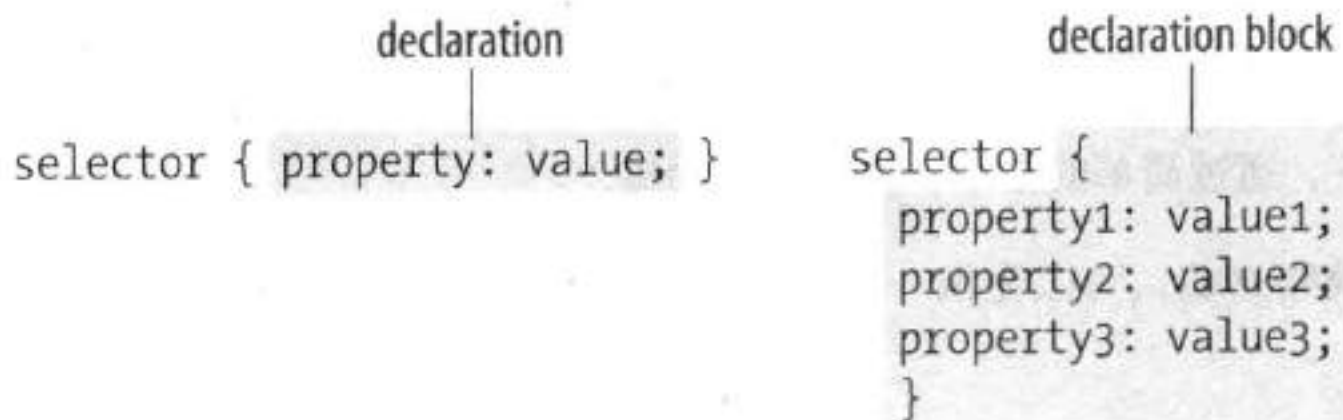


图11-3：部分样式表规则

选择器

在上一个小样式表的例子中，h1和p元素都用作选择器。最基本的选择器类型称为元素类型选择器。每个规则定义的属性将分别应用于文档中的每个h1和p元素。在后续章节中，我将给你介绍更精妙的用于选取元素的选择器，包括选择出现在具体上下文中的元素和元素组。

精通选择器，即选择最好类型的选择器并策略性地使用。这是成为CSS高手的重要步骤。

声明

声明由属性/值对组成，在一个规则中可以有不止一个声明；例如，上面的p元素的声明有font-size和font-family两个属性。每个声明都必须以分号结束，来与下一条声明隔开（见“注意”）。如果你省略了分号，这条和下条声明都会被忽略。规则中包含的花括号和所有声明常统称为声明块（图11-3）。

因为CSS忽略声明块中的空白和回车，所以作者通常将块中的每条声明写到独自的行中，如同下例中，这样就可以很容易找到应用于选择器的属性，并知道什么时候样式规则结束。

```

p {
  font-size: small;
  font-family: sans-serif;
}
  
```

注意，实际上这里没有改变——依然有一对花括号，每条声明后面都有分号等。唯一的不同是插入了一些换行和用于对齐的空白字符。

样式表的核心在于能应用于标准属性的集合。完整的CSS规范为所有元素定义了几十种属性，从文本缩进到表头如何被读出来。本书涵盖了大部分常用并受到良好支持的属性。

注意：技术上讲，块中最后一个声明之后不需要分号，但我建议你养成用分号结束声明的习惯。这为在规则后面再添加声明提供方便。

提供计量值

当提供计量值时，单位应该紧跟着数字，如下：

```
margin: 2em;
```

在单位之前加空白，将导致属性失效。

```
错误：margin: 2 em;
```

可以忽略0值的计量单位：

```
margin: 0;
```


值由属性决定。一些属性采用长度计量值，一些采用颜色值，其他的有预定的关键字列表。当使用属性时，需要知道它接受哪些值；然而，很多情况下，通过简单的常识，你就可以知道。

在我们继续之前，可以先做一些实践，请接着练习11-1再编写一些样式规则。

练习11-1 你的第一个样式表（续）

在文本编辑器中打开`twenties.html`，在文档的`head`部分，你可能发现，我已经给你建立包含样式规则的`style`元素。`style`元素用于将样式表嵌入HTML文档的页头。

开始，我们只添加本节中刚看到的小样式表。把下列规则写入文档，如你所见。

```
<style type="text/css">
h1 {
  color: green;
}
p {
  font-size: small;
  font-family: sans-serif;
}
</style>
```

保存文件并在浏览器中查看。你将注意到一些变化（如果浏览器已经使用`sans-serif`字体，那你可能只看到尺寸的变化）。如果没有，回去检查是否包含正反花括号和分号。很容易不小心丢失这些字符，而导致样式表失效。

现在，我们将要作变动，添加样式表来看看写样式表是多么容易，再看看改动的效果。下面是一些可以尝试的（记住：为了在每次浏览器重载后看到变化，你需要在每次修改后保存文档）。

- 使`h1`元素变“灰（gray）”并在浏览器中查看。再将它变“蓝（blue）”。最后，变为“红（red）”。（我们使用第13章中可用颜色名完全列表中的颜色。）
- 添加新规则，使`h2`元素也为红色。
- 使用下列声明给段（`p`）元素添加100像素的左空白边。

```
margin-left: 100px;
```

记住你可以把这个新的声明添加到`p`元素的已有规则中。

- 给`h2`标题的左空白边添加100像素。
- 使用下列声明给`h1`元素底部添加红色、1像素的边框：

```
border-bottom: 1px solid red;
```

- 使用`float`属性将图像移到空白边右边，并使其可以被文本环绕。下列规则中的快捷`margin`属性可以给图像添加上下0像素、图像左右各12像素的空白（与第14章中的风格相同）。

```
img {
  float: right;
  margin: 0 12px;
}
```

当所有的完成之后，文档将如图11-4所示。

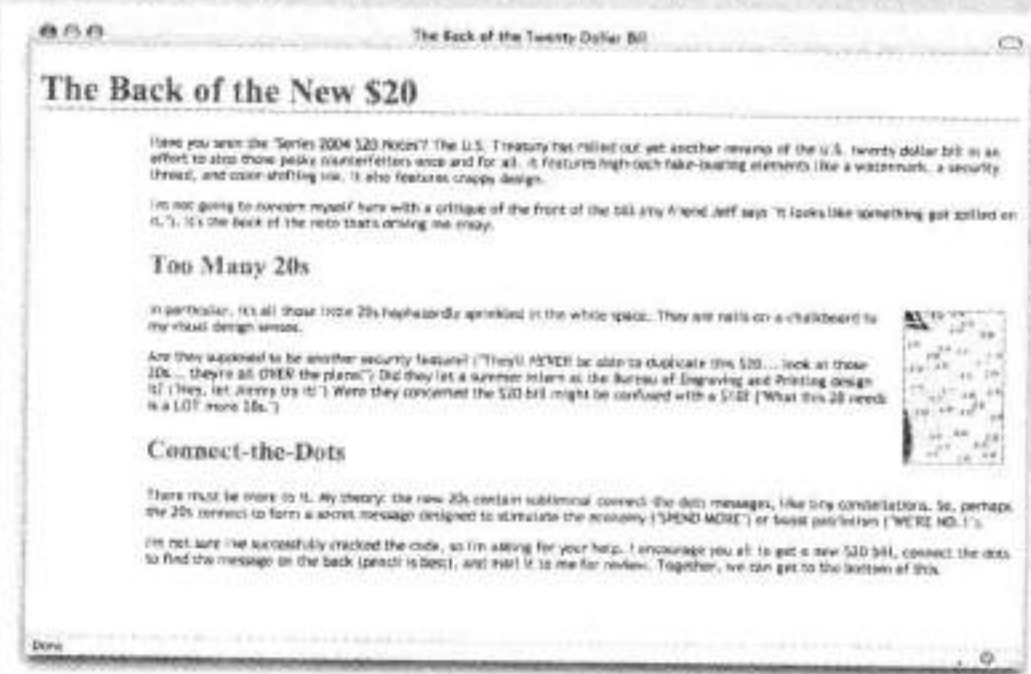


图11-4：添加本例中小的样式表之后的文章。不漂亮，但焕然一新

3.将样式附加到文档

在上个练习中，我们使用了style元素，将样式表嵌入文档内。有三种方法可以将样式信息应用到HTML文档，这只是其中一个。你可以马上一一尝试，但预览这些方法和术语会很有帮助。

外部样式表。外部样式表是一个独立的、纯文本的包含很多样式规则的文档。必须使用.css后缀名。然后.css文档链接或导入一个或多个HTML文档中（我们将在第13章讨论如何做）。用这种方法，网站中的所有文件可以共享样式表。这是最强大也是推荐的附加样式表到内容的方法。

嵌入样式表。这是我们在练习中用到的样式表的类型。它位于文档中，使用style元素，只对该文档有效。style元素必须位于文档的开头。例子同样包含注释（见侧栏“样式表中的注释”）。

```
<head>
  <title>required document title here</title>
  <style >
    /* style rules go here */
  </style>
</head>
```

样式表中的注释

有时，在样式表中做一些工作注释是有用的。CSS有自己的注释语法，如下所示：

```
/* comment goes here */
```

当样式表被解析时，/* 和*/之间的内容将被忽略，这意味着，你可以在样式表的任何地方留下注释，包括在规则中。

```
body {
  font-size: small;
  /* front-size:large; */
}
```

注释的一个作用就是对样式表的部分内容进行标注，使得以后查找起来更方便，例如：

```
/* Layout style */
或
/* FOOTER STYLE */
```

CSS的注释也有助于将样式声明暂时隐于设计过程中。当我尝试大量的样式时，我可以通过将这些样式包含在/*和*/中从而实现快速切换，在浏览器中进行核对，然后删除注释属性使得样式再次出现。这远远比重新输入一连串指令要快得多。

注意：在HTML 4.01和XHTML 1.0/1.1中，style元素必须包含能识别其内容的type属性：

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

在HTML 5中，已经不包含type属性了。

style元素可能包含在media特性中，用来给具体的媒介（比如，显示屏、打印或手持设备）选取样式。这些也将在第13章进行讨论。

内联样式。你可以使用元素本身的style特性，在单个元素上应用属性/值对，如下：

```
<h1 style="color: red">Introduction</h1>
```

要添加多个属性，只需要将它们用分号隔开，如下所示：

```
<h1 style="color: red; margin-top: 2em">Introduction</h1>
```

内联样式只应用于它们所在的特殊元素。尽量不用内联样式，除非必须覆盖嵌入样式表或外部样式表中的样式。当内联样式把表现层信息散布到结构标记中时，你会发现它的问题重重。内联样式很难修改，因为需要找到源码中每一个style特性。

练习11-2给了我们一个编写内联样式的机会，并观察工作机制。由于早先列出的理由，在此之后我们将不再使用内联样式，所以这是你的机会哦。

大概概念

有一些大的思想你需要理解，从而熟知层叠样式表的行为。现在我将给你介绍这些概念，尽管我们刚才还在讲述样式属性，但我们的演讲必须慢下来。这些思想必然会在后续章节中再次遇到并详细描绘。

继承

你眼睛的颜色和你父母的相同吗？你继承了他们的发色吗？你是单眼皮吗？嗯，就像父母把他们的性状传递给他们的孩子一样，HTML元素也会把某些样式属性传递给它包含的元素。注意练习11-1中，当定义p元素的样式为小的sans-serif字体时，第二段的em元素也变小、字体也变为sans-serif了，即使我们没有特意为它编写样式规则（图11-5）。这是因为它继承了它所在段落的样式。

Unstyled paragraph It's the **back** of the note that's driving me crazy.

Paragraph with style rule applied

```
p {font-size: small; font-family: sans-serif;}
```

It's the **back** of the note that's driving me crazy.

The emphasized text (em) element is small and sans-serif even though it has no style rule of its own. It *inherits* the styles from the paragraph that contains it.

图11-5：em元素继承了应用于段落的样式

练习11-2 应用内联样式

打开文章twenties.html，保持练习11-1得到的最后结果。如果你能完全完成，你将有一个给h2元素应用颜色的规则。

现在，写一个内联样式，设置第二个h2元素为粉红色。通过在h2开标签中使用下列style特性，来完成任任务。

```
<h2 style="color: gray">
Connect-the-Dts</h2>
```

保存文件并在浏览器中打开它。现在第二个标题是灰色的，在嵌入的样式表中覆盖红色集。另一个h2标题则不受影响。

文档结构

这里需要理解你的文档结构。如我之前所提过的，HTML文档有隐性的结构或层次。例如，我们使用的示例文章有一个html根元素，根元素包含head元素和body元素，而body元素又包含标题和段落元素。以此类推，一些段落包含内联元素，比如图像（img）和强调文本（em）。你可以把结构看做一个倒置树，从根来分枝，如图11-6所示。

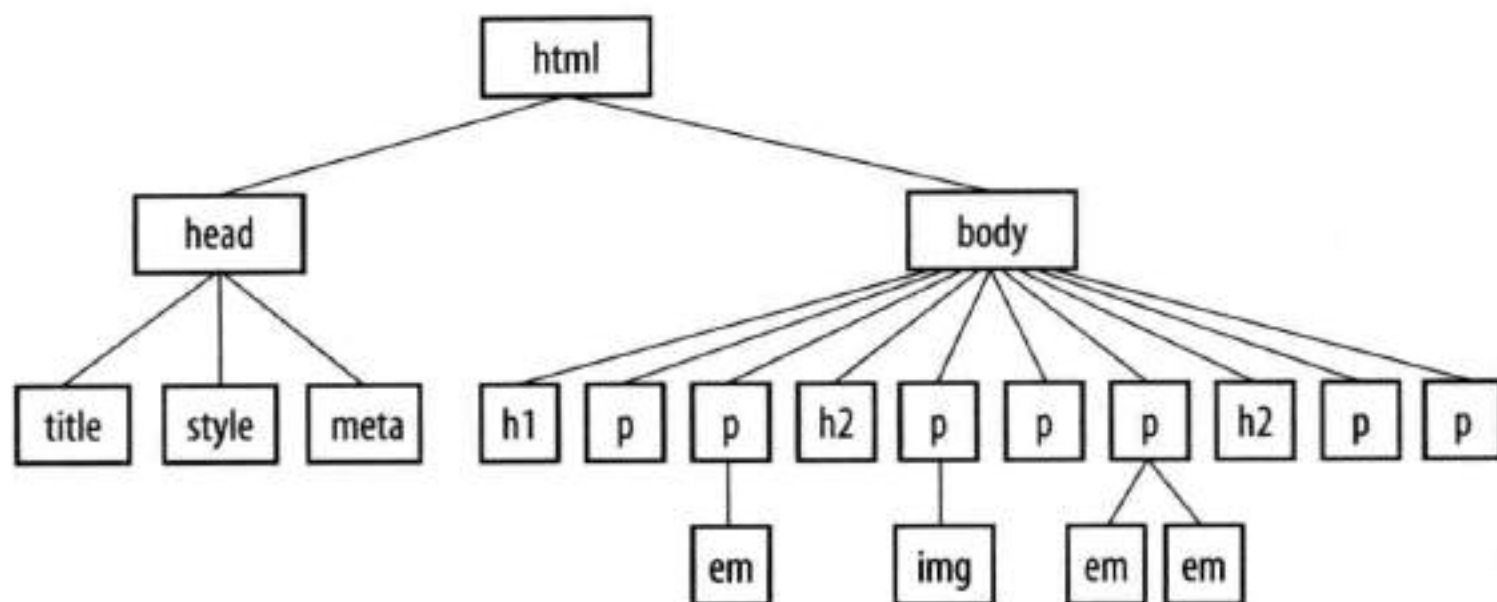


图11-6：示例文档twenties.html的文档树结构

父与子

联想起元素间的关系，文档树变成了一个家族树。包含在指定元素里的所有的元素都称为后代。例如，图11-6中文档中所有h1、h2、p、em和img元素都是body元素的后代。

一个元素直接包含在另一个元素中（之间没有其他级别），前者称为后者的子代，相反，后者为前者的父代。例如，em元素是p元素的子代，而p元素是它的父代。

所有比特定元素级别高的元素在层级上都可以称为它的祖先。有相同父代的两个元素互称同胞。我们不讨论“阿姨”和“表亲”，类推到此为止。这个可能看起来比较理论化，但在写CSS选择器时将会用到。

传递下去

当你使用p元素作为选择器编写一个字体相关的样式规则时，规则将应用于文档中所有的段落和这些段落包含的内联文本。回到图11-5，可以看到em继承应用于它的父代（p元素）的样式属性。图11-7使用文档结构图来描绘发生了什么。注意img元素被排除在外，因为字体相关的属性不能应用于图像。

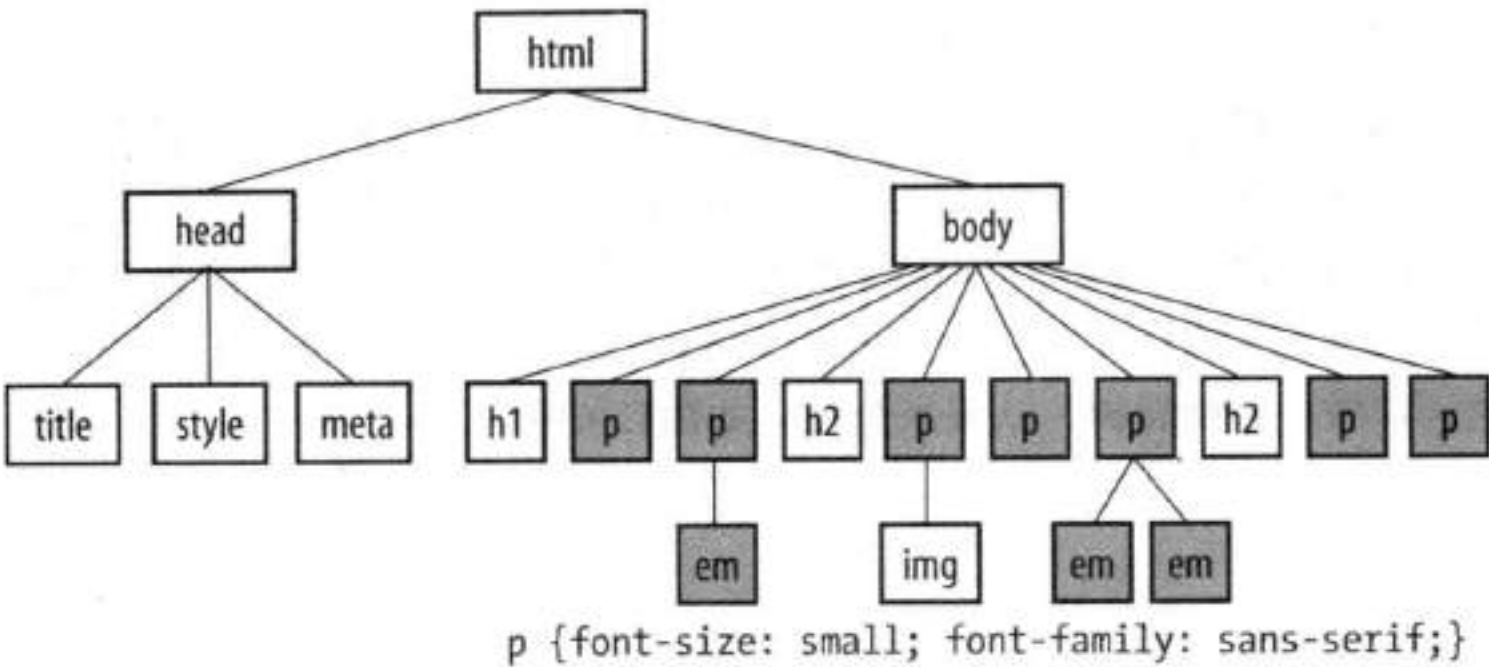


图11-7：某些应用于p元素的属性被它们的子代继承

注意，我说的是“某些”属性被继承。注意到这点很重要，有些样式表属性能被继承，而另一些不能。通常，与文本样式相关的属性：`font-size`、`color`、`style`等，都可以传承。如`border`、`margin`、`background`等影响元素周围面积的属性往往不能传承。你自己想想，会发现很有意义。例如，如果你在段周围加边框，你并不希望段所包含的每个内联元素（如`em`、`strong`或`a`元素）周围也有边框。

编写样式表时你可以突出继承性的优势。例如，如果你想所有的文本元素显示为Verdana字体，可以给文档中每一个元素编写单独的样式规则并设置`font-face`属性为Verdana。更好的方法是写一个样式规则，应用`font-face`属性到`body`元素，然后让包含在`body`元素中的所有文本元素继承这个样式（图11-8）。

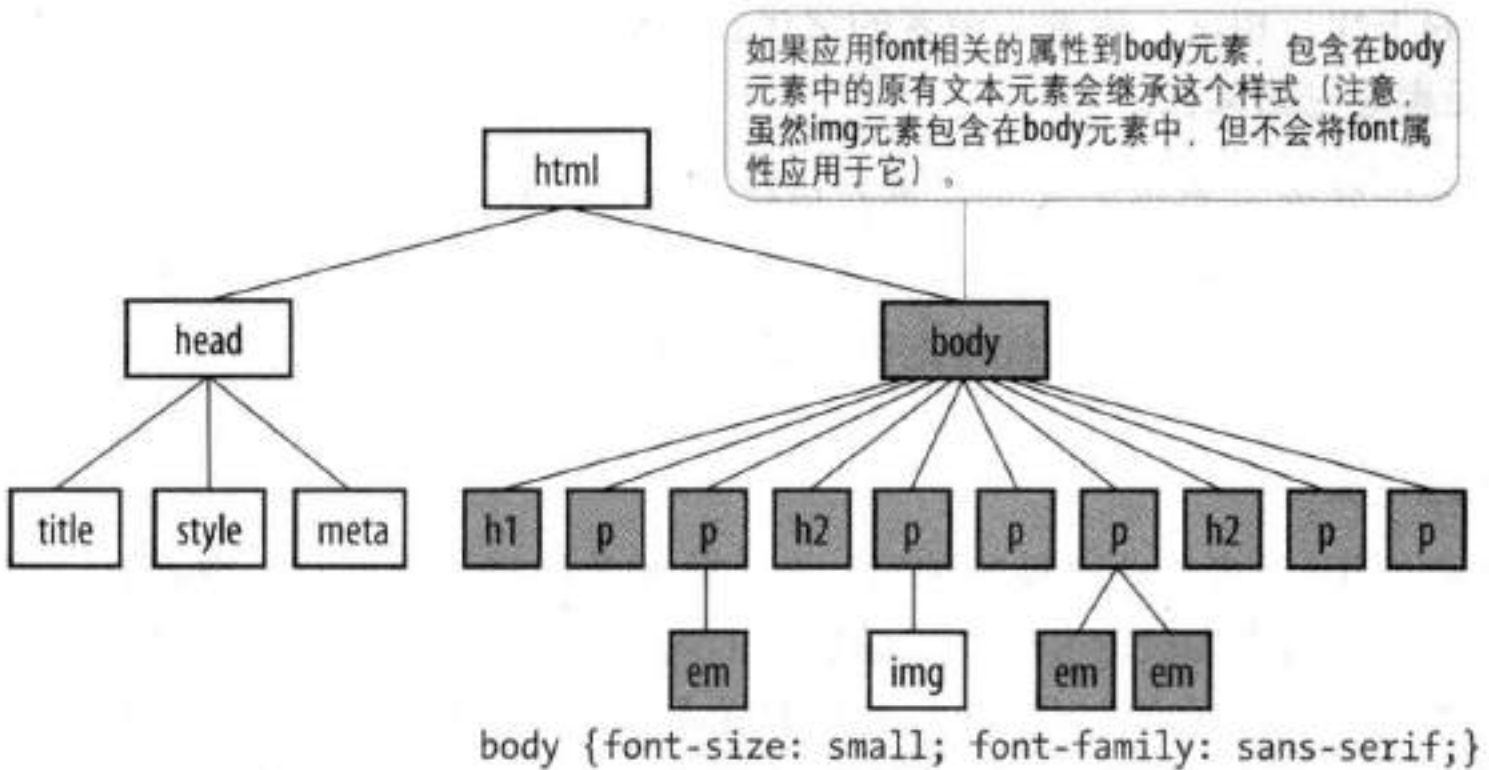


图11-8：文档中所有元素继承某些应用于body元素的属性

应用于具体元素的任何属性将覆盖该属性的继承值。回到本章的例子，我们可以指定`em`元素以`serif`字体显示，那么就会覆盖继承来的`sans-serif`设置。

CSS提示

当你学习一个新的属性时，最好留意它是否可继承。本书中列出的每个属性都注明了继承性。大多数情况下，继承性如你所料。

警告： 浏览器样式可能会覆盖body的样式，所以要小心样式出现意外。

样式冲突：层叠

曾经疑惑过为什么叫“层叠”样式表？CSS允许我们给同一个文档应用多个样式表，这意味着必将引发冲突。例如，如果文档的导入样式表说h1元素应该是红色，但它的嵌入样式表有一个规则将h1变成粉红色，那么浏览器会怎么做？

编写样式表规范的人已经预想到这个问题，并设计了一个层级系统，给不同的样式信息源分配不同的权重。层叠（cascade）与当多个样式信息源争夺网页中元素的控制权时如何处理有关：样式信息被传承下去，直到被更高权重的样式命令所覆盖。

例如，如果你没有应用任何样式信息到网页，那么它将根据浏览器内部样式表显示（称为默认显示，在W3C中称为用户代理样式表）。个人用户也可以提供自己的样式（用户样式表），会在他们的浏览器中覆盖默认值。然而，如果网页的作者为文档提供了样式表（作者样式表），那么就会把用户和用户代理样式都覆盖。唯一的例外是，如果使用者已经将该样式识别为“重要”，此时，各用户就可以应用自己的样式到文档，如侧栏“读者样式表”中所讨论的。

样式表源是一个层级，该层级决定了哪个样式占据较大权重。我们学过，有三种将样式信息附加到源码文档的方法，它们也有一个层叠顺序。通常来说，样式表与内容越近，它的权重就越高。嵌入样式表出现在文档中的style元素内，它比外部样式表权重高。本节开始的例子中，h1元素最后将是嵌入样式表中指定的粉红色，而不是更低权重的外部.css文件中指定的红色。内联样式比嵌入样式表权重更高，因为没有什么能比元素开标签中的style特性更接近内容了。这种效果将在练习11-2中显示。

为防止指定的规则被覆盖，可以用!important指示器来指定它很“重要”，这样做的原因在侧栏“指定重要度”中解释。侧栏“样式表层级”提供了从普通到特殊的层叠顺序的概况。

优先级特性

一旦选择了可用的样式表，仍然会有冲突；因此，需要在规则层继续层叠。当同一个样式表中的两个规则冲突时，选择器的类型可以用来决定胜出者。优先级特性更高的选择器，被赋予更高的权重，以此来覆盖冲突的声明。

马上就要讨论优先级特性，因为我们只看过一种选择器（而且还是优先级特性最低的那种）。现在，使用优先级特性和一些选择器的概念来覆盖其他目标。我们将在第12章再次遇到它，那时你会知道更多的选择器类型。

当一个样式表的两个规则冲突时，选择器的类型将会决定胜出者。

指定重要度

如果你不想规则被后来的冲突规则覆盖，那么就在属性值之后、分号之前添加一个!important指示器。例如，要让段内文本总是蓝色（blue），可以使用下列规则：

```
p {color: blue !important;}
```

即使浏览器在文档后面遇到内联样式（它将覆盖文档级（document-wide）样式表），如下：

```
<p style="color: red">
```

段内文字仍然是蓝色的，因为带有!important指示器的规则不能被作者的样式表的其他规则覆盖。

覆盖这个!important规则的唯一方法是，使用读者（用户）样式表中也被标记为!important的冲突规则。它能

保证特殊读者的需求，比如对于弱视者的大字，它不会被覆盖。

基于上一个例子，如果读者的样式表包括这个规则：

```
p {color: black;}
```

文本依然是蓝色，因为所有的作者样式（即使那些没有!important标记的）都比读者的样式优先。然而，如果冲突的读者样式做了!important标记，如下：

```
p {color: black !important;}
```

那么段中文字将变黑，并且不能被任何作者提供的样式所覆盖。

样式表层级

样式信息可以有不同的来源，从通用到特殊的列表如下。列表下方的项将覆盖上面的：

- 浏览器默认设置
- 用户样式设置（在浏览器中作为“读者样式表”设置）
- 链接的外部样式表（用link元素添加）
- 导入的样式表（使用@import函数添加）
- 嵌入样式表示（由style元素添加）
- 内联样式表（由开标签中的style特性添加）
- 任何被作者标记为!important的样式规则
- 任何被读者（用户）标记为!important的样式规则

规则顺序

最后，如果相同权重的样式规则冲突，那么列表中最后出现的将是“赢”。以下面三个规则为例：

```
<style>
  p { color: red; }
  p { color: blue; }
  p { color: green; }
</style>
```

这种情况下，段落文本将为绿色（green），因为样式表中最后一条规则，即文档中最接近内容的那个规则，可以覆盖前面的规则。当冲突的样式发生在一个声明块中时，同样的事情就会发生：

```
<style>
  p { color: red;
      color: blue;
      color: green;}
</style>
```

最后的颜色是绿色，因为最后的声明覆盖了前面的两个。当你采用复合属性时，在一个规则里，很容易不小心覆盖之前的声明，对这种情况要铭记于心。

盒子模型

在讨论“CSS大概念”的时候，唯一适合于介绍CSS可视格式化系统的基础是：盒子模型。想象盒子模型的最好方法是：浏览器将网页中所有元素（块和内联元素）都看做被包含在一些矩形盒子里。你可以应用属性（比如，border、margin、padding和background）到这些盒子中，甚至在网页中重定位它们。

我们将在第14章讨论盒子模型的更多细节，但先来简单感觉下盒子模型对你将来的好处，尤其在后面两章讨论文本和背景的时候。

为了用浏览器粗略地看这些元素，我已经编写了样式规则，来给我们示例的每个内容元素添加边框。

```
h1 { border: 1px solid blue; }
h2 { border: 1px solid blue; }
p { border: 1px solid blue; }
em { border: 1px solid blue; }
img { border: 1px solid blue; }
```

图11-9显示了结果。边框突出显示了每个块元素盒子的形状。内联元素（em和img）周围也有一些盒子。注意，块元素盒子扩展并填满了浏览器窗口的可用宽度，这是块在普通文档流中的自然属性。内联盒子只包围它们包含的字符和图像。

组合选择器

嘿！这是给你展示便捷的样式规则快捷方式的好机会。如果你需要应用相同的样式属性到很多元素，那么可以把这些选择器组合成一个规则，通过逗号隔开。这一个规则前面列出的五个规则有相同的效果。组合它们将提高将来的编辑效率，并减小文件尺寸。

```
h1, h2, p, em, img { border: 1px solid blue; }
```

现在，你的工具箱里有两个选择器类型了：简单元素选择器和组合选择器。

注意：这个“后来居上”的规则同样适用于CSS中其他的上下文。另外，源码中列在后面的外部样式表优先于列在它们前面的。

突击测试

你能猜到我不只是添加border属性到body元素，而且还把它继承给组合选择器中的所有元素吗？

答案：

因为border属性是一个不可继承的属性。



图11-9：所有元素周围的规则显示了它们的元素盒子

CSS简史

CSS的第一个官方版本（CSS一级介绍，又称作CSS1）正式发布于1996年，包括向网页元素添加字体、颜色和空白指令的属性。遗憾的是，由于缺乏可靠的浏览器支持，很多年来CSS都不能被广泛采纳。

CSS二级（CSS2）发布于1998年。它最大的特点是添加了用于定位的属性，该属性让CSS可能用来布局网页。这也为定义样式引入了其他媒介类型（比如打印、手持设备和读屏器）专用的样式和更精妙的元素选择方法。CSS二级，修订版1（CSS2.1）对CSS2做了小改动，同时成为2011年最佳推荐品。

CSS三级（CSS3）不同于之前的版本，因为它被分成了若干个单模块，每个模块都有其相应的属性，如动画、多列布局和边界。虽然一些模块正处于标准化进程中，但其他的仍属于实验性的。这样，浏览器开发人员就能够开始（我们可以开始使用！）逐一使用属性，而不是等待整个规范是否“ready。”事实上，一些浏览器已经支持尚在开发中的CSS三级（CSS3）的特性，即便他们没有获得普遍认同，只要能够撤销和没有内容丢失即可。它们可以作为“frosting”用于另外一个稳定的设计（或者换句话说，作为一种强化）。

要与不同种类的CSS特征保持同步，可参见W3C的CSS Current work page (www.w3.org/Style/CSS/current-work)。

继续使用CSS

本章涵盖层叠样式表的所有基础知识，包括规则语法、应用样式到文档的方法，以及继承、层叠和盒子模型的重要概念。从现在开始，样式表将不再神秘，你只是通过添加属性和选择器到你的知识库，以及扩展这里介绍的概念，来在本章基础上构建样式表。

CSS是一个宽泛的主题，超过了本书的范围。书店和网上有很多关于各种技术级别的样式表资料。我已经汇集了在我学习期间发现的最有用的资源列表。我还提供了协助编写样式表的流行工具列表。

书籍

除了下面的书，其实还有很多关于CSS的好书，但这些书教会了我，我觉得需要推荐它们。

《Cascading Style Sheets: The definitive guide》，作者Eric Meyer (O'Reilly Media)

《CSS: The Missing Manual》，作者David Sawyer McFarland (O'Reilly Media)

《Handcrafted CSS: More Bulletproof Web Design》，作者Dan Cederholm (New Riders)

《CSS Cookbook: Quick Solution to Common CSS Problems》，作者Christopher Schmitt (O'Reilly Media)

网上资源

下面的网站是样式表在线探索之旅的好起点。

万维网协会 (www.w3.org/Style/CSS)。万维网协会指导Web技术开发，包括CSS。

A List Apart (www.alistapart.com/topics/code/css)。这本在线杂志以一些关于最先进的基于标准的网页设计的最好的思想和写作为特色。由Jeffrey Zeldman 和 Brian Platz于1998年创建。

CSS-tricks (css-tricks.com)。这是Chris Coyier关于CSS的博客。Chris喜欢CSS并最终把他的研究心得分享到了他的站点上。

CSS工具

下面是一些我个人推荐的工具。

网络开发者扩展插件

网络开发者高度评价这个由Chris Pederick编写的网络开发者扩展插件。这个扩展插件添加了一个工具条到浏览器中，工具条中的工具能让你分析和操作窗口中的任何网页。你在浏览的时候，既可以编辑网页的样式表，也可以获取HTML和图像的信息。它还可以验证CSS、HTML，以及网页的可访问性。同时，它可用于Chrome、Firefox和Mozilla浏览器。你可以从chrispederick.com/work/

*webdeveloper*网页获得这个工具。注意，Safari有类似的内置检查器（去开发→显示网络检查器）。

Web写作程序

当前所见即所得的写作程序（如Adobe Dreamweaver和Microsoft Expression Web）可以配置为，在你设计网页时自动编写样式表。缺点是它们并不总是用最高的效率来编写样式表（例如，它们过度使用class特性来创建样式规则）。尽管如此，它们给样式表设计一个好的开头，之后你就可以手动编辑了。

自我测验

下面的问题检查你的CSS基础知识。答案提供在附录A中。

1. 识别下面规则的不同部分：

```
blockquote { line-height: 1.5; }
```

选择器：_____ 值：_____

属性：_____ 声明：_____

2. 当应用嵌入样式表到文档时，段落将变为什么颜色？为什么？

```
<style type="text/css">
  p{color:purple;}
  p{color:green;}
  p{color:gray;}
</style>
```

3. 重写下列每个CSS例子。其中一些完全错误，一些可以写得更高效。

```
a. p {font-family: sans-serif;}
   p {font-size: 1em;}
   p {line-height: 1.2em;}
```

```
b. blockquote {
   font-size: 1em
   line-height: 150%
   color: gray }
```

```
c. body
   {background-color: black;}
   {color: #666;}
   {margin-left: 12em;}
   {margin-right: 12em;}
```

```
d. p {color: white;}
   blockquote {color: white;}
   li {color: white;}
```

```
e. <strong style="red">Act now!</strong>
```

4. 当下列样式规则应用于如图11-10结构的文档时，在图表中圈出你

认为将显示为红色的所有元素。这条规则使用了你从未见过的选择器，但你可以通过常识来判断。

```
div#intro { color: red; }
```

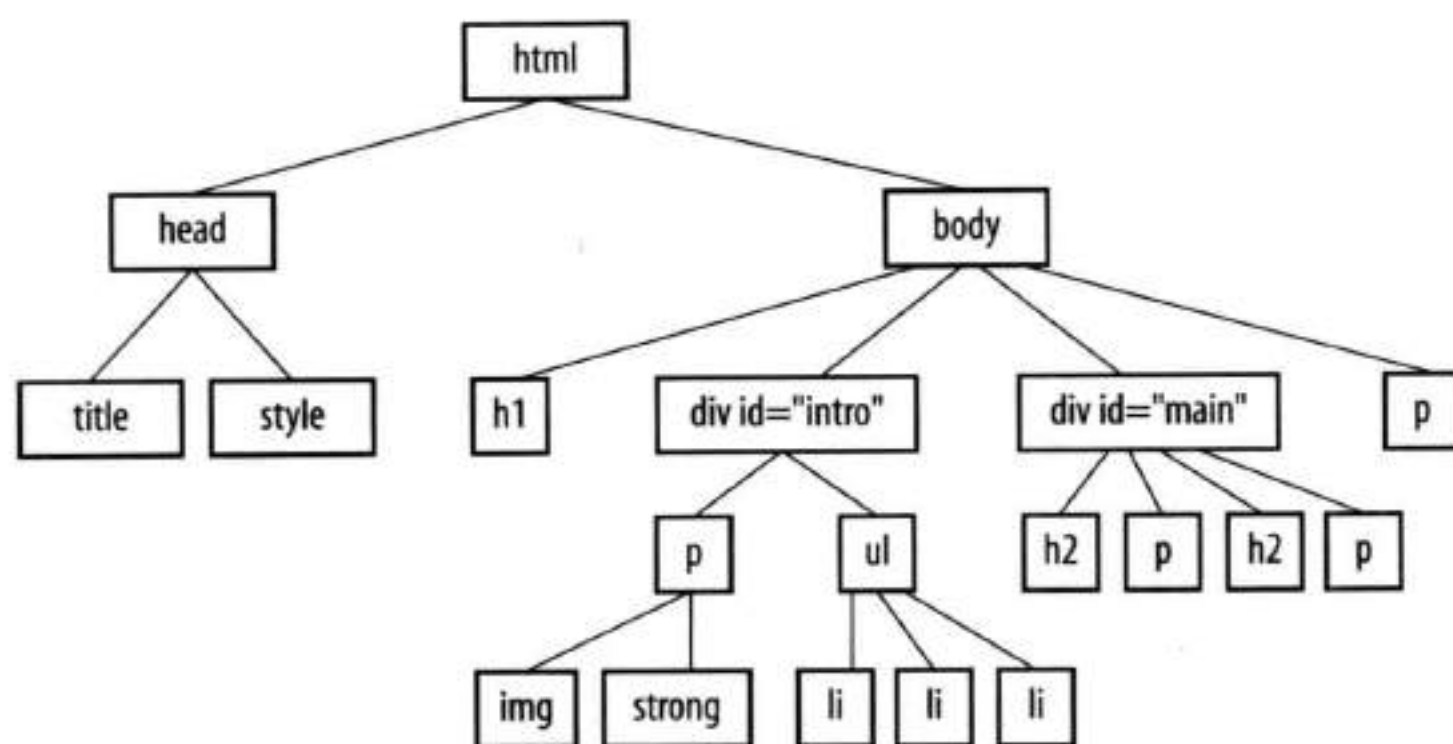


图11-10：示例文档的文档结构

第12章 格式化文本（使用更多选择器）

既然你已经学习了格式化文本的基础知识，你是否已经准备好更深入地学习了呢？到本章结束，你将学习15个新的用来操作文本外观的CSS属性。在此过程中，你同时将学习如何对带有特殊id和class属性以及处于特定上下文的目标元素使用强大的选择器。

Web的自然属性使指定类型复杂，听起来很令人沮丧，特别是如果你有印刷设计的经验(或在文字处理程序中格式化文本)。无法确定地知道，你指定的字体是否可用又或者当它触及你的用户浏览器时，类型会以何种形式出现。我们将进行最好地练习以应对我们之后会遇到的众多挑战。

在本章中，我们将会把第5章后面提到的Black Goose Bistro在线菜单摆放整齐。我鼓励大家做这些练习来亲身感受这些属性是如何工作的。图12-1显示了上次我们看到的和这次我们加工之后将看到的。这并算不上什么杰作，因为我们仅仅对CSS做了浅显的研究，但至少文本更需要改进。

字体属性

当我设计一个文本文档（用来打印的或者Web的）时，我首先要做的事情之一是指定字体样式。在CSS中，字体样式通过一个小的字体相关的属性系列，包括文字字体、大小、粗细和特殊样式。也有一些快捷属性可以让你出人意料地快速指定字体特性。

本章内容

字体相关的属性
Web字体和文字颜色
文本行设置如行高、缩进和对齐
文本处理如下划线、
首字母大写和阴影效果
字母和文字间距
继承关系(上下文的)、
ID和class选择器
优先级特性101

快速浏览

字体相关属性

font-family
font-size
font-weight
font-style
font-variant
font



图12-1：这一章对Black Goose Bistro菜单加工前后的对照视图

指定字体名称

为文本选择一个字体（在CSS中称作font family）是一个良好的开端。从简单的开始：使用字体（font family）属性和属性值。

font-family

- 属性值：一个或多个用逗号隔开的字体名称或通用字体库名称 | 可继承
- 默认值：依赖于浏览器
- 适用对象：所有元素
- 是否可继承：是

如同下例，使用font-family属性来指定一个字体名称或多个字体名称列表（也称作font stack）。

```
body { font-family: Arial; }
tt { font-family: Courier, monospace; }
p { font-family: "Duru Sans", Verdana, sans-serif; }
```

下面是一些重要的语法要求：

- 所有的字体名称，除了通用字体库，首字母必须大写。例如，使用“Arial”而不使用“arial”。
- 还要用逗号隔开多个字体名称，就像第二个和第三个例子所展示的那样。
- 特别注意，用空格隔开字母的字体名称（例如，第三个例子中的Duru Sans）必须放在引号内。

关于属性列表

书中每个新的属性列表都包括属性如何表现，以及如何使用属性的信息。这是对属性列表每个部分的快速解释。

属性值

它们是可以接受的属性值。预定义的关键值出现在代码的字体中（如small、italic或small-caps），并且拼写要一字不差。

默认值

没有指定属性值时，默认值将会默认调用。特别要注意的是，浏览器可能使用与CSS中设置的默认值不同的样式表的属性值。

适用对象

一些属性仅适用于特定的元素，比如block和table元素。

是否可继承

它表示属性能否传递给所选元素的子孙元素。继承的相关解释请参考第11章。

你或许会问：“为什么要指定多个字体呢？”这是个好问题，它使我们面临一个挑战，那就是为网页指定字体。

字体限制

浏览器只能显示它们可以进入的字体。那就意味着，字体已经安装在用户的硬件驱动上了。然而在2010年，浏览器支持的使用CSS font-face规则的嵌入式网络字体盛极一时，因此，设计师可以使用自己的字体。更多信息参见侧栏的“初识网络字体”。

但是回头看看font-family规则。如果浏览器识别不了（例如，如果用户的计算机中没有安装Futura字体或者提供的网络字体不能载入），那么，甚至当你指定字体为Futura时，浏览器的默认字体也将替代Futura字体。

幸运的是，当第一选择不可用时，CSS允许你提供一个备用字体的清单。如果第一个设定的字体识别不了，浏览器就尝试下一个，乃至整个清单，直至识别到有效的字体。在上面的第三个例子中，如果浏览器识别不了Duru Sans字体，它将会使用Verdana字体，如果Verdana字体也不可用，那么它会用sans-serif字体库的其他字体来替换。

初识网络字体

1998年开始，网页字体出现，但由于浏览器的不兼容，从未真正使用。幸运的是，现在一切发生了转机，网络字体已成为绝对可行的选择。Web形势一片大好。

有关网络字体的内容还有很多，这里仅仅是对富有挑战性的重点的介绍。

什么花费了你大量的时间？

网络字体有两个主要的障碍。第一个，不同的浏览器支持不同的字体格式。大多数字体源于开放类型（OTF）或者正确类型（TTF）格式，但IE仅认同嵌入式开放类型（EOT）。

好消息是，每一个所有浏览器厂商，甚至是IE，都在使用新的网页包装字体标准。这个新的格式，Web开放字体类型（WOFF）涵盖了网络包装字体文件。现在IE 9支持WOFF，将来的某一天会成为所有人的必需品。然而，我们仍需要在众多格式中选定相同的字体。

在网页上提供字体的另一个问题是，字体公司都担心他们的字体被窃取和下载。字体创造代价高昂，所以是非常有价值的。很多字体许可都限定用户和使用字体的机器数量，并且不包含“允许免费下载”的条款。

所以，为了链接网络字体，你需要用所有浏览器支持的并且是合法的方式来使用文字。有两种一般的方式：自创或者使用网络字体服务。下面分别看看这两种方式：

自创

在“自创”选项中，找出你想要的字体，将所有需要的格式置于服务器中，使用CSS3@font-face规则链接到你的网页。

第一步：找出字体。这可能会有一定的挑战性，因为所有商业字体的EULA并不包含网络使用。如果可以，还是去购买额外的网络许可。然而，由于需要，许多公司对于网络使用字体是开放的，并且有越来越多的开放字体库是免费的。Ethan

Dunham的Fontspring服务网站（Fontspring.com）提供网络许可的购买，有了网络许可你就可以在自己的电脑或网站上使用。Ethan Dunham的站点FontSquirrel（fontquirrel.com）是一个可免费用于商业目的的开放字体源。

第二步：以多种格式保存。在写本书时，多样化的格式已经成为一种现实。有许多工具可以将你的源字体转换为其他格式，有一种服务会用到你的字体并且给予你所需要的一切—Font Squirrel的"@font-face Generator"（www.fontquirrel.com/fontface/generator）。在该网页上传你的字体，它会以TTF、EOT、WOFF、SVG以及你所需要的CSS编码的形式返回该字体，以方便使用。记住，仅对网络使用允许的字体（不管是免费的开放的还是商业的）使用该服务。注意，你将获得比使用Generator质量更好的字体版本。

第三步：上传至服务器。开发者一般将他们的文件字体置于跟CSS文件一样的目录内，但这只是一种偏好。如果你从FontSquirrel下载文件包，确保将他们置于你能找到的地方。

第四步：写编码。用CSS文档中的@font-face规则，将字体链接到你的站点。你可以在样式表中引用规则赋予字体的font-family名称。同时规则也列明了不同格式字体文件的位置。这些十字形的浏览器编码例子，也是由Ethan Dunham开发的。我推荐阅读整篇文章www.fontspring.com/blog/the-new-bulletproof-font-face-syntax。也可以看看Paul Irish的升级版本paulirish.com/2009/bulletproof-font-face-implementation-syntax/。

```
@font-face {
  Font-family: 'Font name';
  Src: url('myfont-webfont.eot?#iefix')
    format('embedded-opentype');
```



```

Url('myfont-webfont.woff') format('woff'),
Url('myfont-webfont.ttf') format('truetype')
Url('myfont-webfont.svg#svgFontName')
  format('svg')
}

```

然后，你只需参考在你的字体规则中已设定的字体名称，像这样：

```
p{font-family:Font_name;}
```

使用字体嵌入式服务器

如果看上去很复杂，你只需与一个能解决所有困难的字体嵌入式服务器签约即可。付费就可以享用高质字体，服务器为公司掌管着字体的许可和保护。同时，服务器还可以提供使嵌入字体像复制粘贴那般容易的界面和工具。

这种服务器有很多收费形式，有的是根据月份付费，有的是根据字体付费，有的需要额外付费。还有从免费到每月几百美元不等。

这里有一些比较受欢迎的字体嵌入式服务器，可以去网上搜一下。

Google网络字体 (www.google.com/webfonts)

Google网络字体是免费的，它提供了上百种免费且常用的开放源字体。你要做的就是选择一种字体，然后复制粘贴它所形成的编码。如果你没有字体预算同时还对字体比较挑剔，就可以采用这种方式。在本章第一个练习中我们会使用它。

Adobe中的Typekit (www.typekit.com)

Typekit是首家网络字体服务器，现在是Adobe的一部分。他的服务器使用JavaScript以一种改良的方式将字体链接到你的站点。我也会推荐他们优秀作品的博客 (blog.typekit.com/category/type-rendering/)

Fonts.com

Fonts.com夸口说他们拥有最多的字体集。如果你需要个别特殊的字体，他们也许会有。

其他的服务器还包括WebINK (www.extensis.com/en/WebINK)、Typotheque (www.typotheque.com/webfonts) Font Live (www.fontslive.com) 和Fontdeck (fontdeck.com)。他们的费用结构和字体数是不一样的，你也可以到处看看。

总结

具体使用哪种方式给你的网站增加字体取决于你自己。如果你喜欢完全控制，那么就选用自创字体。如果你需要你的客户端依赖的某个有名的字体，你可以去那些收费的网络字体服务器中找。如果你只是找免费的试验网络字体，你可以去Google网络字体中找。

现在你应该已经对网络字体有了一个很好的基础性了解。接下来就是在很短的时间里进行改变，如果你已经做好了准备，那就开始自己搜索吧。

通用字体库

我们需要对最后一个选项“其他的sans-serif 字体”进行更多的讨论。

“sans-serif”字体只是指定font-family属性用的五个通用字体库之一。当你指定通用字体库时，浏览器将从这些格式上的类别中选择一个可用的。图12-2给出了各个通用字体库的示例。通用字体库的名称不需要把首字母大写。

serif

例如，*Times*、*Times New roman*、*Georgia*

Serif字体中某些字母最后的笔画带有装饰性的衬线或平板状的附加部分。

sans-serif

例如，*Arial*、*Arial Black*、*Verdana*、*Trebuchet MS*、*Helvetica*、

Geneva

sans-serif字体中字母笔画平直，在最后没有衬线。

monospace

例如, *Courier*、*Courier New*、*Andale Mono*

在monospace字体（也可以称作恒宽字体）中，一行中的所有的字母占用同样大小的空间。例如，大写的W不会比小写的i更宽。与monospace字体相比较，成比例的字体（如你现在阅读的文字字体）给不同的字母分配不同的宽度。

cursive

例如, *Apple Chancery*、*Zapf-Chancery*、*Comic Sans*

Cursive字体模仿手写外观。

fantasy

例如, *Impact*、*Western*或其他装饰性字体

Fantasy字体纯粹是装饰性的，适用于标题和其他显示类型。Fantasy字体由于缺乏跨平台的可用性和易读性，很少用于网页文本。

指定字体的策略

指定网页字体的最佳方案是从你的第一选项开始，然后提供类似的替代项，最后加上至少使用户处于正确的格式范畴的通用字体库。例如，如果你想使用一个竖直的sans-serif字体，你必须从你最欣赏的字体（Futura）入手，列一些较常用的字体（Univers、Tahoma、Geneva）清单，然后以通用的sans-serif字体结尾。对于列出的字体数量没有限制，但多数设计者会控制在十个以内。

字体库: Futura, Univers, Tahoma, Geneva, sans-serif;

好的字体集应该包括大多数电脑上内置的字体相关的字体。遵从那些源于Windows操作系统，MacOS操作系统和Linux操作系统以及流行软件包如Microsoft Office和Adobe Creative Suite的字体，有助于你从网络安全的字体清单中进行选择。下列网站提供的图表及统计数字，对于寻找那些通用字体是很有帮助的。

- Linux、苹果和window操作系统中的预置字体完全指导 (www.apaddedcell.com/sites/www.apaddedcell.com/files/fonts-article/final/index.html)
- 字体矩阵 (media.24ways.org/2007/17/fontmatrix.html)
- 编码字体的网站样式反馈和字体生成器 (www.codestyle.org/css/font-family/index.shtml)



图12-2: 五个通用字体库的例子

如果你对学习掌握字体有兴趣,我可以推荐下列文章,但必须确保用你自己的网站搜索去寻找更新的推荐。

- Vivien的“Striking Web Sites with Font Stacks that Inspire” (www.inspirationbit.com/striking-web-sites-with-font-stacks-that-inspire/)
- Nathan Ford的“Better CSS Font Stacks” (unitinteractive.com/blog/2008/06/26/better-css-font-stacks/)

因此,正如你看到的,给Web指定字体更像只是提供一些建议。你不能绝对控制用户将看到的字体。你的建议可能会获得首选,但也可能会不得不被其他选择替代。这也是你需要学会去适应的原因。

看起来,现在是开始格式化Black Goose Bistro菜单的好时候。每次我们学习一个新属性,都将给它添加一个新的样式规则。

有史以来最好的字体

有许多网络文章都在兜售所谓“有史以来最好的字体”,你使用的字体也往往会成为一种偏好。下面这些推荐都受到了Michael Tuck的“8个最终字体”(www.sitepoint.com/eight-definitive-font-stacks)的启发,同时也包含了Windows、Mac和Linux的后备字体。

窄对称 (Times-based)

Cambria, "Hoefler Text", "Nimbus Roman No9 L Regular", Times, "Times New Roman", Serif;

宽对称 (Georgia-based)

Constantia, "Lucida Bright", Lucidabright, "Lucida Serif", Lucida, "Dejavu Serif", "Liberation Serif", Georgia, serif;

窄的非对称 (Arial-based)

Univers, Calibri, "Liberation Sans", "Nimbus Sans L", Tahoma, Geneva, "Helvetica Neue", Helvetica, Arial, sans-serif;

宽的非对称 (Verdana-based)

"Lucida Grande", "Lucida Sans Unicode", "Lucida Sans", "Liberation Sans", Verdana, sans-serif;

单一间距

"Andale Mono Wt", "Andale Mono", "Lucida Console", "Liberation Mono", "Courier New", Courier, monospace;

练习12-1 格式化菜单

在这个练习中，我们将给Black Goose Bistro菜单的文档menu.html添加字体属性。这个菜单文档可以在www.learningwebdesign.com/4e/materials下载。你可以在文本编辑器中打开文档，也可以在浏览器中打开查看“以前”的状态，与图12-1近似。请关注这个文档，因为这个练习会一直添加更多的属性。

这次练习中我加入了一个嵌入式字体，从中可知有谷歌网络字体是多么的方便。

1. 这次练习我们将使用嵌入式样式表，首先在网页文档head部分添加一个style标签元素，如下所示：

```
<head>
  <title>Black Goose Bistro</title>
  <style>

  </style>
</head>
```

2. 我想将网页中所有文字都以Verdana字体或sans-serif字体库的一些其他字体显示。不需要给文档中的每一个元素定义规则，只需要给body元素定义规则，body元素包含的所有元素将会继承这个规则。把这条规则加入嵌入式样式表中：

```
<style>
  body {font-family: Verdana, sans-serif;}
</style>
```

3. 我想为Black Goose Bistro夏季菜单的标题使用一种奇幻的字体，所以我在Google网络字体 (www.google.com/webfonts) 上找到了叫做Marko One的免费字体。Google提供了将字体链接到我的HTML文件的编码，它应该在文档的开头，因而复制的时候需要保证复制准确。

```
<head>
  <title>Black Goose Bistro</title>
  <link href="http://fonts.googleapis.com/
css?family=Marko+One" rel="stylesheet">
</head>
```

4. 编写一个用于h1元素的规则。注意，我已经将Georgia或其他的serif字体设为后备字体。

```
<style>
  body {font-family:Verdana,sans-serif;}
  h1 {font-family:"Marko One",Georgia,
serif;}
</style>
```

5. 保存文档，并在浏览器中重新载入页面。效果如图12-3所示。注意，你需要网络和浏览器查看Marko One的标题行字体。在下一部分，我们将会操作字体尺寸。

图12-3：仅改变字体后的菜单

Black Goose Bistro • Summer Menu

Baker's Corner, Seekonk, Massachusetts

Hours: Monday through Thursday: 11 to 9, Friday and Saturday: 11 to midnight

Appetizers

This season, we explore the spicy flavors of the southwest in our appetizer collection.

Black bean purses

Spicy black bean and a blend of mexican cheeses wrapped in sheets of phyllo and baked until golden. \$3.95

Southwestern napoleons with lump crab — **new item!**

Layers of light lump crab meat, bean and corn salsa, and our handmade flour tortillas. \$7.95

Main courses

Big, bold flavors are the name of the game this summer. Allow us to assist you with finding the perfect wine.

Jerk rotisserie chicken with fried plantains — **new item!**

Tender chicken slow-roasted on the rotisserie, flavored with spicy and fragrant jerk sauce and served with fried plantains and fresh mango. **Very spicy.** \$12.95

Shrimp sate kebabs with peanut sauce

Skewers of shrimp marinated in lemongrass, garlic, and fish sauce then grilled to perfection. Served with spicy peanut sauce and jasmine rice. \$12.95

Grilled skirt steak with mushroom fricasee

Flavorful skirt steak marinated in asian flavors grilled as you like it*. Served over a blend of sauteed wild mushrooms with a side of blue cheese mashed potatoes. \$16.95

* We are required to warn you that undercooked food is a health risk.

指定字体尺寸

用适当命名的font-size属性来指定文本的字体尺寸。

font-size

属性值:	<i>length unit</i> <i>percentage</i> xx-small x-small small medium large x-large xx-large smaller larger inherit
默认值:	medium
适用对象:	所有元素
是否可继承:	是

可以用以下几种方法来指定文本:

- 指定尺寸用一个css的长度单位（见后面的“CSS计量单位”，这是一个完整的列表），如下所示：

```
h1 { font-size: 1.5em; }
```

当指定多个单位时，确保单位紧跟在数字后面，之间没有间隔。

错误 `h1 { font-size: 1.5 em; }` /*em单位前面有空格*/

- 作为百分比值，在元素默认或继承的基础上增大和减小尺寸。

```
h1 { font-size: 150%; }
```

- 使用绝对关键字（xx-small, x-small, small, medium, large, x-large, xx-large），在大多数当前的浏览器中，medium相当于默认字体尺寸。

```
h1 { font-size: x-large; }
```

- 使用相对关键字（larger或者smaller）来调整文字尺寸，使之比周围文字更大或更小。

```
strong { font-size: larger; }
```

下面继续一个关键的问题，尽管有这么多的选择，但当前网页设计中可接受的font-size属性值只有em为单位的计量值，百分比值（或者将两者合二为一）。下面我会解释其他的字体值，让我们以通常的方法开始我们的讨论。

em单位和百分比值都是一种相对计量，这意味着它们是基于另一种字体尺寸，即父元素的字体尺寸。

百分比值

在这个例子里，h1元素的父元素（即body元素）的font-size已被指定

为100%默认文档尺寸（一般是16像素）。h1元素从body元素继承了16像素，同时应用了150%的继承值，即24像素。例如，使用者用30像素的字体在视屏浏览器上阅读，那么将导致h1元素变为45像素，但与主体文本相关的部分仍为原样。

```
body { font-size: 100%; }  
h1 { font-size: 150%; } /* 16的150% 等于24 */
```

CSS计量单位

CSS3提供了多种计量单位。他们分为两大类：绝对单位和相对单位。

相对单位

相对单位基于其他事物的尺寸，例如，文本默认尺寸，或者父级元素的尺寸。

px 像素，在CSS2.1中也称作相关像素，因为它随着显示器分辨率的不同而变化

em 与当前字体尺寸相等的计量单位

ex 大约等于字体中小写字母x的高度

以下单位是CSS3中新加入的。浏览器支持需要花费一段时间。

rem 根em，等于根元素（html）的em尺寸

ch 零宽度，等于目前0的字体尺寸宽度

vw 单位宽度，等于当前浏览器窗口宽度的百分之一

vh 单位高度，等于目前浏览器窗口高度的百分之一

vm 单位最小值等于vw和vm中较小的值

绝对单位

绝对单位有预定义的意思或者在现实世界中有等价物。

px 像素定义为等价于CSS3中一英寸的96分之一的绝对量称

pt point (在CSS2.1中1 point = 1/72 英寸)

pc picas (1 pica = 12 points)

mm 毫米

cm 厘米

in 英寸

由于绝对单位与计算机显示屏不存在对应关系，它们应该避免在网页样式表中使用。如果样式表用于打印文档，那么它们正合适。

你有没有注意到，像素（px）在相对单位和绝对单位中都出现了？因为W3C还没有完全确定下来。另一种定义是，在实践中扮演灵活性弱于相对单位的绝对计量的角色。

em计量

em是一个相对的计量单位，在传统的印刷中，是基于大写字母M的宽度（因此，命名为“em”）。在CSS规范中，em是两行之间没有多余的空间时基线之间的距离（也叫做行距）。在基字体尺寸为16的文本中，一个em等于16像素；在字体尺寸为12的文本中，一个em等于12像素，依此类推，如图12-4所示。

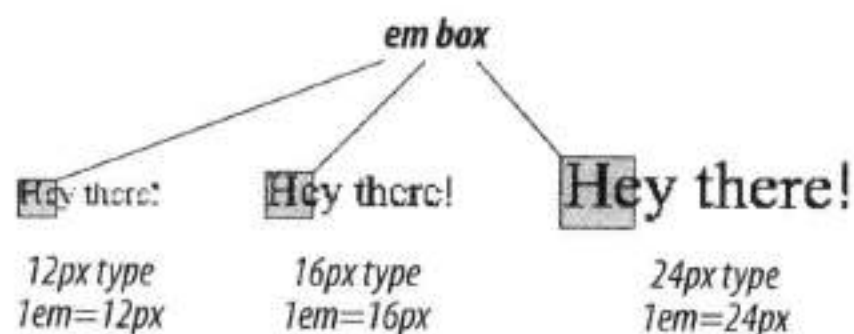


图12-4：一个基于文本字体尺寸的em

当一个文本元素的em的尺寸可以由浏览器来计算时，em单位就可以用于所有种类的计量，如网页元素的缩进，页边距和宽度等。

在文本尺寸中，em的值像一个比例因子，类似于百分比。正如在前面的例子，如果基字体大小为16像素，设定h1的元素字体尺寸为1.5em即24像素高。

```
body { font-size: 100%; }
h1 { font-size: 1.5em; } /* 1.5 x 16 = 24 */
```

em最佳范例

写作本书时，使得em显示一致的最佳方案是将body元素的尺寸设为100%，然后像我们在上个例子中所做的那样，用em来设定之后的文本元素的尺寸。这保留了使用者的首选文本尺寸，也确保了文本元素的比例。

以下列举了使用em的一些问题。一个是由于错误，当在文本中设定尺寸时，浏览器和平台之间存在一些不协调。

另一个是，基于元素的继承尺寸，意味着，它们的尺寸基于应用的上下

注意： 不要把em单位与HTML中用来强调文本的em元素相混淆。它们是完全不同的两回事。

为了计算%和em值，使用这个公式：
目标尺寸÷内容尺寸=结果

文。如果有许多嵌入式元素，那么它们尺寸的大或小与每个嵌入等级有关。举个例子，会使你对此更了解。

文档的body大小为100%（16像素），但你想要的是一篇14像素的文章。用14除以16，你可以得到0.875em。现在如果将article中的h2元素设为18像素，那em尺寸就不是基于16像素的主体文本尺寸，而是基于h2的article元素的18像素尺寸。我们用18除以14，得到最后的计量值为em1.28571429，这是一个无限循环小数，但这里不需要算得这么精细。

```
body {font-size:100%}  
article {font-size:.875em;}  
    /*based on inherited size of the body text*/  
article h2 {font-size:1.28571429em;}  
    /*based on the article font size,not body*/
```

Ethan Marcotte 多年前就提出公式“目标÷文本=结果”，该公式可用于排版页面布局以及其他尺寸任务。在这本书中将再次提及该公式。

所以，请密切关注该公式。同时注意编写样式规则的方式。关注侧栏的“Root EM的介绍”，以便为解决问题寻找方法。

像素和绝对测量值

虽然许多开发者为精确控制而倾向于选择像素字体测量值，但一般认为这样做过于死板，使用相对测量似乎更合适。只要控制px，绝对单位比如pt、pc、in、mm、cm就会失效，因为它们彼此之间是不相关的（尽管它们对于印刷样式表可能是有用的）。

另一个像素font-size值的缺点是，IE不认可用像素来调整文本尺寸。这意味着即使用户无法阅读，也不能调整文本为10或11像素。这在可使用性方面是个大问题。IE 7或更高级版本允许整个页面的调整，这是一个改进，但并不是一个理想的用户体验。

Root EM 的介绍

在CSS3中有个基于根（html）元素字体尺寸的rem相关测量。如果你指定html元素的尺寸，那么rem测量中指定的所有元素都会与这个尺寸相关，而不是他们的继承尺寸。这就避免了em可能的恶化。Rem的缺点是，IE 8或是更早的浏览器不支持它，所以，你需要以像素尺寸来提供后备字体。而支持rem的浏览器则使用最新的声明。

```
html{
  font-size: 100%;
}
#main{
  font-size: 12px;
  font-size: .75rem;
}
```

rem单位在Web开发社区很流行。如果想要了解更多，我推荐Jonathan Snook的文章“Font Sizing with rem” (snook.ca/archives/html_and_css/font-size-with-rem)。

使用特征关键字

其他指定字体尺寸的方式是预先定义的绝对关键字的一种：xx-small、x-small、small、medium、large、x-large和xx-large。这些关键字并没有对应的精确的计量数值，但是相互之间保持着固定的比例。默认值是当前浏览器的medium尺寸值。图12-5呈现了在默认值为16像素的浏览器中各个关键字产生的效果。其中包括Verdana字体和Times字体，可以看出，即使基于相同的尺寸，small与更小的尺寸在可读性方面仍存在着很大的差异。

特征关键字的优势在于当前标准模式的浏览器不会让关键字指定尺寸小于9像素，以免难以辨认。缺点在于，尺寸关键字是不精确和难以预料的。例如，虽然大多数浏览器的每个尺寸级别增长120%，但是某些浏览器使用的比例因子为150%。

图12-5：绝对关键字指定的文本尺寸

相对关键字，larger和smaller，用于对照父元素文本的字体尺寸来调整字体尺寸。尺寸变化的确切数值，取决于每一个浏览器，而不是你能控制的。尽管有这种限制，如果精确比例不是很严格，那么调大一点或调小一点还是很容易的。

This is an example of the default text size in Verdana.

xx-small | x-small | small | medium | large | x-large | **xx-large**

This is an example of the default text size in Times.

xx-small | x-small | small | medium | large | x-large | **xx-large**

练习12-2 设定字体尺寸

改进文本元素的尺寸设定，使在线菜单更加精致。在文本编辑器中打开 *menu.html*，按照下列步骤操作。你可以在任何时刻保存文档，在浏览器中欣赏你的工作成果。在此过程中，你可以自由地尝试其他尺寸值。

- 1. 在网页中有很多方法设定文本大小。在这个例子中，我认为最好的方法就是，将body元素设为100%，为之后的em计量做好准备。

```
body {
  font-family:Verdana, sans-serif;
  font-size:100%
}
```

- 2. 我更喜欢正文的文本元素显示为14像素而不是普通默认的16像素（如果用户觉得太小，可以在浏览器中调大）。通过添加一个新的群组选择器，并使用目标公式 $(14) / \text{context}(16) = 0.875$ ，将p元素和d1元素的大小设为0.875em。那么我可以只用87.5%来达到同样的效果。

```
p,d1{
  font-size:.875em;
}
```

- 3. 现在开始操作标题尺寸。设定标题（h1）元素为body元素文本尺寸的1倍半。h2s可用默认的文本尺寸（1em）。

```
h1 {
  font-family:"Marko One",
  Georgia, serif;
```

```
font-size:1.5em
}
h2 {
  font-size:1em;
}
```

图12-6显示了font-size属性改动之后的效果。

Black Goose Bistro • Summer Menu

Baker's Corner, Seekonk, Massachusetts
Hours: Monday through Thursday: 11 to 9, Friday and Saturday: 11 to midnight

Appetizers

This season, we explore the spicy flavors of the southwest in our appetizer collection.

Black bean purses
Spicy black t
Southwestern nap
Layers of ligl

Black Goose Bistro • Summer Menu

Baker's Corner, Seekonk, Massachusetts
Hours: Monday through Thursday: 11 to 9, Friday and Saturday: 11 to midnight

Main courses

Big, bold flavors

This season, we explore the spicy flavors of the southwest in our appetizer collection.

Jerk rotisserie chi
Tender chick
plantains an
Shrimp sate keba
Skewers of s
sauce and ja
Grilled skirt steak
Flavorful sk
with a side o

Main courses

Big, bold flavors are the name of the game this summer. Allow us to assist you with finding the perfect wine.

Jerk rotisserie chicken with fried plantains — **new item!**
Tender chicken slow-roasted on the rotisserie, flavored with spicy and fragrant jerk sauce and served with fried plantains and fresh mango. **Very spicy.** \$12.95
Shrimp sate kebabs with peanut sauce
Skewers of shrimp marinated in lemongrass, garlic, and fish sauce then grilled to perfection. Served with spicy peanut sauce and jasmine rice. \$12.95
Grilled skirt steak with mushroom fricasee
Flavorful skirt steak marinated in asian flavors grilled as you like it*. Served over a blend of sauteed wild mushrooms with a side of blue cheese mashed potatoes. \$16.95

* We are required to warn you that undercooked food is a health risk.

图12-6: font-size属性微小改动之后的在线菜单

字重（粗细）

学过字体库和尺寸之后，剩下的字体属性都是直接明了的。例如，如果你想要一个元素以粗体显示，可以使用font-weight属性调整字体的粗细。

font-weight

属性值: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit

默认值: normal

适用对象: 所有元素

是否可继承: 是

如你所见，font-weight属性有许多预定值，包括描述性用语（normal、bold、bolder和lighter）和9个可用字重的数值（100~900）。因为Web中使用的普通字体大多只有两个字重：正常（normal（或roman））和粗体（bold），在大多数情况下你只会用到（bold）字重。除非文本是粗体的（如强调句或标题），通常使用normal属性值使文本以普通字重显示。

数字表示法是个有趣的想法，但是因为在字重的范围内没有很多的字体，而且浏览器支持的质量参差不齐，所以数字表示法很少使用。通常不管是什么浏览器，字重600及以上都表示粗体。如图12-7所示。

关于继承

你会看到在CSS属性的关键值列表中包含继承属性inherit。inherit值可以显式地使一个元素继承它的父元素的样式属性值。在需要覆盖该元素的其他样式，并使该元素与它的父元素总是保持一致时，继承性是非常有用的。

normal | **bold** | **bolder** | lighter
100 | 200 | 300 | 400 | 500
600 | **700** | **800** | **900**

在Safari中渲染

normal | **bold** | **bolder** | lighter
100 | 200 | 300 | 400 | 500
600 | **700** | **800** | **900**

在Firefox (Mac)中渲染

图12-7: font-weight属性值的效果

Black Goose Bistro • Summer Menu

Baker's Corner, Seekonk, Massachusetts
Hours: Monday through Thursday: 11 to 9, Friday and Saturday: 11 to midnight

Appetizers

This season, we explore the spicy flavors of the southwest in our appetizer collection.

Black bean purées

Spicy black bean and a blend of mexican cheeses wrapped in sheets of phyllo and baked until golden. \$3.95

Southwestern napoleons with lump crab — new item!

Layers of light lump crab meat, bean and corn salsa, and our handmade flour tortillas. \$7.95

Main courses

Big, bold flavors are the name of the game this summer. Allow us to assist you with finding the perfect wine.

Jerk rotisserie chicken with fried plantains — new item!

Tender chicken slow-roasted on the rotisserie, flavored with spicy and fragrant jerk sauce and served with fried plantains and fresh mango. **Very spicy.** \$12.95

Shrimp saté kebabs with peanut sauce

Skewers of shrimp marinated in lemongrass, garlic, and fish sauce then grilled to perfection. Served with spicy peanut sauce and jasmine rice. \$12.95

Grilled skirt steak with mushroom fricasee

Flavorful skirt steak marinated in asian flavors grilled as you like it*. Served over a blend of sauteed wild mushrooms with a side of blue cheese mashed potatoes. \$16.95

* We are required to warn you that undercooked food is a health risk.

图12-8: 在菜单中对dt元素使用font-weight属性

练习12-3 文本粗体化

回到菜单。我决定把所有的菜单项名称改为粗体。我不想把每一个菜单项都用标签包起来——这是1996年的做法！我也不想用strong元素去标记它们——这是不准确的语义。正确的做法应该是，为合乎语义的dt（定义项）元素应用一种样式，使这些文本全为粗体。并把这个规则加入样式表中，保存文件，在浏览器中查看（如图12-8所示）。

```
dt { font-weight:bold; }
```


字体样式（斜体）

Font-style属性影响着文本的形态，也就是说，字母形状会是垂直的（normal）还是倾斜（italic和oblique）的。

Font-style

- 属性值：normal|italic|oblique|inherit
- 默认值：normal
- 适用对象：所有元素
- 是否可继承：是

Italic和oblique都是斜体字。不同之处在于italic样式设计为字母弯曲，而oblique样式只是把普通字体倾斜。事实上，在大多数浏览器上，它们看起来完全一样（图12-9）。你也许只会用font-style属性将文本设置为italic或者将已经斜体化（如强调句）的文本显示为normal。

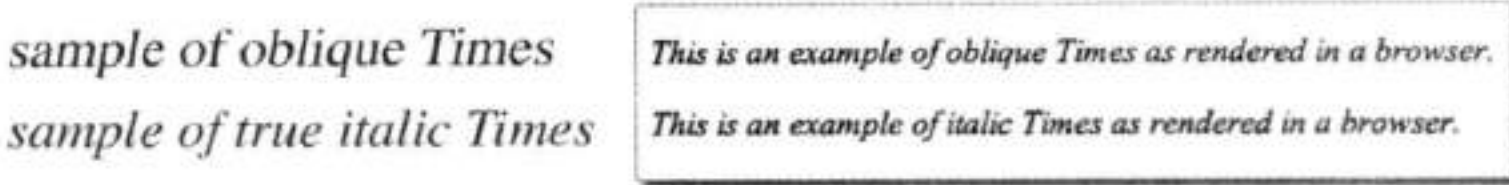


图12-9: italic和oblique文本

练习12-4 倾斜文本

因为菜单项名称是粗体的，一些标记为strong的文本并不很突出，所以我将它们设定为italic样式来进一步强调。要做到这一点，只需要简单地把font-style属性应用到strong元素即可。

```
strong { font-style: italic; }
```

再保存并重新载入。详细的外观样式如图12-10所示。



图12-10: 将font-style属性应用到strong元素

字体变量（小型的大写字母）

一些字样用“小型的大写字母”字样变体。这是一个独特的字体设计,用小型的大写字母替换小写字母。`font-variant`属性使得设计者能够为文本元素指定诸如小型大写字母的字体。

font-variant

属性值:	<code>normal small-caps inherit</code>
默认值:	<code>normal</code>
适用对象:	所有元素
是否可继承:	是

大多数情况下,真正的`small-caps`字体是不可用的,所以浏览器通过将字体的大写字母缩小来模拟小型大写字母。对于字体的执著爱好者,这显然不够理想,并且导致不一致的笔画粗细,但是给一小部分文字添加字体变量是可以接受的。我们将在练习12-6中使用`font-variant`属性。

字体快捷属性

为每个文本元素指定多个字体属性将显得重复而且冗长,所以CSS的创造者提供了快捷字体属性,可以把所有字体相关的属性编译到一个规则中去。

font

属性值:	<code>font-style font-weight font-variant font-size/line-height font-family inherit</code>
默认值:	依赖列表中每个属性的默认值
适用对象:	所有元素
是否可继承:	是

刚看到的`font`属性值是用空格字符隔开的、所有字体相关属性值的列表。在这个属性中,属性值的顺序很重要:

```
{ font: style weight variant size/line-height font-family }
```

按顺序,`font`属性至少应包括`font-size`属性值和`font-family`属性值。遗漏一个或者放错顺序将导致整个规则失效。这是包含最少属性值的例子:

```
p { font: 1em sans-serif; }
```


一旦满足尺寸和字体库的要求，其他的属性值就是可选的，而且可以出现在font-size属性值前面的任何位置。当样式、字重或者字体变量遗漏，这些属性将会被转化为normal状态。还有一个属性值，line-height，我们之前没有见过。它调整了文本线的高度，增加了文本线间的间距。它的位置仅次于font-size属性值，之间用斜线隔开，如下面的例子所示：

```
h3 { font: oblique bold small-caps 1.5em/1.8em Verdana, sans-serif; }
h2 { font: bold 1.75em/2 sans-serif; }
```

我们使用快捷font属性来对h2标题作一些改变。

练习12-5 使用快捷font属性

上次对菜单作了调整，然后短暂休息。为了节约空间，我们可以将为h1元素指定的所有属性与快捷字体属性相融合。

```
h1 {
  Font:bold 1.5em "Marko One", Georgia, serif;
}
```

你可能觉得在这个规则里包含字重值为bold属性是多余的。毕竟，h1元素已经是粗体了，对吗？不要忘记，在快捷字体属性中，如果你遗漏了一个值，快捷字体属性将重置为这个属性的默认值，并不仅是浏览器的默认值。

在这种情况下，用font声明的font-weight的默认值是normal。因为如果不在font属性中明确使用bold，那么我们写的样式表将覆盖浏览器中关于标题的属性设置，h1标题将显示为字重为normal的文本。快捷字体属性可以取巧，就像这样，但是注意不要有遗漏，也不要覆盖你依靠的默认值和继承值。

你可以保存成果，再在浏览器中查看，如果你操作正确，它应该跟前面步骤的结果是一样的。

改变文本颜色

在第11章已经看到如何去改变文本的颜色，老实说，这里没有什么可以多说的。你可以用color属性改变文本颜色。

color

属性值：	颜色值（名称或数字表示） inherit
默认值：	依赖于浏览器与用户习惯
适用对象：	所有元素
是否可继承：	是

使用color属性非常简单。color属性值可以是预定义的颜色名称（参见侧栏的“颜色名称”），也可以是描述具体的RGB三原色的数值。下面的例子，都可以让文档的h1元素显示为灰色。

```
h1 { color: gray; }
h1 { color: #666666; }
h1 { color: #666; }
h1 { color:rgb(102, 102, 102);}
```

现在不用担心这个数值，只是让大家看下。RGB颜色将在第13章中详细讨论，所以在这里，我只用几个颜色名称作演示。

color属性是可继承的，所以可以通过设定body元素的color属性值来改变文档中所有文本的颜色，示例如下：

```
body { color: fuchsia; }
```

好了，你可能不想所有的文字都是紫红色，但是你应该知道该怎么办。

准确地说，color属性并不是一个严格意义上的文本相关的属性。事实上，根据CSS规范，它是用来改变元素的前景（相对于背景）颜色的。元素的前景由它包含的文本和边框组成。

当给一个元素（包括图像元素）设定color属性时，这个颜色也会应用到边框，除非已经指定了border-color属性来覆盖它。在第14章中会深入讨论边框和边框颜色问题。

在给在线菜单添加颜色之前，我想讲点别的，介绍更多类型的选择器，给文档中的目标元素样式添加灵活性。

更多选择器类型

目前，我们一直在用元素名称作为选择器。在第11章，你看到选择器如何组成一个逗号分隔的列表，而把这些属性一次应用到多个元素。下面是你所知道的选择器的例子。

```
元素选择器    p { color: navy; }
组合选择器    p, ul, p, td, th { color: navy; }
```

显而易见，这样选择的缺点是，属性（例子中，海军蓝的文本）将会应用到文档中每一个段和其他列表中的元素。有时，我们只想把一个规则应用到特定的段或段落。在本节中，我们会讲述三个选择器类型：后代选择器，ID选择器和类选择器。

快速浏览

颜色名称

CSS2.1 中定义的17个标准颜色名称：

black	white	purple
lime	navy	aqua
silver	maroon	fuchsia
olive	blue	orange
gray	red	green
yellow	teal	

更新的CSS3颜色模块允许在样式表指定140个颜色名称。单个样本可参见learningwebdesign.com/colornames.html。

元素名称间的空格表示第二个元素必须包含在第一个元素中。

后代选择器

后代选择器的目标是包含在其他元素中的元素（所以称为后代）。这是关联选择器的一个例子，因为元素的选择基于它的上下文或与其他元素的关系。侧栏“其他关联选择器”列出了更多。

后代选择器用空格隔开的列表表示。这个例子的目标是强调文本（即em元素），并且只是列表项（即li元素）中出现的强调文本。段落或其他元素中的强调文本不会受到影响（图12-11）。

```
li em { color: olive; }
```

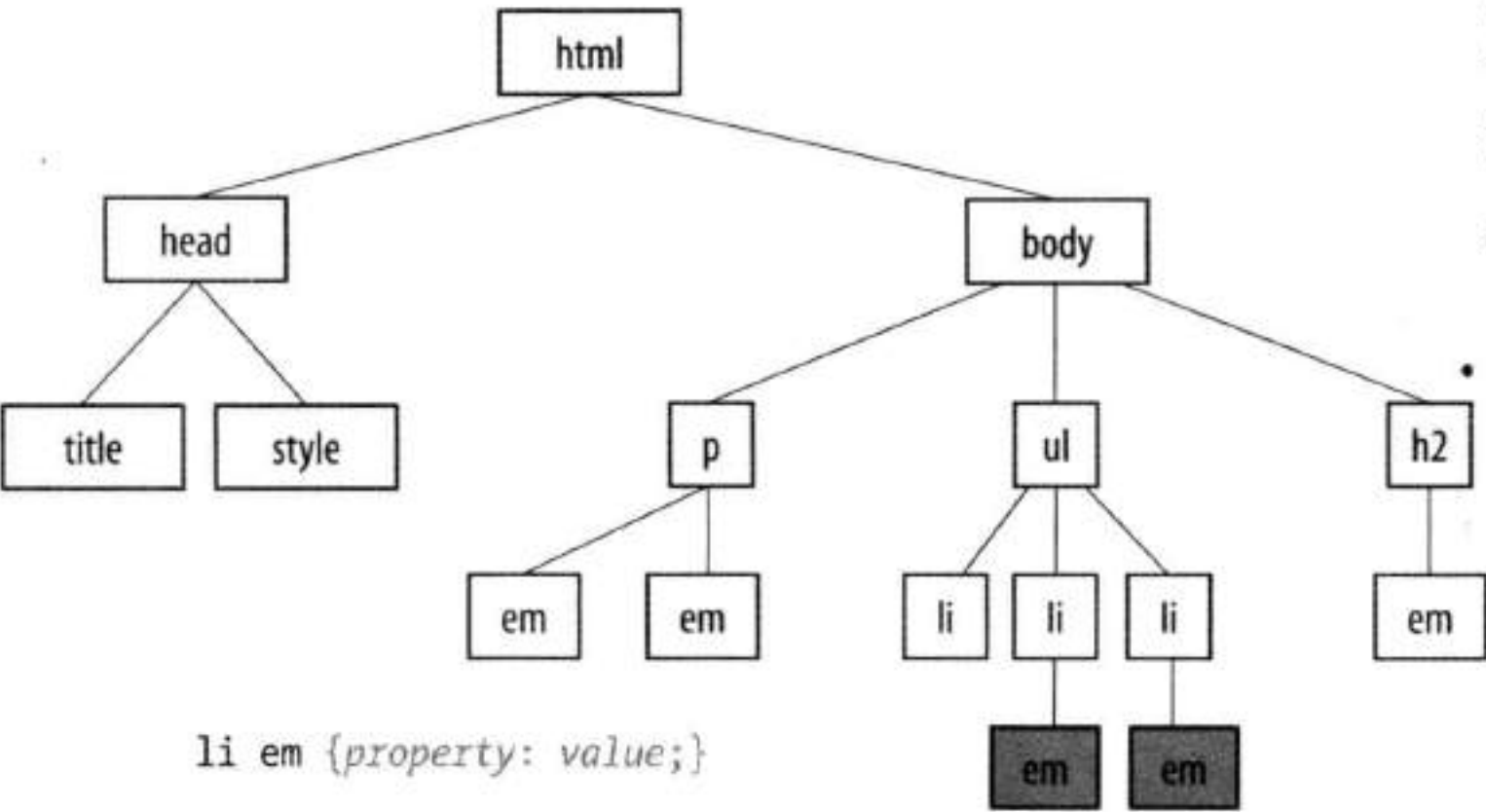
就像我们前面看到的，下面例子展示了关联选择器如何组合到逗号隔开的列表中。这个规则的目标是em元素，而且仅仅出现在h1、h2和h3标题元素中。

```
h1 em, h2 em, h3 em { color: red; }
```

后代选择器也可以嵌套多层。下面例子的目标是有序列表（即ol元素）中的锚（即a元素）中的em元素。

```
ol a em { font-variant: small-caps; }
```

图12-11：只有li元素中的em元素被选中，其他em元素不受影响



其他关联选择器

后代选择器是四种关联选择器之一（称为CSS3规范的集合），其他三个分别是子选择器、相邻同胞选择器和一般同胞选择器。

子选择器

子选择器类似于后代选择器，但它的目标只能是特定元素的直接子代。之间没有其他级别。子选择器用大于符号（>）表示。当且仅当强调文本被p元素直接包含，这条规则才对它有影响。段落中链接（a）中的em元素不受影响。

```
p > em {font-weight:bold }
```

相邻同胞选择器

相邻同胞选择器用于直接来自同一父元素的元素上。它使用加号（+）来表示。对于跟在h1之后的段落有特殊的处理方式。而其他段落则不受影响。

```
h1 + p {font-style: italic;}
```

一般同胞选择器

一般同胞选择器选择了这样一类元素，它们与指定元素源于同一父辈，并且在它之后发生。他们彼此之间不需要直接连接。这种类型的选择器是CSS3中的新内容，它们不能被IE 8和更早的系统所支持。下列规则选择的是与h1来自同一父辈元素（如section和article）以及出现在h1之后的h2。

```
h1 ~ h2 {font-weight:normal; }
```

ID选择器

回顾第5章，我们学习了给元素指定唯一标识名称（即id引用）的id属性。这个属性可以用于所有元素，并且在定义普通的div元素和span元素中广泛使用。

#字符标志着ID选择器。

ID选择器允许通过id值来选定目标。ID选择器的标志是#字符，这个也叫做hash标志。

下面例子是一个带有id引用的表单项。

```
<li id="catalog1234">Happy Face T-shirt</li>
```

现在可以使用ID选择器给列表项写一条样式规则，就像这样（注意id引用前的#字符）：

```
li#catalog1234 { color: red; }
```


提示

id属性值必须以字母 (A-Z或a-z) 开头。除了字母, 名称中还可以包含数字 (0-9)、连字号 (-)、下划线 (_)、冒号 (:) 和句点 (.)。注意, 冒号和句点是不能使用的, 因为它们被用作选择器时, 会与CSS语法混淆。

句点 (.) 符号表示类选择器。

因为文档中id值是唯一的, 所以忽略元素名也是可以接受的。这条规则与上一条等效:

```
#catalog1234 { color: red; }
```

也可以使用ID选择器作为关联选择器的一部分。这个例子中, 样式只能被应用到在标识为“links”的元素中出现的li元素中。这样, 不借助其他的标记, 就可以区别对待links中的列表项和网页中其他的列表项。

```
#link li { margin-left: 10px; }
```

你应该开始见识到选择器的厉害了, 你可以看到它们是如何配合着精心策划的语义标记来使用的。

类选择器

最后一个选择器类型, 回顾文本样式属性。在第5章学习的另一个元素标识是class标识, 它把元素分到相关组合里去。不同于id属性, 多个元素可以共用一个class名称。不仅如此, 一个元素还可以属于多个类。

选择属于同一类的元素, 你应该可以猜到了, 用类选择器。在选择器的开头, 类名用句点 (.) 表示。例如, 要选择所有class="special"的段落, 可以用这个选择器 (句点表示下一个单词是类选择器):

```
p.special { color: orange; }
```

在应用属性到同一个类的所有元素时, 忽略选择器中的元素名 (确保有句点, 它是类选择器的标志)。下例将选中标记了class="special"的所有段落和其他元素。

```
.special { color: orange; }
```

通用选择器

CSS2引入了通用元素选择器 (*), 它匹配所有的元素 (就像编程语言里的通配符)。样式规则:

```
* {color: gray; }
```

使文档中的每个元素都为灰色。在关联选择器中同样适用, 如下例, 选择intro部分的所有元素

```
#intro * { color: gray; }
```

通用选择器会导致某些浏览器中表单控件的问题。如果网页中包含输入表单, 最安全的方法是避免使用通用选择器。

优先级特性101

在第11章介绍了术语优先级特性，选择器优先级特性的值越大，在遇到样式规则冲突时，其权重越高。既然已经对选择器又多了些认识，正好再来了解下这个非常重要的概念。

实际上，CSS计算优先级特性很复杂，但在大多情况下，这份优先级特性从大到小排列的列表已经够用了。

- ID选择器高于（并覆盖）类选择器
- 类选择器又高于（并覆盖）关联选择器
- 关联选择器又高于（并覆盖）独立选择器
- 独立元素选择器

所以，例如，如果样式表中strong元素有两个冲突的规则：

```
strong { color: red; }
h1 strong { color: blue; }
```

关联选择器（h1 strong）比元素选择器有更多的优先级特性和权重。

在战略上使用优先级特性可以使样式表更简单，标记更少。例如，可以给一个元素设定样式（如p元素），然后在需要的时候通过更高优先级特性的选择器来覆盖。

```
p { line-height: 1.2em; }
blockquote p { line-height: 1em; }
p.intro { line-height: 2em; }
```

在这些例子中，blockquote中的p元素的行高低于正常段落。然而，因为类选择器优先级特性高于关联选择器，所以即使是blockquote中的段落，所有属于intro类的段落行高都为2em。

要精通CSS，理解继承和优先级特性的概念是至关重要的，但是还远远不够，学习一些关于优先级特性的知识也是必要的。参考文章见侧栏“更多关于优先级特性”。

现在，回到菜单。幸运的是，Black Goose Bistro已经在语义上完全标记了，所以在选择具体元素时有很多选项。在练习12-6中可以试试这些新选择器。

更多关于优先级特性

这一章概述的优先级特性足以带你入门，但是当你获得更多经验并且你的样式表变得更加复杂的时候，你会发现你需要对内部动作进行更透彻的理解。

优先级特性严密计算的科学解释参照文章 CSS介绍（CSS Recommendation），网址 www.w3.org/TR/CSS21/cascade.html#specificity。

Eric Meyer 在他的书Cascading Style Sheets, The Definitive Guide (O'Reilly Media) 中，提供了对这个系统彻底的、更易理解和接受的描述。

如果想在线寻求帮助，我推荐 Vitaly Friedman 的 Smashing Magazine 文章，CSS 优先级：你必须知道的事情（coding.smashingmagazine.com/2007/07/27/css-specificity-things-you-should-know/）。

如果想进一步了解星球大战，可以选择 Andy Clarke 的 CSS：优先级战争（CSS: Specificity Wars）（www.stuffandnonsense.co.uk/archives/css_specificity_wars.html）。

练习12-6 使用选择器

这次，我们将添加更多的样式规则，使用到目前学过的后代、ID和类选择器联合字体属性和颜色属性。

1. 我想给靠近某个菜单名称的“new item!”元素添加颜色。它们标记为strong，所以可以给strong元素应用颜色属性。内嵌样式表中加入这条规则，保存文件，在浏览器中重新载入。

```
strong { font-style:italic ;color: maroon; }
```

我们可以看到这条规则发挥作用了，但现在描述中的strong元素“Very spicy”也变为栗色，这并不是我想要的。解决方案是使用关联选择器只选中在dt元素中出现的strong元素。删除color声明，同时创建一个只选中strong元素的新规则。

```
dt strong { color: maroon; }
```

2. 从文档源代码中你可以发现内容被分为三个唯一的div: info、appetizers和entrees。定义样式时我们可以利用这些好处。现在，简单地把header部分的文字设定为teal。由于颜色属性的可继承性，属性只需要应用到div元素，它们将传递到h1元素和p元素。

```
#info { color: teal; }
```

3. 下面我们来做一些更奇异的事，在不影响网页中其他段落的同时，将header元素里的段落设置为斜体。关联选择器再次成为解决问题的方案。这条规则只选中文档中包含在info部分的段落。

```
#info p { font-style: italic; }
```

4. 我想专门处理菜单上所有的价格。幸运的是，它们都已经用span元素标记，如下：

```
<span class="price">$3.95
</span>
```

所以现在我们唯一要做的就是用类选择器写一条规则，把字体改为Georgia字体或者serif库中的某个字体，然后设定为斜体、灰色。

```
.price {
  font-family: Georgia, serif;
  font-style: italic;
  color:gray;
}
```

5. 类似地，改变header部分的已经标记为属于“label”类的文本的外观，使它们更突出。

```
.label {
  font-weight: bold;
  font-variant: small-caps;
  font-style: normal;
}
```

6. 最后，页面底部有一个警告，我想将它缩小，变成红色。它已经被定义为“warning”类，所以我们可以以此选中这个段落来设定样式。操作的过程中，我将相同的样式应用于网页前面的sup元素（脚注的星号），这样它们就匹配了。注意，由于使用了组合选择器，所以不需要单独编写规则。

```
p.warning, sup {
  font-size: x-small;
  color: red;
}
```

图12-12显示了这些改变的效果。

BLACK GOOSE BISTRO • SUMMER MENU

Baker's Corner, Seekonk, Massachusetts

HOURS: MONDAY THROUGH THURSDAY: 11 to 9, FRIDAY AND SATURDAY: 11 to midnight

Appetizers

This season, we explore the spicy flavors of the southwest in our appetizer collection.

Black bean purses

Spicy black bean and a blend of mexican cheeses wrapped in sheets of phyllo and baked until golden. \$3.95

Southwestern napoleons with lump crab — new item!

Layers of light lump crab meat, bean and corn salsa, and our handmade flour tortillas. \$7.95

Main courses

Big, bold flavors are the name of the game this summer. Allow us to assist you with finding the perfect wine.

Jerk rotisserie chicken with fried plantains — new item!

Tender chicken slow-roasted on the rotisserie, flavored with spicy and fragrant jerk sauce and served with fried plantains and fresh mango. **Very spicy.** \$12.95

Shrimp sate kebabs with peanut sauce

Skewers of shrimp marinated in lemongrass, garlic, and fish sauce then grilled to perfection. Served with spicy peanut sauce and jasmine rice. \$12.95

Grilled skirt steak with mushroom fricasee

Flavorful skirt steak marinated in asian flavors grilled as you like it*. Served over a blend of sauteed wild mushrooms with a side of blue cheese mashed potatoes. \$16.95

* We are required to warn you that undercooked food is a health risk.

图12-12: Black Goose Bistro在线菜单的当前状态

字行设置

下一批文本属性与整行的文本有关，而与字符的形状无关。类似于打印，它们允许网页的作者格式化文本，使用缩进，额外的行距（行之间的空间）和不同的水平对齐方式。

行高

Line-height属性定义了文本中基线之间的最小距离。传统定义为快捷字体属性的一部分。基线是虚拟的线，位于字符的底部。CSS中的行高类似于传统印刷中的行距。虽然行高是从基线到基线计算来的，但大部分浏览器把文本上下的多余空间分开，从而把它置于整行高度的中间。（图12-14）。

Line-height之所以称作指定“最小”距离，是因为如果在行中放一个高的图片，那么文本行的高度会扩大来容纳图片。

line-height

属性值：	数值 长度 百分比 normal inherit
默认值：	normal
适用对象：	所有元素
是否可继承：	是

例子展示了三种不同的方法，用来将行高设置为字体尺寸的两倍。

```
p { line-height: 2; }  
p { line-height: 2em; }  
p { line-height: 200%; }
```

当指定单独一个数字时，就像第一个例子那样，数字作为被当前字体尺寸乘的比例因子，用来计算line-height的值。行高也可以用CSS长度单位指定，但是最好使用相对单位em。em单位和百分比值都是基于当前字体的尺寸。在这三个例子中，如果当前文本字体尺寸是16像素，那么计算之后行高应该为32像素（如图12-13所示）。

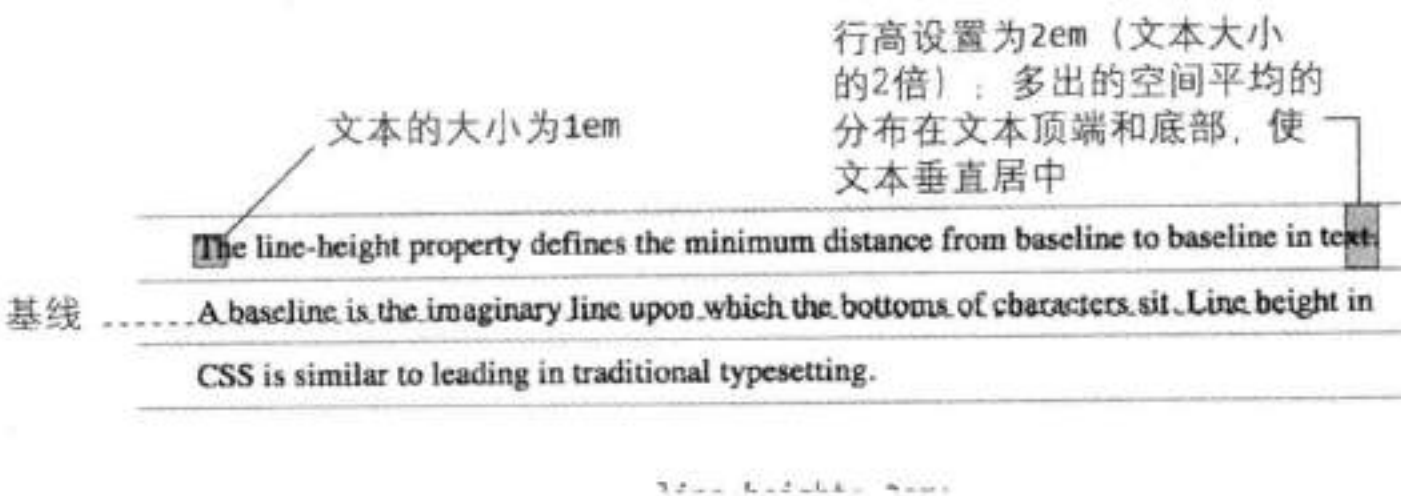


图12-13: 在CSS中, 行高是从基线到基线的距离, 而浏览器把文本放在垂直方向的中间

注意: `Text-indent`属性只缩进块的第一行。如果你想缩进一个文本块整边的空格, 可以添加`margin`属性或者`padding`属性。

设计师可能习惯于同时指定缩进和页边距, 但是为了与CSS处理它们的方法保持一致, 指定页边距将在第16章作为盒子模型的一部分来讨论。

缩进

`text-indent`属性使文本的第一行缩进具体数目的长度 (见“注意”)。

`text-indent`

属性值:	长度 百分比值 inherit
默认值:	0
适用对象:	块级元素和表格的单元格
是否可继承:	是

可以给`text-indent`属性指定长度值或百分比值。百分比值基于父元素计算得来。下面是一些例子, 结果如图12-14所示。

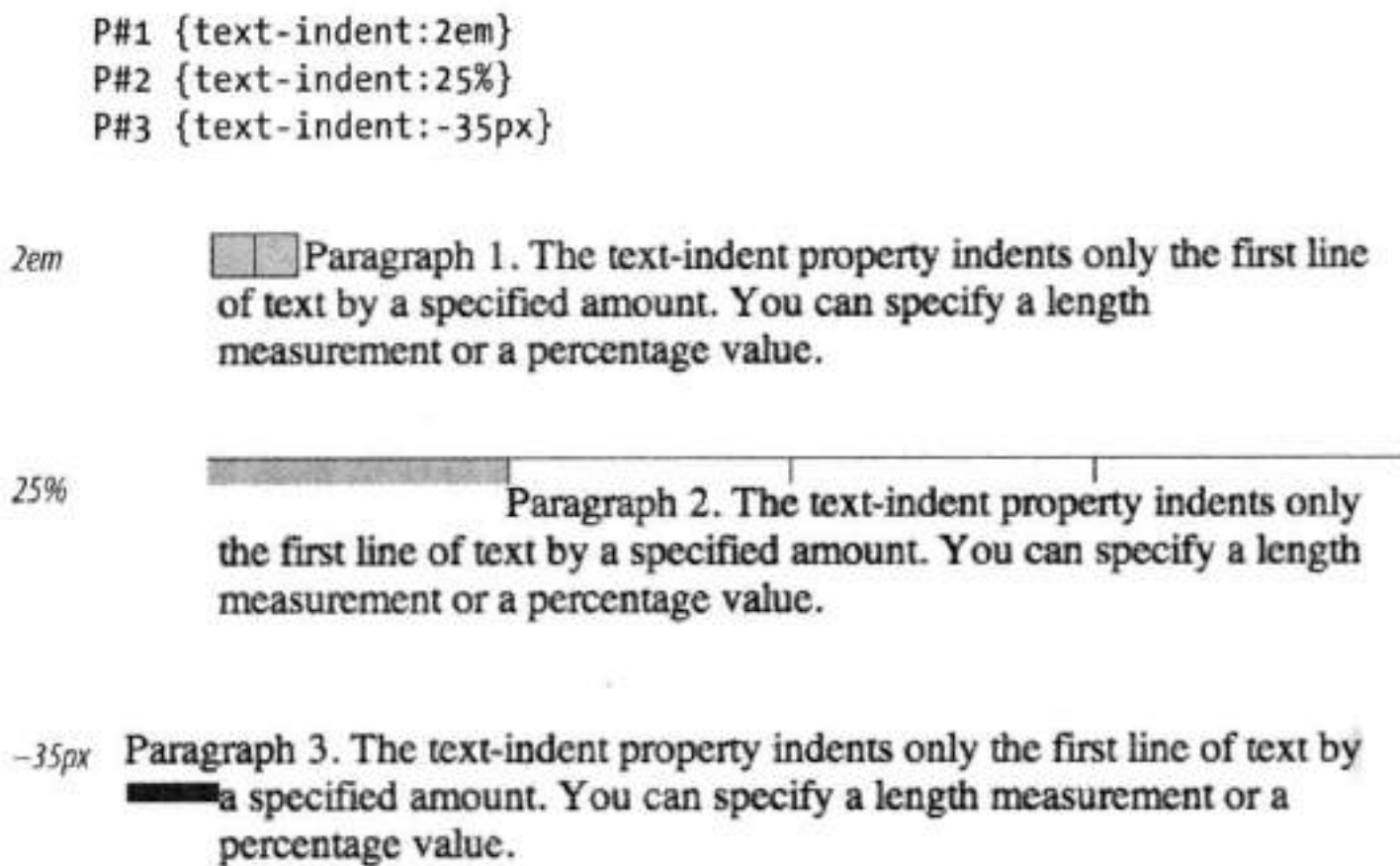


图12-14: `text-indent`属性的例子

注意第三个例子, 指定一个负值而运行正常。这样会导致文本的第一行左边缘挂出 (也称作挂起缩进)。

`text-indent`属性可继承, 但是计算结果值传递给后代元素没有价值。

所以如果div元素设置为800像素宽，10%的缩进，值为80像素的text-indent属性会向下传递（而不是10%）给div包含的元素。

水平对齐

通过指定text-align属性，在文字处理或桌面发布程序中，可以对齐网页文本。

text-align

属性值：	left right center justify inherit
默认值：	从左到右阅读的语言为left；从右到左阅读的语言为right
适用对象：	块级别的元素和表格的单元格
是否可继承：	是

可以相当直截地使用。各种text-align属性值的效果如图12-15所示。

text-align: left	文本左边对齐
text-align: right	文本右边对齐
text-align: center	文本位于文本块的中间
text-align: justify	文本与左右边都对齐

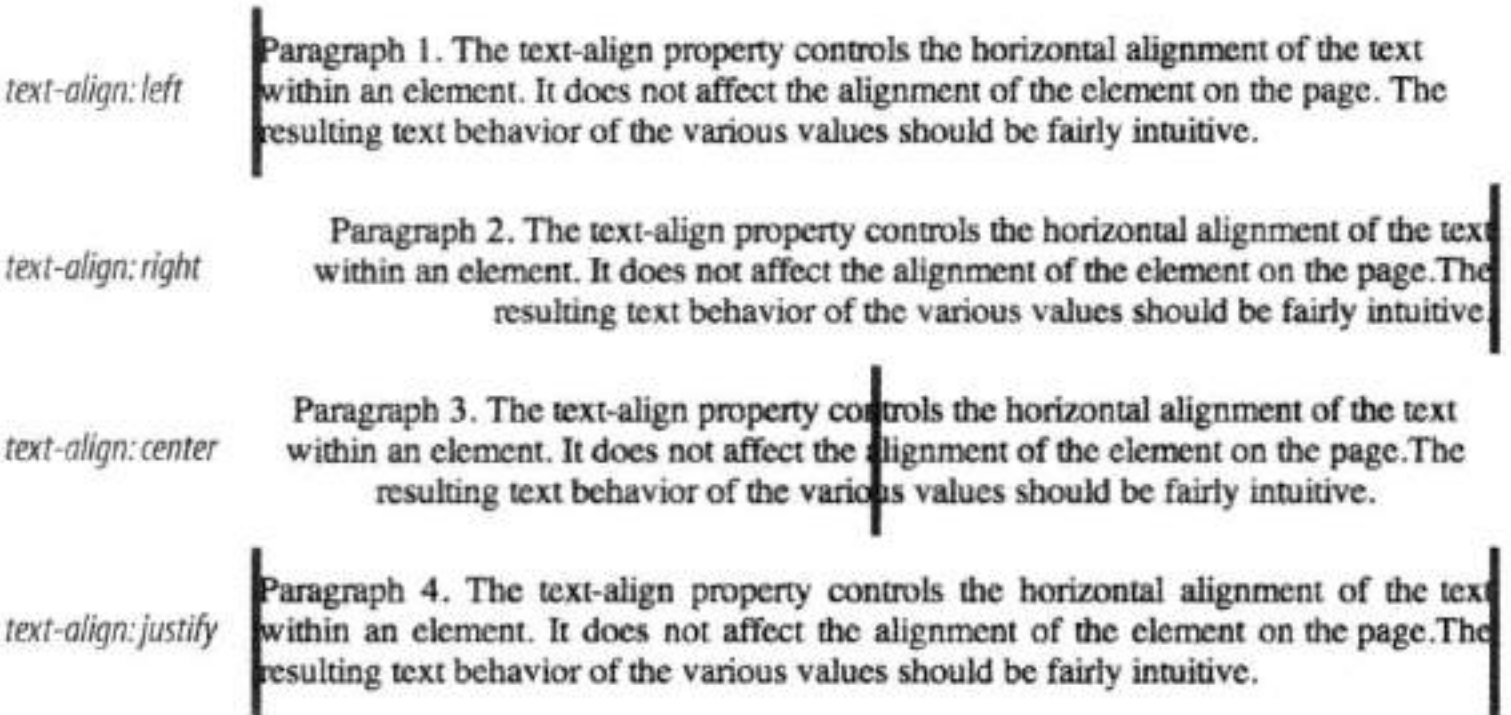


图12-15：text-align属性值的例子

好消息：还只剩4个属性可以学习！我们将做好准备在Black Goose Bistro的菜单上试试效果。

设计提示

如果使用挂起的缩进，确保也要对元素应用左边距属性。否则，挂起的文本可能会从浏览器窗口的左边缘消失。

注意：CSS3文本模块定义了两个相关的属性：text-align-last（对齐最后一行）和text-justify（为了更好的对齐控制，指定如何在文本中插入空格），但在写本书时，这些属性还没有获得完全的支持。

I've got laser eyes.

text-decoration: underline

I've got laser eyes.

text-decoration: overline

I've got laser eyes.

text-decoration: line-through

图12-16: text-decoration属性值的例子

注意: CSS3文本模块进一步增强了text-decoration, 包含text-decoration-line、text-decoration-color、text-decoration-style、text-decoration-skip和text-decoration-skip-position, 但是在写本书时, 这些还都处于试验阶段。

下划线和其他的“修饰”

如果想放一根线在文本之上、之下或者横穿文本, 或是如果你想去掉链接的下划线, 那么text-decoration属性正是你所需要的。

text-decoration

属性值:	none underline overline line-through blink
默认值:	none
适用对象:	所有元素
是否可继承:	否, 但是因为线也会在子元素通过, 所以它们看起来也被“修饰”了。

text-decoration的属性值很直观, 如图12-16所示。

text-decoration: underline	元素下面划线
text-decoration: overline	文本上面划线
text-decoration: line-through	横穿文本划线
text-decoration: blink	使文本闪烁

text-decoration属性的最常见用法是取消链接自动出现的下划线, 如下:

```
a { text-decoration: none; }
```

对于text-decoration属性, 有几点提醒:

- 首先, 如果想去掉链接下的下划线, 确保有其他的提示作为补偿, 如颜色和字重。
- 相对地, 因为下划线是“点这里”的强烈视觉提示, 所以给非链接文本添加下划线会产生误导和混淆。请考虑使用斜体来代替。
- 最后, 没有理由让你的文本闪烁。也不要那么做。Internet explorer浏览器对此尚不支持。

改变字母大写

我记得, 当年桌面发布程序推出了一项特殊的功能, 自动实现字母大写。这样很容易看到标题字母全部大写的效果, 而不需要重新输入。CSS中text-transform属性包含这个功能。

text-transform

属性值:	none capitalize lowercase uppercase inherit
默认值:	none
适用对象:	所有元素
是否可继承:	是

当你应用text-transform属性到文本元素时，看到的大小写改变了，而不用在源代码里修改。属性值如下（图12-17）：

text-transform: none	与源代码里的一样
text-transform: capitalize	每个单词的第一个字母大写
text-transform: lowercase	所有字母小写
text-transform: uppercase	所有字母大写

And I know what you're thinking.

text-transform: none (as was typed in)

And I Know What You'Re Thinking.

text-transform: capitalize

and i know what you're thinking.

text-transform: lowercase

AND I KNOW WHAT YOU'RE THINKING.

text-transform: uppercase

图12-17：text-transform属性改变字符显示时的大小写，不管源代码中是怎样的

空格

本章的最后两个文本属性用来在显示的字母之间（letter-spacing）和单词之间（word-spacing）插入空格。

letter-spacing

属性值:	长度值 normal inherit
默认值:	normal
适用对象:	所有元素
是否可继承:	是

word-spacing

属性值: 长度值|normal|inherit
 默认值: normal
 适用对象: 所有元素
 是否可继承: 是

letter-spacing和word-spacing属性的作用就像它的名字一样显而易见: 添加空格到文本的字母中 (letter-spacing) 或者字行的单词间 (word-spacing)。图12-18显示了应用这些规则的效果。

```
<p>Black Goose Bistro Summer Menu</p>
```

例一

```
p { letter-spacing: 8px; }
```

例二

```
p { word-spacing: 1.5em; }
```

值得一提的是, 当指定一个em计量值, 这个计算后的值会传递给子元素, 即使子元素字体尺寸小于父元素。

练习12-7中, 我们将最后一次光顾Black Goose Bistro的菜单, 添加一些新的属性, 再作一次改动。

```
letter-spacing: 8px;
```

B l a c k G o o s e B i s t r o S u m m e r M e n u

```
word-spacing: 1.5em;
```

Black Goose Bistro Summer Menu

图12-18: letter-spacing属性 (上) 和word-spacing属性 (下)

文本阴影

给页面中的文本以及图形元素设置“流行”的阴影是过去十年流行的做法。有一种方法可以给带有text-shadow属性的CSS的文本添加阴影。文本阴影显示在文本之后, 但又在背景和边框之前。

除了IE以外的所有浏览器均支持文本阴影, 不过据说IE10会支持文本阴影。

text-shadow

属性值：水平移动、垂直移动、模糊半径、颜色|none

默认值：none

适用对象：所有元素

是否可继承：是

text-shadow属性值应由颜色和三个计量值（水平移动、垂直移动和模糊半径）决定。最简短的文本阴影声明的例子参见图12-19。

<pre>H1{ color:darkgreen; text-shadow:.2em.2em silver; }</pre>	<pre>h2 { color:darkgreen; text-shadow:-.3em -.3em silver; }</pre>
--	--

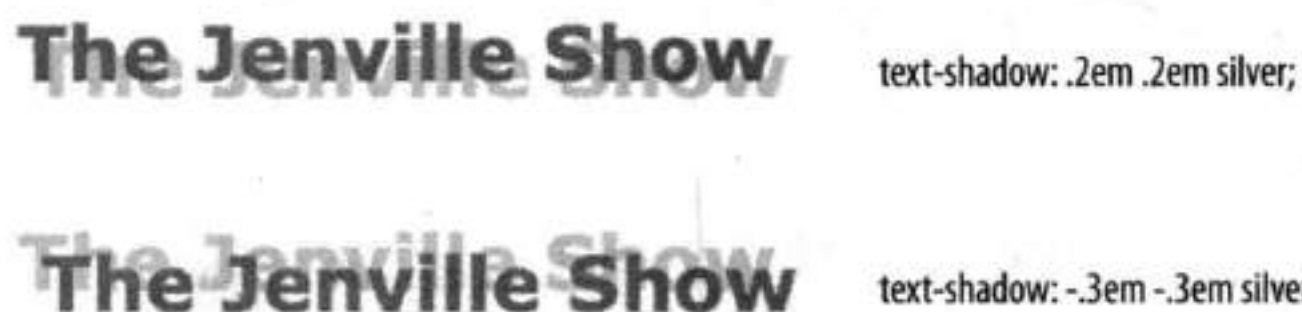


图12-19：一个最简短的文本阴影

第一个计量值是水平偏移，即将阴影定位于文档的右方某位置。第二个是垂直移动，将阴影下移指定的数量值。颜色指定是最后一个声明。如果颜色属性被遗漏，那么阴影颜色就与整个文本相同。

以上内容应该让你对前两个计量值的效果有了一定的了解，但是阴影的形状却不好理解。它所需要的是一个模糊的半径计量。Zero(0)表示不模糊，值越高，则模糊半径就越柔和（图12-20）。一般而言，你只需要调整计量值，直至达到你想要的效果。

你甚至可以对一个文本元素使用多种阴影。当多种阴影效果列出时，清单中的第一个会被最先应用，随后的阴影效果都会是在第一个的基础上显示的。你也可以通过直接定位色彩阴影使得文本高亮显示。图12-21演示了使用text-shadow的一些技术。

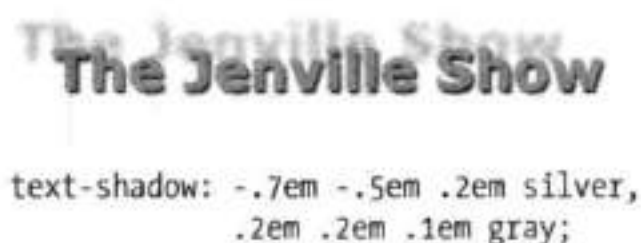
你可以好好体验下文本阴影，但不要用得太过夸张。阴影不仅会使文本难以阅读，而且添加阴影会降低网页性能（如滚动条、鼠标响应等），这对于缺乏处理能力的移动浏览器是个大问题。此外，注意不要使阴影遮住了文本本身。使用不支持阴影的浏览器的人们不会遇到这样的情况。

我的建议是将阴影作为一种锦上添花的效果，确保即便阴影不出现，也不会带来负面的影响。

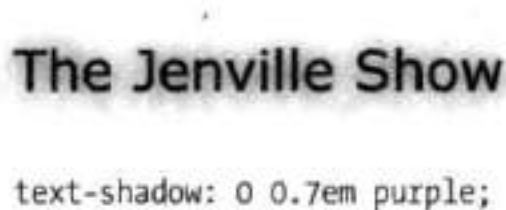


图12-20：对文本阴影添加模糊半径

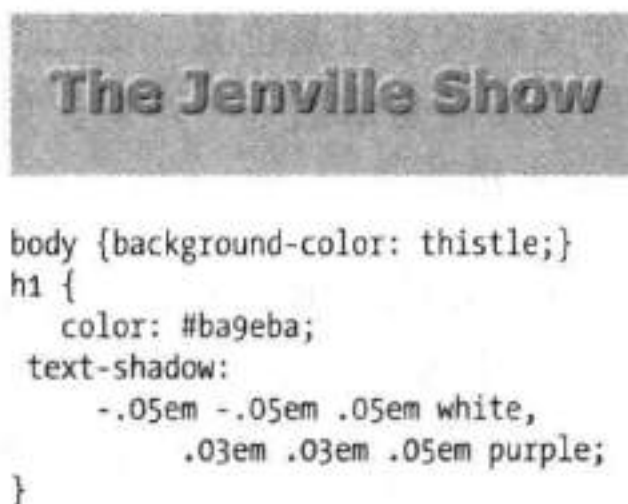
多个阴影



高亮显示

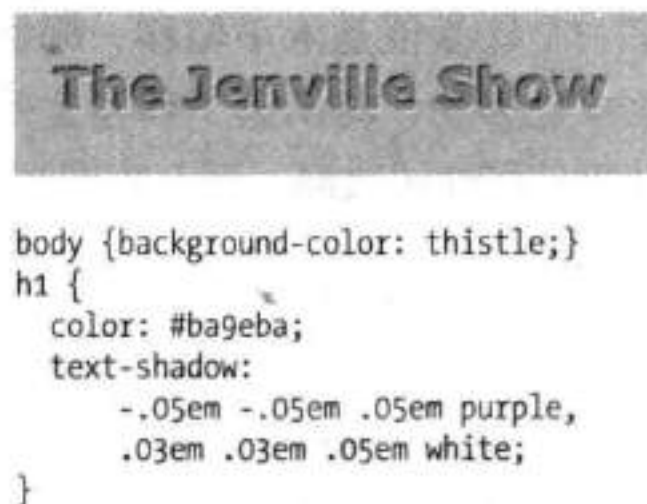


凸显效果



要达到凸显的效果，可以使用微小的偏移，在文本上方设置浅色的阴影，并在文本背后设置深色的阴影

浮雕效果



要达到浮雕的效果，浅色的阴影放置在文本下方，深色阴影则显示在文本上方

图12-21：文本阴影的特殊效果

下面来说一下不支持阴影的浏览器，是IE吗？IE 9以及更早的版本识别不了文本阴影属性，但它们有别的替代法。接下来的文章，包括Zoltan的Du Lac Hawryluk 都讨论了本书之外你需要去探究的替代方法。我建议好好搜索一下当今使用最多的方法。

- “Full CSS Text Shadows-Even in IE” (www.useragentman.com/blog/2011/06/29/full-css3-text-shadows-even-in-ie/)

- “CSS Text Shadow-Can It Be Done in IE Without JavaScript? (www.useragentman.com/blog/2011/04/14/css3-text-shadow-can-it-be-done-in-ie-without-javascript/)

其他文本属性

为了节省空间，作为一本入门级的书，一些属性这里并没有讲到过。但是我想作为一个无所保留，倾囊相授的作家，在这里需要介绍它们。

vertical-align

属性值: baseline|sub|super|top|text-top|middle|textbottom|bottom|percentage|length|inherit

指定一个内嵌元素的基线在垂直方向与周围文本的基线对齐。它同样适用于给单元格(td元素)设定内容的垂直对齐方式。

white-space

属性值: normal|pre|nowrap|pre-wrap|pre-line|inherit

指定布局中空格的处理方法。例如，属性值为pre表示，空字符与源码中保持一致，类似于HTML中的pre元素。

visibility

属性值: visible|hidden|collapse|inherit

用来隐藏元素。设定为hidden值时，元素不可见，但是它占的空间还在，只会在正文中留一个空白。元素还在，只是你看不到。

text-direction

属性值: ltr|rtl|inherit

指定文本的阅读方向，从左到右(ltr值)或从右到左(rtl值)

unicode-bidi

属性值: normal|embed|bidi-override|inherit

与Unicode的双向性有关。建议运用Unicode双向运算法则，允许作者定义内嵌的级别。如果你不明白它的意思，不用担心，我也不明白。

Font-size-adjust

属性值: number|none

对确保后备字体使用时的一致性而言，基于X高度的文档元素的尺寸调整，是一个相当复杂的新系统。我将用W3C解释余下部分: www.w3.org/TR/css3-fonts/#font-size-adjust-prop.

练习12-7 完成菜单

我们再次润饰在线菜单menu.html。每一步完成之后，保存文件，在浏览器中看看运行的效果，确保每一步都正确。最终完成的样式表见附录A。

1. 首先，记得我已经对body元素作过一些全局改动。对于字体库，我改变了想法。我认为serif字体比Georgia字体更适合用于小酒馆的菜单。再运用line-height属性增大文本的行高，使之更易阅读。然后更新body元素的样式规则，如下所示：

```
body {
  font-family: Georgia, serif;
  font-size: small;
  line-height: 1.75em;
}
```

2. 我也想重新设计文档的header部分。首先，删除整

条规则来移除teal的颜色设定。完成之后，使h1元素为紫色，header部分的段为灰色。可以在已有的规则上添加颜色修饰。

```
#info { color: teal; } /* delete */
h1 {
  font: bold 1.5em "Marko One", Georgia, serif;
  color: purple; }
#info p {
  font-style: italic;
  color: gray}
```

3. 接着，模仿奇特的打印菜单，运用text-align属性将网页中的关键元素置于中心。用组合选择器写一些规则，标题和info部分放在中央。

```
h1, h2, #info {
  text-align: center; }
```


设计提示

增加字母间距到小型文本是我最喜欢的标题设计技巧之一。为了增加元素的吸引力，在大型文本中增加字母间距是一个好的选择。

练习12-7 完成菜单（续）

4. 我想将ID为“Appetizer”和“Main Course”的h2标题变得特殊一些。不采用大而粗的样式，我想将所有字母大写，多加字母间空格和颜色，从而使标题更引人注目。下面是h2元素的新规则，包含以上所有的改动。

```
h2 {
  font-size: 1em;
  text-transform: uppercase;
  letter-spacing: .5em;
  color: purple; }
```

5. 现在我们马上就完成了，只剩在h2标题下的这些段落的几个调整了。将这些居中并设为斜体。

```
H2 + p {
  Text-align:center;
  font-style: italic; }
```

注意：在IE 6中，使用相邻同胞选择器（h2+p）来选定h2后的任何段落是无效的，应该选用上下文选择器#appetizers p和#entrees p。

6. 接下来，给菜单项名称（在dt元素内）添加浅一点的颜色。选用CSS3颜色模块中的sienna。注意，dt元素内的strong元素将保持栗色，因为strong元素中的颜色会覆盖从父元素继承来的颜色。

```
dt {
  font-weight: bold;
  color: sienna; }
```

图12-22：格式化后的黑天鹅小酒馆菜单

Black Goose Bistro • Summer Menu

Baker's Corner, Seekonk, Massachusetts

HOURS: MONDAY THROUGH THURSDAY: 11 to 9, FRIDAY AND SATURDAY: 11 to midnight

APPETIZERS

This season, we explore the spicy flavors of the southwest in our appetizer collection.

Black bean purses

Spicy black bean and a blend of mexican cheeses wrapped in sheets of phyllo and baked until golden. \$3.95

Southwestern napoleons with lump crab — *new item!*

Layers of light lump crab meat, bean and corn salsa, and our handmade flour tortillas. \$7.95

MAIN COURSES

Big, bold flavors are the name of the game this summer. Allow us to assist you with finding the perfect wine.

Jerk rotisserie chicken with fried plantains — *new item!*

Tender chicken slow-roasted on the rotisserie, flavored with spicy and fragrant jerk sauce and served with fried plantains and fresh mango. *Very spicy.* \$12.95

Shrimp sate kebabs with peanut sauce

Skewers of shrimp marinated in lemongrass, garlic, and fish sauce then grilled to perfection. Served with spicy peanut sauce and jasmine rice. \$12.95

Grilled skirt steak with mushroom fricasee

Flavorful skirt steak marinated in asian flavors grilled as you like it*. Served over a blend of sauteed wild mushrooms with a side of blue cheese mashed potatoes. \$16.95

* We are required to warn you that undercooked food is a health risk.

7. 最后，在h1标题下添加阴影。

```
h1 {
  font:bold 1.5em "Marko
One",Georgia, serif;
  color:purple
  text-shadow:.1em.1em .2em
lightslategray;}
```

终于完成了！图12-22显示了菜单现在的样子，比没有定义样式的版本好多了，我们是只用文本属性做到的。注意，我们在此过程中并没有动文档中的一个字符。这就是结构与样式分离的美妙。

改变列表的数字编号

在本章结束有关文本属性之前，我想告诉你们一些为列表编号的方法。如你所知，在众多有序、无序列表之前，浏览器已经自动嵌入数字编号。大多数情况下，标记的显示是由浏览器决定的。然而，CSS提供了一些可以让作者自己选择标记的使用、类型和位置的属性。

选择标记

使用list-style-type属性去选择出现在每个列表项之前的标记类型。

list-style-type

- 计量值: none|disc|circle|square|decimal|decimal-leading-zero|lower-alpha|upper-alpha|lower-latin|upper-latin|lower-roman|upper-roman|lower-greek|inherit
- 默认值: disc
- 适用对象: ul、ol和li（或显示值是list-item的元素）
- 是否可继承: 是

通常，开发者使用list-style-type属性，通过将这个属性值设置为none来移除数字编号。在制作水平导航菜单或者网络表单目录时，使用列表标记是很方便的。你知道它的意思就够了，不必使用这些麻烦的标记。

disc、circle和square计量值形成了如同浏览器运行时的编号形状（图12-23）。遗憾的是，不能对形成的编号的外观（尺寸、颜色等）进行修改，所以基本上还是使用浏览器的默认方式。

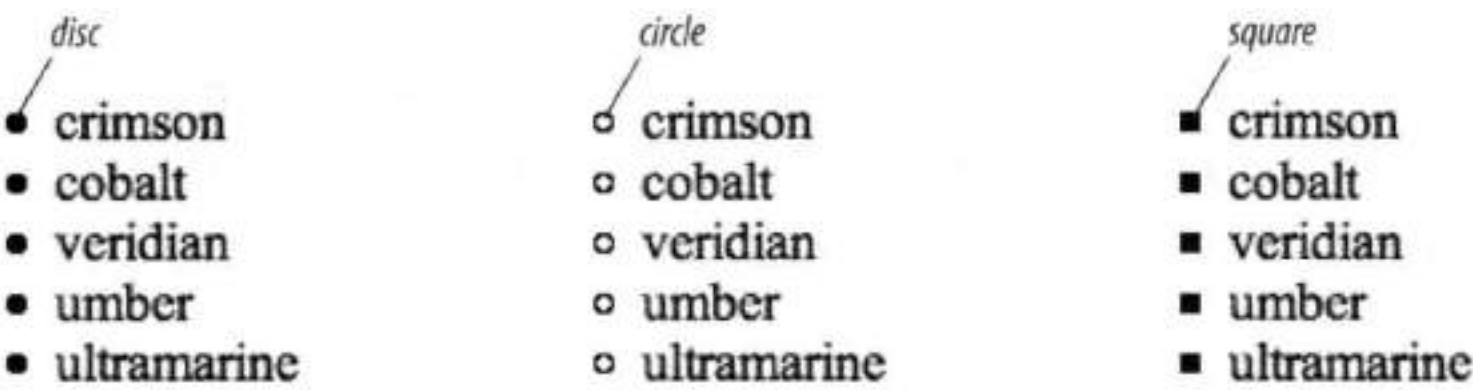


图12-23: list-style-type的计量值 disc、circle和square

注意：当前浏览器支持CSS2.1list-style类型的这部分文档。CSS3在标记功能方面有所延伸，包括作者定义自己的列表样式的方法，并且支持多种语言版本。（www.w3.org/TR/css3-lists/）。

注意：CSS3运用其新的@counter-style规则，引入了box、check、diamond和dash marker types类型。规约里有详细说明。

列表项目显示作用

你可能注意到了，列表样式应用于“显示为列表项目的元素”。通过设置display属性到list-item，CSS2.1 说明书中允许元素像列表项目一样执行。这个属性可用于任何HTML元素或者其他在XML语言中的元素。例如，你能够通过设定段落元素属性为list-item来将一系列段落自动编号。如下例所示：

```
p.bulleted {
  display: list-item;
  list-style-type: upper-alpha;
}
```

注意：CSS3为这个属性添加了hanging值。这与inside类似，但是标记会显示在外边，同时靠近阴影区域的左边界，如图12-24所示。

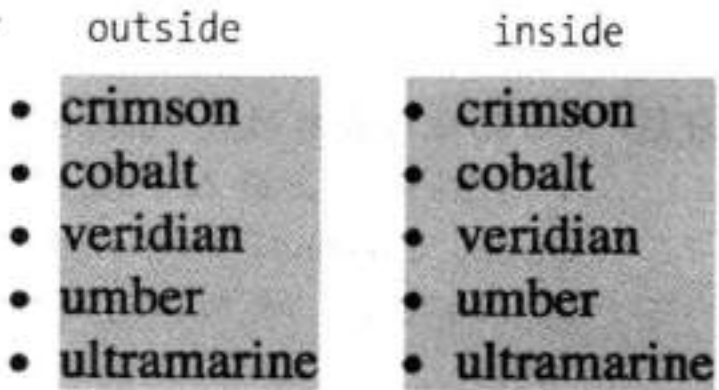


图12-24：list-style-position属性

余下的关键词（表12-1）指定了不同的数字和字母类型

表12-1字母和数字系统 (CSS2.1)

Keyword	System
decimal	1, 2, 3, 4, 5...
decimal-leading-zero	01, 02, 03, 04, 05...
lower-alpha	a, b, c, d, e...
upper-alpha	A, B, C, D, E...
lower-latin	a, b, c, d, e...（与lower-alpha相同）
upper-latin	A, B, C, D, E...（与upper-alpha相同）
lower-roman	i, ii, iii, iv, v...
upper-roman	I, II, III, IV, V...
lower-greek	α, β, γ, δ, ε ...

标记位置

默认情况下，标记处于目录区域的外边，显示为悬挂式缩进。list-style-position属性允许将编号置于目录区域内，成为列表内容。

list-style-position

- 属性值：inside|outside|inherit
- 默认值：outside
- 适用对象：ul、ol和li（或显示值为list-item的元素）
- 是否可继承：是

为了显示目录区域的边界，我已经在图12-24的列表项目中应用了背景颜色。当位置设为outside时，你可以看到标记超出了目录区域；当设为inside时，标记包括在目录区域内。

```
li {background-color: #F99;}
ul#outside {list-style-position: outside;}
ul#inside {list-style-position: inside;}
```

制订自己的编号

你也可以将自己的图像作为编号用于list-style-image属性。

list-style-image

- 属性值：url|none|inherit
- 默认值：none

适用对象： ul、ol和li（显示值为list-item的元素）

是否可继承： 是

List-style-image属性值是作为标记的图像的URL。list-style-type设disc为备份，以防图像不显示或属性不被浏览器或其他用户设备所支持。结果如图12-25所示。

```
ul {
  list-style-image: url(/images/happy.gif);
  list-style-type: circle;
  list-style-position: outside;
}
```




 Puppy dogs
 Sugar frogs
 Kitten's baby teeth

图12-25：使用图像作为标记

自我测验

下面有几个问题，看看你对选择器和文本格式化的基础的掌握情况。

1. 这是一个写选择器的实践机会。利用图12-26中的图表，写一些规则，让下列说明中每一个元素都显示为红色（color:red;）。尽可能有效地编写选择器，我已经给你写了第一个。

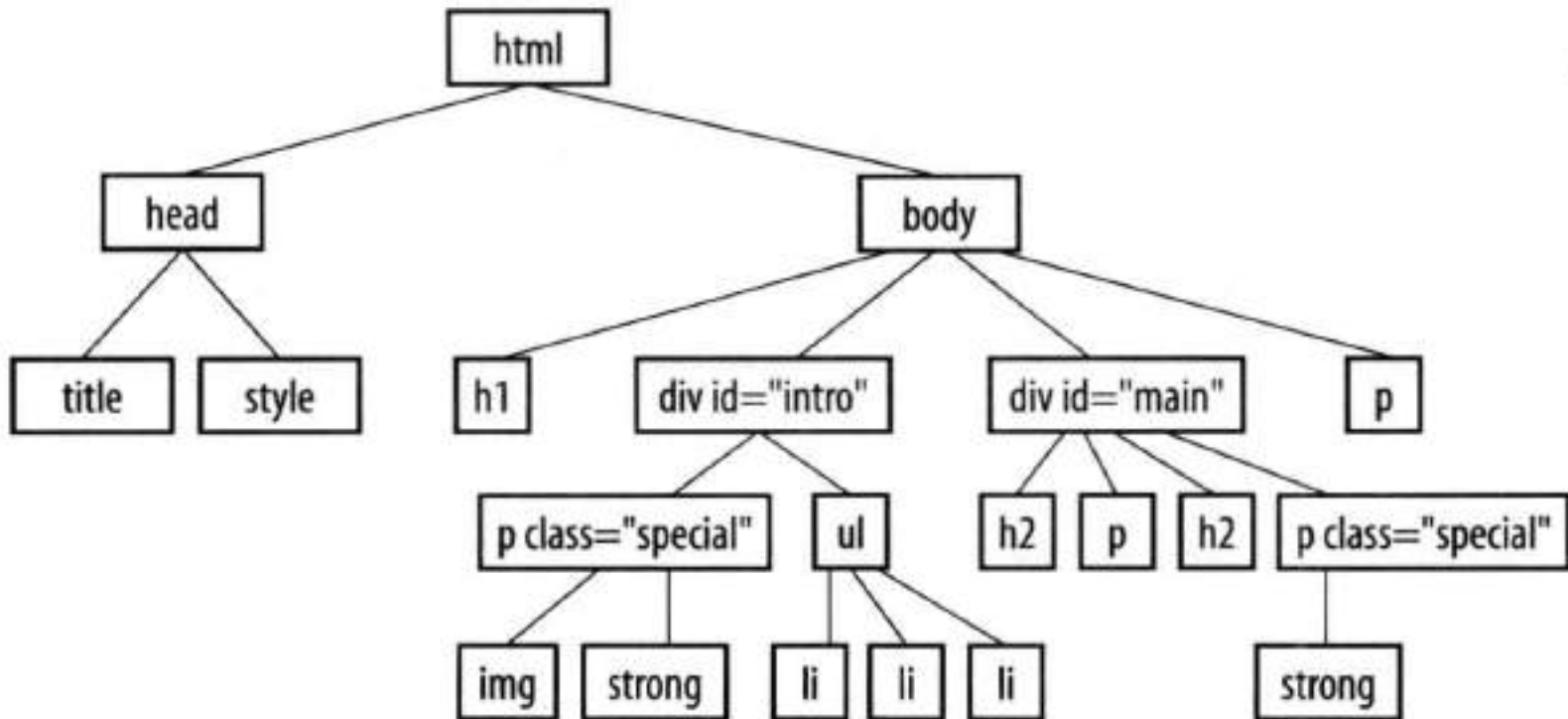


图12-26：文档结构例子

- a. 文档中所有文本元素body {color: red;}
- b. h2元素
- c. h1元素和所有的段落
- d. 属于“special”类的所有元素
- e. “intro”块中的所有元素
- f. “main”块中的所有strong元素
- g. 附加题：只要h2元素之后出现的段（提示：在IE 6中这个选择器将无效）

注意：list-style快捷属性，将样式值、定位值和图像值等结合起来。例如：

```
ul { list-style: url(/images/happy.gif) circle outside; }
```

对于所有的快捷属性，要注意不要覆盖之前样式表中的列表样式属性。

2. 根据图12-27将样式属性与文本例子配对。

- a. ____ {font-size: 1.5em;}
- b. ____ {text-transform: capitalize;}
- c. ____ {text-align: right;}
- d. ____ {font-family: Verdana; font-size: 1.5em;}
- e. ____ {letter-spacing: 3px;}
- f. ____ {font: bold italic 1.2em Verdana;}
- g. ____ {text-transform: uppercase;}
- h. ____ {text-indent: 2em;}
- i. ____ {font-variant: small-caps;}

default font and size

Look for the good in others and they'll see the good in you.

- ❶ Look For The Good In Others And They'll See The Good In You.
- ❷ Look for the good in others and they'll see the good in you.
- ❸ Look for the good in others and they'll see the good in you.
- ❹ Look for the good in others and they'll see the good in you.
- ❺ Look for the good in others and they'll see the good in you.
- ❻ LOOK FOR THE GOOD IN OTHERS AND THEY'LL SEE THE GOOD IN YOU.
- ❼ Look for the good in others and they'll see the good in you.
- ❽ LOOK FOR THE GOOD IN OTHERS AND THEY'LL SEE THE GOOD IN YOU.
- ❾ **Look for the good in others and they'll see the good in you.**

图12-27: 文本例子

CSS回顾：字体和文本属性

本章中，我们学完了用于格式化文本元素的所有属性。下面是按字母顺序所作的总结。

属性	描述
font	多种字体属性组合的快捷属性
font-family	指定字体和通用字体库
font-size	字体的尺寸
font-style	指定italic或oblique字体
font-variant	指small-caps字体样式
font-weight	指定字体的粗细
letter-spacing	字母之间插入空格
line-height	相邻行基线之间的距离
text-align	文本水平对齐方式
text-decoration	下划线、上划线和贯穿线
text-direction	文本从左到右阅读还是从右到左阅读
text-indent	文本块中第一行的缩进量
text-transform	在文本下面添加阴影点
text-transform	改变显示时文本的大小写
unicode-bidi	使用unicode双向运算法则
vertical-align	调整内嵌元素相对于基线的水平位置
visibility	元素是否显示或可见
white-space	空格像源代码里一样显示
word-spacing	在单词之间插入空格

第13章 颜色和背景（附加更多选择器和外部样式表）

你见过1993年的Web吗？按照现在的标准，你会发现那时的网页是相当沉闷的，所有背景都是灰色，所有的文字都是黑色。然后有了Netscape Navigator浏览器，用极少的属性简单（但受欢迎）的控制字体颜色和背景。这么多年来，我们都是这样做的。但值得庆幸的是，现在样式表里的属性把那些旧属性淘汰了。

本章将讲述很多内容，将介绍指定颜色和背景的所有属性。本章还会谈到所有选择器类型的集合，告诉你如何创建外部样式表。首先要谈论的是CSS中指定颜色的选择项，我们还会初步介绍计算机显示器的颜色本质。

指定颜色值

在样式表中有两种主要的方法来指定颜色值：我们一直在用的方法——使用预定的颜色名称：

```
color: red; color: olive; color: blue;
```

或者，使用更广泛的、具体的RGB三原色（计算机显示器的颜色模型）数值。你应该看到过以这种方式表达的颜色值：

```
color: #FF0000; color: #808000; color: #00F;
```

过一会儿再详细讲述RGB三原色，先来学习简短轻快的标准颜色名称。

颜色名

最直观的指定颜色的方法是直呼其名，遗憾的是，我们不能给颜色任意起

本章内容

CSS中的颜色名称

指定RGB颜色值

前景和背景色

伪类和伪元素选择器

添加拼贴背景图像

渐变色

外部样式表

注意：扩展的颜色名也叫做X11颜色名，是由Unix的X Window系统创建的。

名并且期望这些名字起作用。这些名字必须是CSS介绍中预定的颜色关键字之一。CSS1和CSS2采用原本从HTML4.01中引入的16个标准颜色名。CSS2.1引入orange后，总共17个（图13-1）。CSS3增加了可以支持140个可供选择颜色名的模块。现在对于burlywoody以及我长久以来的偏爱papayawhip，我已经可以对它们指定名字了！扩展的颜色显示在图13-2中，但是如果你想有更详细的了解，可以访问www.learningwebdesign.com/colornames.html。

颜色名很容易使用——只需要放到任意颜色相关属性的属性值处：

```
color: silver;
background-color: gray;
border-bottom-color: teal;
```

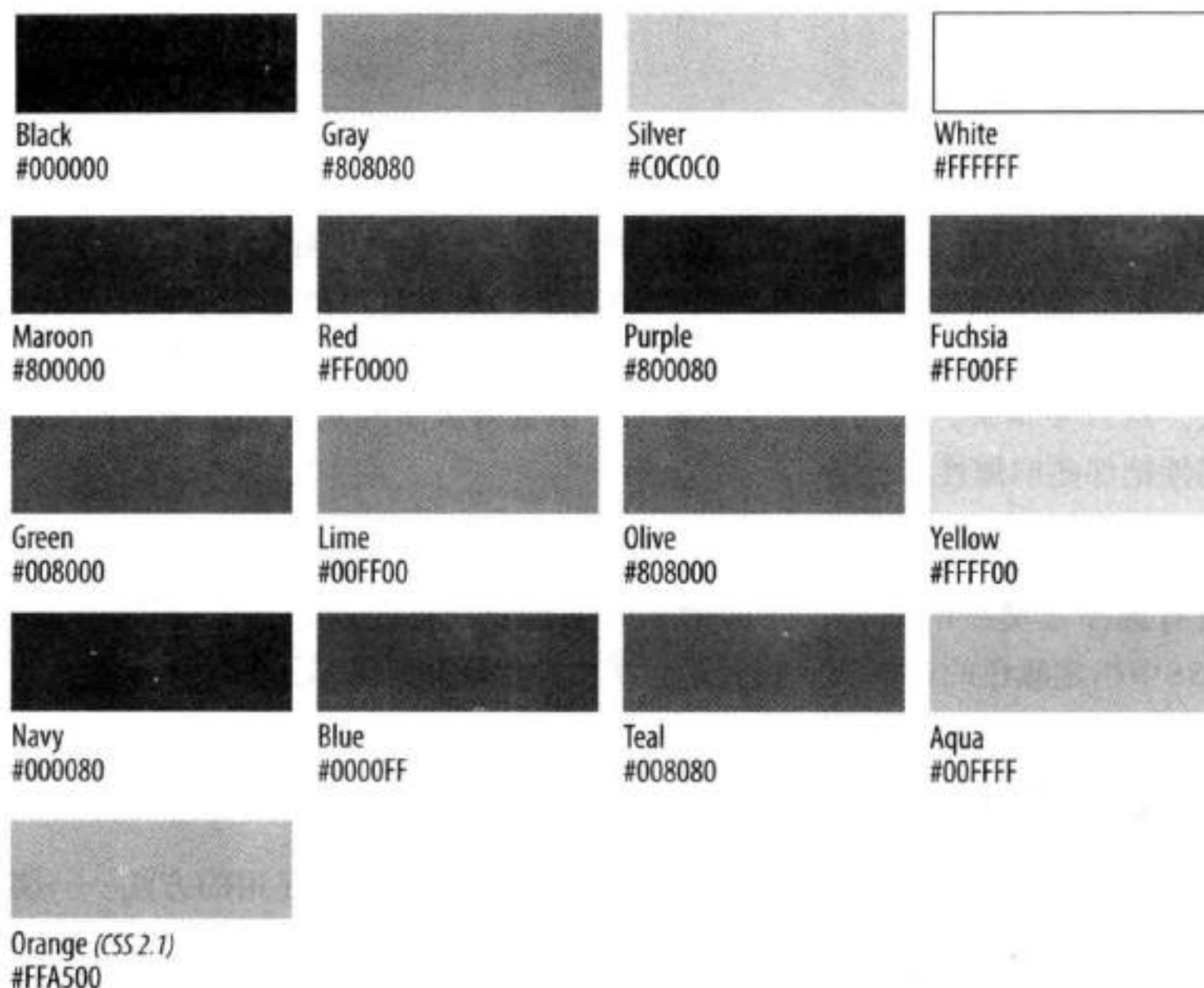


图13-1：CSS2.1中的17个标准颜色名

RGB颜色值

颜色名称很好用，但如你所见，它们是有限的。到目前，指定颜色最普遍的方法还是使用RGB值。它也提供了无数的颜色供你选择。

对于不熟悉计算机如何处理颜色的人来说，在进入CSS语法之前，先从基础的开始学习。

aliceblue 240, 248, 255 F0F8FF	cornsilk 255, 248, 220 FFF8DC	darkturquoise 0, 206, 209 00CED1	hotpink 255, 105, 180 FF69B4	lightskyblue 135, 206, 250 87CEFA	midnightblue 25, 25, 112 191970	peru 205, 133, 63 CD853F	snow 255, 250, 250 FFFAFA
antiquewhite 250, 235, 215 FAEBD7	crimson 220, 20, 60 DC143C	darkviolet 148, 0, 211 9400D3	indianred 205, 92, 92 CD5C5C	lightslategray 119, 136, 153 778899	mintcream 245, 255, 250 F5FFFA	pink 255, 192, 203 FFC0CB	springgreen 0, 255, 127 00FF7F
aqua 0, 255, 255 00FFFF	cyan 0, 255, 255 00FFFF	deeppink 255, 20, 147 FF1493	indigo 75, 0, 130 4B0082	lightsteelblue 176, 196, 222 B0C4DE	mistyrose 255, 228, 225 FFE4E1	plum 221, 160, 221 DDA0DD	steelblue 70, 130, 180 4682B4
aquamarine 127, 255, 212 7FFFD4	darkblue 0, 0, 139 00008B	deepskyblue 0, 191, 255 00BFFF	ivory 255, 240, 240 FFFFD0	lightyellow 255, 255, 224 FFFFE0	moccasin 255, 228, 181 FFE4B5	powderblue 176, 224, 230 B0E0E6	tan 210, 180, 140 D2B48C
azure 240, 255, 255 F0FFFF	darkcyan 0, 139, 139 008B8B	dimgrey 108, 105, 105 696969	khaki 240, 230, 140 F0E68C	lime 0, 255, 0 00FF00	navajowhite 255, 222, 173 FFDEAD	purple 128, 0, 128 800080	teal 0, 128, 128 008080
beige 245, 245, 220 F5F5DC	darkgoldenrod 184, 134, 11 8B6914	slateblue 106, 144, 255 6A5ACD	lavender 230, 230, 250 E6E6FA	limegreen 50, 205, 50 32CD32	navy 0, 0, 128 000080	red 255, 0, 0 FF0000	thistle 216, 191, 216 D8BFD8
bisque 255, 228, 196 FFE4C4	darkgray 169, 169, 169 A9A9A9	firebrick 178, 34, 34 B22222	lavenderblush 255, 240, 245 FFF0F5	linen 250, 240, 230 FAFAD2	oldlace 253, 245, 230 FDF5E6	rosybrown 188, 143, 143 BC8F8F	tomato 255, 99, 71 FF6347
black 0, 0, 0 000000	darkgreen 0, 100, 0 006400	floralwhite 255, 250, 240 FFFAF0	lawngreen 124, 252, 0 7CFC00	magenta 255, 0, 255 FF00FF	olive 128, 128, 0 808000	royalblue 65, 105, 225 4169E1	turquoise 64, 224, 208 40E0D0
blanchedalmond 255, 255, 205 FFFACD	darkkhaki 189, 183, 107 BDB76B	forestgreen 34, 139, 34 228B22	lemonchiffon 255, 250, 205 FFFACD	mediumslateblue 106, 90, 205 6A5ACD	olivedrab 107, 142, 35 8A692D	saddlebrown 139, 69, 19 8B4513	violet 238, 130, 238 EE82EE
blue 0, 0, 255 0000FF	darkmagenta 139, 0, 139 8B008B	fuchsia 255, 0, 255 FF00FF	lightblue 173, 216, 230 ADD8E6	mediumaquamarine 102, 205, 170 66CDAA	orange 255, 165, 0 FFA500	salmon 250, 128, 114 FA8072	white 255, 255, 255 FFFFFF
bluenviolet 138, 43, 226 8A2BE2	darkolivegreen 85, 107, 47 556B2F	gainsboro 220, 220, 220 DCDCDC	lightcoral 240, 128, 128 F08080	mediumorchid 186, 85, 211 BA55D3	orchid 218, 112, 214 DA70D6	sandybrown 244, 164, 96 F4A460	wheat 245, 222, 179 F5DEB3
brown 185, 49, 42 A52A2A	darkorange 255, 140, 0 FF8C00	ghostwhite 248, 248, 255 F8F8FF	lightgoldenrodyellow 250, 250, 210 FAFAD2	mediumorchid 186, 85, 211 BA55D3	orange 255, 165, 0 FFA500	seagreen 46, 139, 87 2E8B57	whitesmoke 245, 245, 245 F5F5F5
burlywood 222, 184, 135 DEB887	darkred 139, 0, 0 8B0000	gold 255, 215, 0 FFD700	lightcyan 224, 255, 255 E0FFFF	mediumpurple 147, 112, 219 9370DB	palegoldenrod 238, 232, 170 EEE8AA	seashell 255, 245, 238 FFF5EE	yellow 255, 255, 0 FFFF00
cadetblue 95, 138, 160 5F9EA0	darkslateblue 153, 50, 204 9932CC	goldenrod 218, 165, 32 DAA520	lightgreen 144, 238, 144 90EE90	mediumslateblue 106, 90, 205 6A5ACD	palegreen 152, 251, 152 98FB98	sienna 160, 82, 45 A0522D	yellowgreen 154, 205, 50 9ACD32
chartreuse 127, 255, 0 7FFF00	darkslategray 233, 150, 122 E9967A	gray 128, 128, 128 808080	lightgray 211, 211, 211 D3D3D3	mediumslateblue 106, 90, 205 6A5ACD	paleturquoise 175, 238, 238 AFEEEE	silver 192, 192, 192 C0C0C0	skyblue 135, 206, 235 87CEEB
chocolate 210, 105, 30 D2691E	darkslategray 233, 150, 122 E9967A	green 0, 128, 0 008000	lightpink 255, 182, 153 FFB6C1	mediumslateblue 106, 90, 205 6A5ACD	paleturquoise 175, 238, 238 AFEEEE	skyblue 135, 206, 235 87CEEB	skyblue 135, 206, 235 87CEEB
coral 255, 127, 80 FF7F50	darkslateblue 72, 61, 139 483D8B	greenyellow 173, 255, 47 ADFF2F	lightsalmon 255, 160, 122 FFA07A	mediumslateblue 106, 90, 205 6A5ACD	papayawhip 255, 239, 213 FFEFD5	slateblue 106, 90, 205 6A5ACD	slateblue 106, 90, 205 6A5ACD
cornflowerblue 100, 149, 237 6495ED	darkslategray 47, 79, 79 2F4F4F	honeydew 240, 255, 240 F0FFD0	lightseagreen 32, 178, 170 20B2AA	mediumslateblue 106, 90, 205 6A5ACD	peachpuff 255, 239, 213 FFEFD5	slategray 112, 125, 144 708090	slategray 112, 125, 144 708090

图13-2: CSS3中的140个扩展颜色名。你需要明白在屏幕上这些色彩看起来可能不太一样

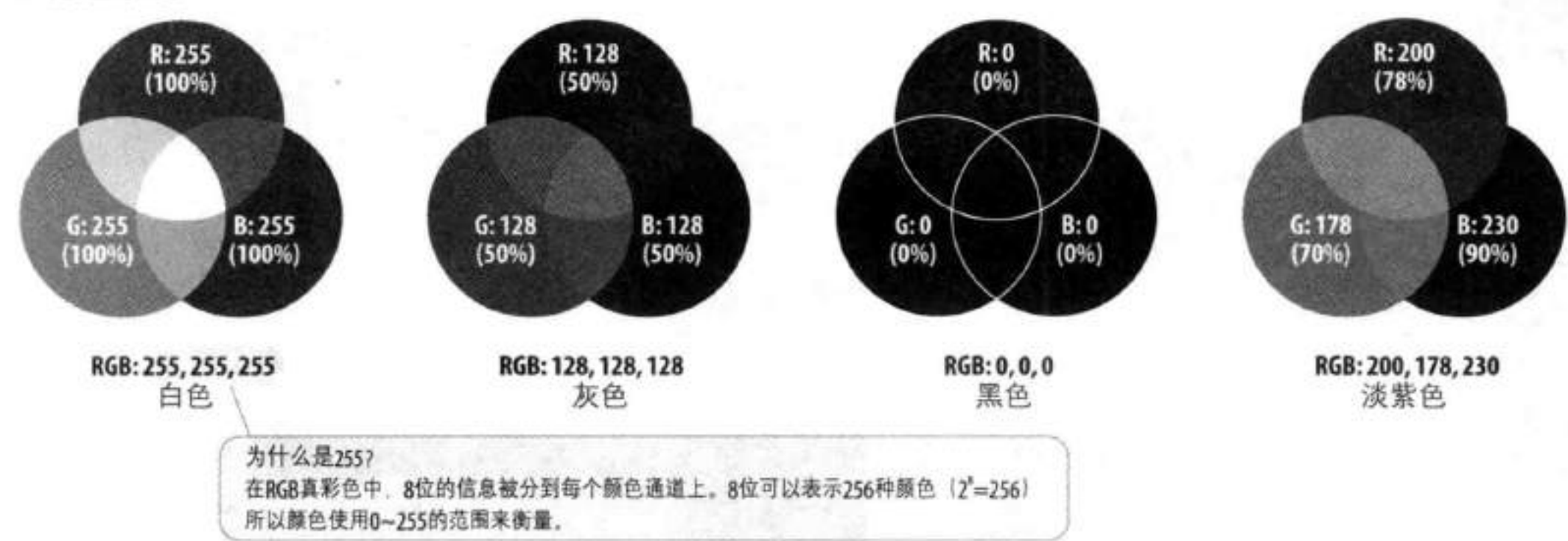
关于RGB颜色的一些看法

通过混合三色光：红、绿和蓝，计算机可以创造你在显示器上能看到的所有颜色，称作RGB颜色模型。通过告诉计算机每种光用多少混合，你可以开出一个颜色配方（各种各样的）。每种颜色“通道”中光的数量通常用一个0（没有）到255（充满）之间的值表示，也可以用百分比表示。三个值越接近255（100%），混合色越接近白色（如图13-3所示）。

显示器上看到的任何一种颜色都可以用这三个数的序列表示：红色值、绿色值和蓝色值。Adobe Photoshop等图像编辑器之所以能够知道图像中每个像素的颜色值，这便是其中一种方法。通过RGB颜色系统，令人愉悦的薰衣草的颜色可以用200，178，230来表示。

图13-3：计算机显示的颜色是由不同数量的红、绿、蓝光（即RGB）混合而成的。每个图中间的颜色就是三个颜色通道混合出来的。每个通道里的光越多，混合色就越接近白色

RGB颜色系统



拾取颜色

拾取颜色和查找RGB颜色值的最简单方法是使用图像编辑工具，比如Adobe Photoshop，Adobe Fireworks或Corel Paint Shop ProPhoto。大多数图像工具提供类似图13-4中Photoshop的颜色拾取器。如果你没有图像编辑工具，那么可以通过在线工具，如ColorPicker.com来选择颜色。

从显示器上定义颜色有很多种方法。其中的两种方法是RGB（Red，Green，Blue）和HSL（Hue，Saturation，Lightness），这两种方法与CSS相关。RGB是最常用且值得信赖的方法。下面将重点讲述这种方法，但也会在侧栏提供关于HSL颜色的更多信息。

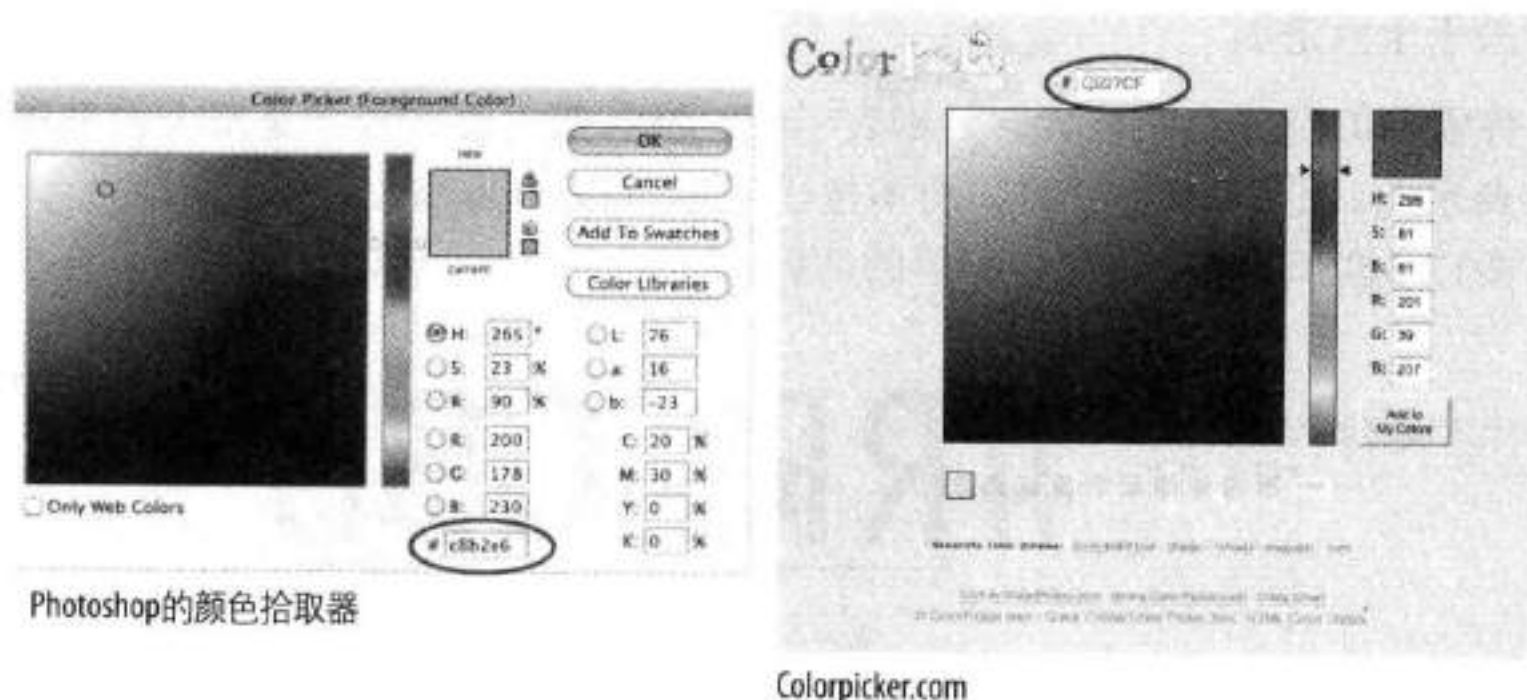


图13-4: Photoshop(左边)和Colorpicker.com(右边)的颜色拾取器,可以为选中的颜色像素提供RGB值

当通过颜色拾取器来从光谱中选择颜色时,就像图13-4中所看到的那样,你会发现红(R)、绿(G)和蓝(B)值都列出来了。再看#标志——这些值还是原来的值,只是转化为16进制,可以用在样式表中而已。待会儿,我将解释这个6位的16进制数值。

在样式表中写RGB值

CSS允许RGB值用数字的形式表示。再来看令人愉快的薰衣草花的颜色,我们可以列出每个0~255的值,添加到样式表中。

```
color: rgb(200, 178, 230);
```

也可以用百分比值列出,虽然这种用法不常见。

```
color: rgb(78%, 70%, 90%);
```

或者,也可以使用颜色拾取器中特意为网页准备的颜色版本表示。6位数与3个RGB数值表示的结果一样,只是把数字转化为十六进制(简称hex)值而已。在下一部分将解释十六进制。注意十六进制RGB值以#符号开头,而且不需要上面看到的rgb()符号。

```
color: #C8B2E6;
```

最后有一个指定十六进制颜色值的快捷方式。如果颜色值恰好由三对两位数表示,例如:

```
color: #FFCC00; 或者 color: #993366;
```

你可以把每对数缩成一位。这样做的好处是可以减小CSS文件的大小。下面的例子就等价于上面的色彩:

```
color: #FC0; 或者 color: 936;
```

HSL颜色

CSS3引入了使用HSL值来指定颜色的功能:色调(hue)、色饱和度(saturation)和色亮度(lightness)。图13-4所示的颜色拾取器也提供所选色彩的HSL值,但是它们把最后一个值称为Brightness(B)。

在这个系统中,颜色散布在彩虹图的周围,其中红色在顶部的位置(12点钟方向)。色调值以色图的度数来衡量:红色是0°、绿色是120°以及蓝色是240°,其他色彩在它们之间。饱和度是一个百分比值,从0%(灰色)到100%(完全饱和)。亮度也是一个百分比值,0%是黑色,100%为白色。

一些人觉得这个色彩系统用起来更直观,因为当你锁定一个色调时,很容易就可以使颜色加强、变深或变浅。RGB并不这么直观,虽然一些有经验的开发者有自己的经验。

在CSS中,HSL值是色调值,加两个百分比值。它们绝不会像RGB一样转为十六进制。下面是样式表中以HSL方式所表示的图13-3所示的薰衣草色彩:

```
color: hsl(265, 23%, 90%);
```

快速浏览

指定RGB值

有四种格式在CSS中提供RGB值:

```
rgb(255, 255, 255)
rgb(100%, 100%, 100%)
#FFFFFF
#FFF
```

这些例子表示的都是白色。

提示

便捷的十六进制值

白色=#FFFFFF或者#FFF（等价于255, 255, 255）

黑色=#000000或者#000（等价于0, 0, 0）

十六进制计算

在windows操作系统中，标准科学计算器中有一个十六进制转换器。Mac计算机用户可以下载免费的用于OSX操作系统的“Mac Dec Bin Calculator”软件（你可以在download.com中搜索到）。

当然，也可以手动计算十六进制值，把数值除以16得到第一位数，用余数作为第二位数。例如，200转换为C8，因为 $200 = (16 \times 12) + 8$ ，以16为基数结果是{12, 8}，在十六进制中表示为C8。我想我还是会一直使用颜色拾取器吧。

关于十六进制

你看到的是三个两位数，分别表示红、绿和蓝。这些数字是用十六进制表示的，它以16为基数，而不像十进制（以10为基数，系统用的）。图13-5显示了十六进制RGB值的结构。

十六进制的RGB值必须以#符号开始

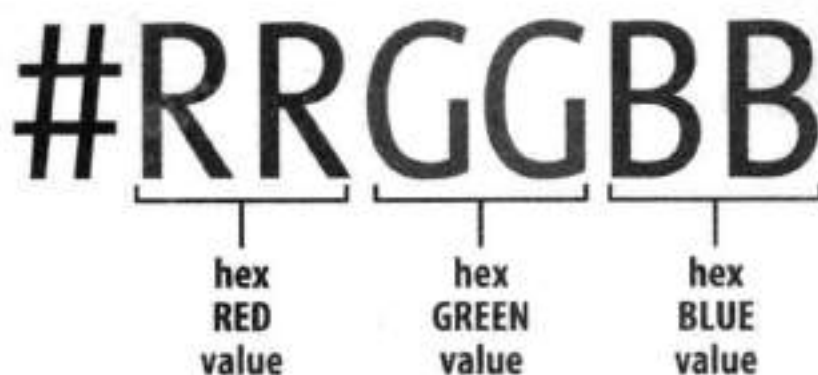


图13-5：十六进制数由三个两位数组成，分别表示红、绿和蓝

十六进制数字系统使用16个数：0~9和A~F（代表10~15的数）。图13-6显示了对应关系。由于减少了存储某些信息的容量，十六进制在计算机中应用很广。例如，转换成十六进制的时候，RGB值从三位数减少到了两位数。

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

<div>00</div> <div>sixteens place ones place</div>		<p>The decimal number 32 is represented as</p> <div>20</div> <p>2 sixteens and 0 ones</p>	<p>The decimal number 42 is represented as</p> <div>2A</div> <p>2 sixteens and 10 ones</p>
---	--	--	---

图13-6：十六进制数字系统以16为基数

既然大部分图像和网页开发软件提供简单的十六进制颜色值的访问方法（如图13-4所示），所以没必要像以前一样，手动将RGB值转换为十六进制。但是，如果你发现有这个必要，可以参考侧栏“十六进制计算”。

RGBA 颜色

RGBA颜色不仅允许定义一个颜色而且可以使颜色按照你的喜好变成透明或不透明。在RGBA中的a代表alpha。Alpha是一种额外的控制透明度的（从0-完全透明到1-完全不透明）通道。下面展示了它的具体写法：

```
color: rgba(0, 0, 0, .5);
```

在括号中前三个创造出的黑色的值是常规的RGB值。第四个值.5是透明度。所以这样表示的颜色是透明度为50%的黑色。这将允许别的颜色和背景模式轻微显示出来（如图13-7所示）。

Playing with RGBa

h1 {color: rgba(0, 0, 0, .1);}

Playing with RGBa

h1 {color: rgba(0, 0, 0, .5);}

Playing with RGBa

h1 {color: rgba(0, 0, 0, 1);}

图13-7: 使用RGBa值的不同透明度的标题

然而这里还有一个问题，就是IE。IE 8和更早的浏览器不支持RGBa颜色，所以你需要为使用这些浏览器的用户提供一个后备方法。最早的浏览器只是选择一个完全透明的颜色，这种颜色近似于样式规则中首先列出来的样子。IE将忽略RGBa值，而当它们在第二位声明时，其他浏览器将会覆盖不透明的颜色。

```
h1 {
  color: rgb(120, 120, 120);
  color: rgba(0, 0, 0, 50%);
}
```

但是对于IE 6~IE 8来说，如果你想要在IE中做出透明度，你可以提供其他选择（例如，一个透明的PNG或者一个IE特有的过滤器）。具体方法是在只有IE能理解的条件语句中写出规则或者元素（参考侧栏“针对IE使用条件注释”）。幸运的是，RGBa被IE 9或者更高的版本支持，所以随着老式的版本逐渐停止使用，我们不需要给自己加上额外的束缚。

注意： HSL色彩也可以通过HSLa颜色来提供一个透明度，与RGBa的语法一样：

```
color: hsla(0, 0%, 0%, .5);
```

针对IE使用条件注释

IE的条件注释语法为IE或者某些特殊版本的IE提供了指定样式的方式。而其他浏览器会忽略这些条件注释，但是IE会应用所指定的样式。条件注释可以在样式表中，或者像下面的例子一样，使用style元素来提供一个单独的嵌入式样式表。确保在正规的样式表后添加条件注释。

举个例子，可以使用RGBa作为备选，条件注释可以在“版本为8或者之前版本”（if lte IE 8），对p元素的背景应用50%透明度的PNG。（透明PNG在第21章中讨论）。

```
<!--[if lte IE 8]>
<style>
  p {background: transparent url(black-50.png);}
</style>
```

```
<![endif]-->
```

在IE 6~IE 8中创建透明性的另一种方式是使用IE过滤规则，这个方式比较精巧，所以我推荐Eric Ferraiuolo的文章“RGBA——IE Fallback”（925html.com/code/rgba-ie-fallback/）。如果想更详细地了解条件注释，可以直接去微软的开发者网络站点详细了解（[msdn.microsoft.com/en-us/library/ms537512\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms537512(v=vs.85).aspx)）。

你可能知道，在网络开发者社区，条件注释的使用是有争议的。一些开发者尽可能避免使用它，而用JavaScript方案来替代。而另一些开发者却根据情况来使用，并不在意它们并不是严格有效的标记。希望有一天，老版本的IE消退在历史的尘埃里，那这个技术也就不再需要了。

颜色值总结

虽然我们花了好几页的篇幅才讲到这里，但是你会发现，其实在样式表中拾取和指定颜色很容易。

- 从预定义的颜色名中选取

或者

- 使用颜色表或图像编辑器来选中一个颜色，复制下RGB值（最好是6位十六进制数）。使用四种RGB值格式中任意一种，把值加入样式表，这样就完成了。或者如果需要，你也可以使用HSL。

事实上，有更有趣的方式去填充一个元素，那就是使用渐变（颜色从一个色调消失到另一个色调），但是这有点像打开潘多拉魔盒，所以我不会在这儿讲述，在本章的最后将会讲到CSS渐变。

前景色

既然已经知道了如何写颜色值，那么就可以开始学习颜色相关的属性了。可以给任何HTML元素指定前景色和背景色。`border-color`属性也采用颜色值，但这方面的内容会在第14章中讨论。

元素的前景由文本和边框（如果已经指定）组成。使用`color`属性指定前景色，就像第12章见到的那样，文本会变得漂亮。下面再次给出`color`属性的细节：

`color`

属性值： 颜色值（颜色名或数字） | `inherit`

默认值： 依赖浏览器和用户习惯

适用对象： 所有元素

是否可继承： 是

在下一个例子中，`blockquote`元素的前景为一种漂亮的绿色，颜色值为R:80、G:140和B:25（用十六进制表示为#508C19）。`blockquote`元素使用了`color`属性，它所包含的

和`em`元素也继承了这个颜色（图13-8）。`blockquote`块的粗虚线也是绿色的。但是，如果应用`border-color`属性到这个元素，`border-color`的颜色就会覆盖绿色前景的设定。

样式规则

```
blockquote {  
  border: 4px dashed;  
  color: #508C19;  
}
```

标记文本

```
<blockquote>
  <p>I'd recommend Honey Gold cereal to anyone who likes cereal. It's
  the <em>only</em> way to start the day!.</p>
  <p>&mdash; Jennifer Robbins, happy consumer</p>
</blockquote>
```

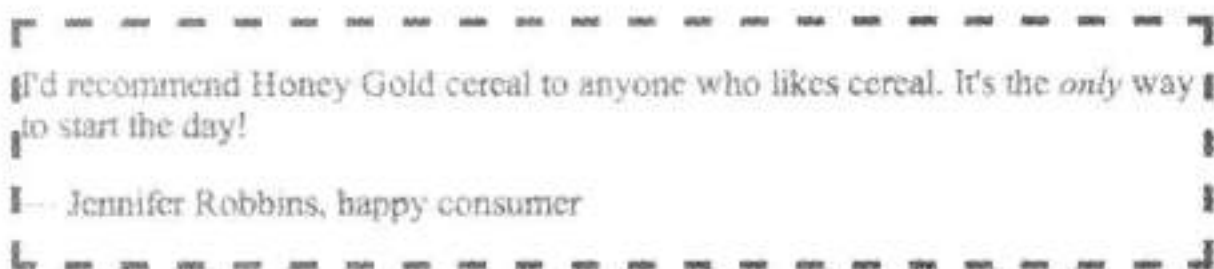


图13-8：应用颜色属性到元素前景

背景色

使用background-color属性就可以把背景色应用到任何元素。

background-color

属性值： 颜色值（颜色名或数字）|transparent|inherit
 默认值： transparent
 适用对象： 所有元素
 是否可继承： 否

背景色填充元素后面的背景，包括内容区域、内容周围的任意padding区域（扩展空间），并且一直扩展到边框的外边缘。当使用background-color属性把这个例子中blockquote块的背景设置为淡蓝色时，看看效果（如图13-9所示）。

```
blockquote {
  border: 4px dashed;
  color: #508C19;
  background-color: #B4DBE6;
}
```

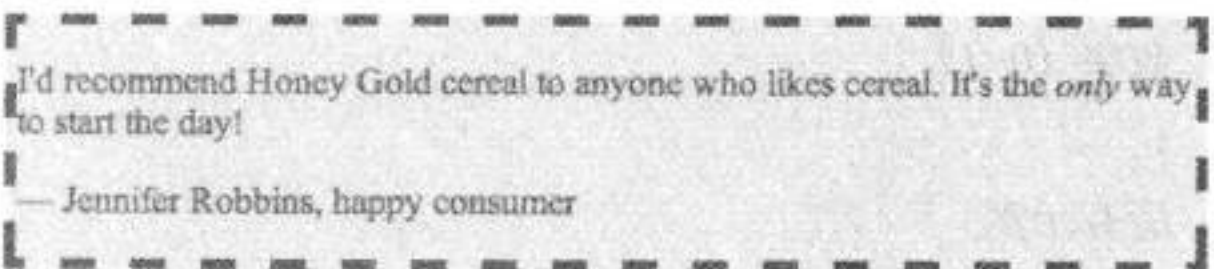


图13-9：给例子中blockquote块添加淡蓝色背景

正如所料，背景色填充了从文本一直到边框后面的区域。仔细看边框中的间隙处，发现背景的确延伸到外边缘，但是背景也只到那里。如果给这个元素应用margin属性，背景也不会延伸到margin区域。当谈论CSS盒

网页调色板

在阅读网页设计的书或者使用网页制作工具（如Dreamweaver或Photoshop）时，可能会遇见网页调色板。网页安全色很容易认出，它们只由十六进制值00、33、66、99、CC和FF组成。

网页调色板是216种颜色的集，它用在只能在显示256种颜色的低端显示器的浏览器中显示颜色。这216种颜色由Windows和Macintosh操作系统颜色的交叉部分组成。

回到大部分用户使用低端显示器的时代，网页设计师坚持使用网页安全色，因为它们显示得流畅而又稳定。然而，因为现在只有少于1%的用户使用256色显示器，所以浏览器几乎不需要把网页中所有的颜色映射到调色板中。这意味着不再需要限制颜色的选择——科技的进步已经让调色板成为老古董了。

设计提示

使用颜色

下面是一些与操作颜色有关的简明提示：

- 在一个网页中使用有限个元素。没有什么比使用太多颜色更容易导致视觉混乱了。我往往是选择一个主要颜色和明亮颜色。我可能也会使用每个颜色的一对稍暗色，但我反对使用太多不同的色调。
- 在指定前景色和背景色的时候，确保有适当的对比度。人们倾向于在网上阅读浅色背景深色文字的文档。在图13-9的例子中，虽然完成了任务，但是事实上，这个例子在对比度测试中失败了。
- 一前一后的设置前景色和背景色（特别对整页而言）是一个好主意。如果用户在样式表中有一个或其他的设置，这样做可以避免颜色冲突和对比度问题。

要设定整个网页的背景，可以在body元素中使用background-color属性。

子模型的时候，还会再认识这些元素。现在只需要知道：如果边框有间隙，背景会穿过显示。

背景色继承是没有价值的，因为所有元素的默认背景色设置都是transparent（透明），父元素的背景色会穿过子元素而显示。例如，可以通过在body元素上应用background-color属性，来改变整个网页的背景色。这个颜色会穿过网页中所有的元素而显示。

除了设定整个网页的背景，你也能改变任何元素的背景，包括块级别的元素（就像上个例子中的blockquote）和内嵌元素。在本例中，用color属性和background-color属性来突出显示标记为“glossary”的词。图13-10中可以看到背景色填充了内联dfn元素创建的小盒子。

样式规则

```
.glossary {  
    color: #7C3306;           /* dark brown */  
    background-color: #F2F288; /* light yellow */  
}
```

标记文本

```
<p>A <dfn class="glossary">baseline</dfn> is the imaginary line upon  
which characters sit.</p>
```

A baseline is the imaginary line upon which characters sit.

图13-10：应用背景色到内联元素

使用不透明度

之前，讨论了RGBa颜色，当被应用于颜色或者背景的时候，它可以添加透明度。CSS3中的opacity属性是同RGBa的效果一样的另一种方式，它可以使一种元素轻微透明。

opacity

属性值： 数字 (0~1)
默认值： 1
适用对象： 所有元素
是否可继承： 否

opacity的值介于0（完全透明）和1（完全不透明）之间。例如0.5表示给了一个元素50%的透明度。不透明的设置应用于整个元素——前景色和背景色。如果你只是想影响一个或者另一个，那么应该使用RGBa颜色值。

在下面列举的例子中（图13-11），一个标题已经被设定为绿色，同时背景色设置为白色。当不透明属性设定后，允许这页的背景透过文本和元素显示。

```
h1 {color: green; background: white; opacity: .25;}
h1 {color: green; background: white; opacity: .5 ;}
h1 {color: green; background: white; opacity: 1;}
```

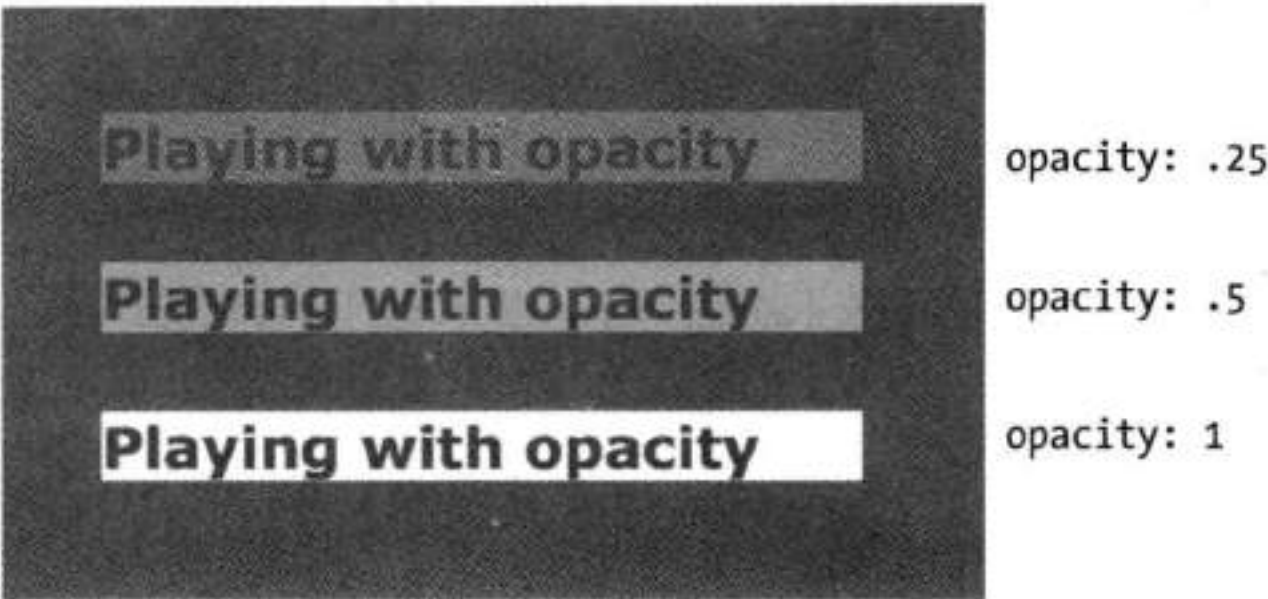


图13-11：对一个元素设置opacity会影响前景和背景色

注意，这种方法在IE 8以及之前的版本是行不通的（你可能看到过）。要在这样的浏览器下设置透明度，需要使用IE限定的过滤器，就像下面的例子中一样。第一个声明用于IE 8，第二个用于IE 7和IE 6。zoom属性保证IE可以识别布局中的元素。理想的情况是，这些规则应该通过条件注释在IE特定的样式表中使用。

```
h1 {
  filter:alpha(opacity=50);
  -ms-filter:"progid:DXImageTransform.Microsoft.Alpha(opacity=50)";
  zoom: 1;
}
```

你可能渴望使用这些颜色和背景属性，但是首先我想介绍火爆的CSS选择器。侧栏列出了你已经熟悉的一些选择器。

伪类选择器简介

你是否注意到，链接在你单击时是一种颜色，你返回这个网页时又是另一种颜色？之所以会如此，那是因为，在后台，浏览器一直在记录你所单击过的链接（术语中也叫做“访问过”）。浏览器也会记录其他链接状态。比如，用户的鼠标是否在链接上，或者是否正在单击。

在CSS中，只需通过使用叫做pseudo-class的特殊选择器，就可以给每一种状态的链接应用样式。这是个奇怪的名称，但是你可以想象它与同一

快速浏览

下面先总结下之前学过的选择器方面的内容：

元素类型选择器

```
p {property: value;}
```

组合选择器

```
p, h1, h2 {property: value;}
```

后代（关联）选择器

```
ol li {property: value;}
```

ID选择器

```
#sidebar {property: value;}
div#sidebar {property: value;}
```

类选择器

```
p.warning {property: value;}
.warning {property: value;}
```

通用选择器

```
*{property: value;}
```

在附录B中提供了CSS3选择器完整的列表。

警告： 当你改变链接和访问过的链接的外观时，要确保他们看起来还像链接。

类的某种状态相关联。但是，类名不在标记中——只是被浏览器记录。它有点像类，所以说它是伪类。

伪类选择器用冒号(:)标示。他们紧跟在元素名称后，例如，`li:first-child`。

在CSS3中有一些伪类选择器，在本章中，我将介绍最常用的以及支持最好的伪类选择器。全部的CSS3选择器，以及相关的描述和例子，见附录B。

链接伪类

最基本的伪类选择器着眼于链接（a元素）上，基于他们是否已被单击。链接伪类是一种动态伪类，因为它们是作为与用户交互的结果，而不仅是一些标记。

<code>:link</code>	对没有被单击的链接应用样式
<code>:visited</code>	对已经被单击的链接应用样式

默认情况下，浏览器通常会将链接显示为蓝色，而将单击过的链接显示为紫色，但你可以通过几个样式规则来改变这种方式。

在这些例子中，我已经将链接的颜色变为栗色，并且设置访问过的链接为灰色。访问过的链接一般比没有访问过的链接看起来更柔和。

```
a:link {
    color: maroon;
}
a:visited {
    color: gray;
}
```

用户操作的伪类

另一种类型的动态伪类目标为用户操作所造成的元素状态。

<code>:focus</code>	应用于元素被选择并且准备输入时
<code>:hover</code>	应用于鼠标指针悬停在元素上时
<code>:active</code>	应用于当元素正在单击或触动的处理过程中

聚焦状态

如果你曾经使用过Web表单，那么你应该熟悉浏览器如何强调显示一个选中的表单元素。当一个元素突出显示，并准备输入时，被认为“聚焦”。`:focus`选择器使元素在获得焦点时，应用自定义样式。

警告： 在IE 6中不支持focus伪类。

在这个例子中，当用户选择了一个文本输入时，它得到了黄色背景色，使其区别于其他形式的输入。

```
input:focus { background-color: yellow; }
```

悬停状态

`:hover`选择器十分有趣。当用户的鼠标悬停在元素上时，`:hover`就会作用于该元素。虽然你可以对任何元素使用悬停状态，但它最常用于链接，它可以改变链接的外观，使用户在视觉上感知可以对链接进行一些操作。

当鼠标悬停在链接之上时，下面的规则提供了一个浅粉红色的背景颜色。

```
a:hover {  
  color: maroon;  
  background-color: #ffd9d9;  
}
```

重要的是要意识到，在触摸屏设备如智能手机和平板电脑上是没有悬停状态的，所以这一块的反馈体验将丢失（见“注意”）。这使得为链接提供与鼠标操作无关的视觉标示非常重要。

活跃状态

最后，`:active`选择器适用于被激活过程中的元素。对于链接来说，当链接被单击或者在触摸屏上通过指尖触动时，就会应用该选择器的样式。这种样式可以只显示一个瞬间，但它可以给一个巧妙的指示，表示要发生一些事情。在下面的例子中，我已经为活动状态加亮了颜色（从栗色为红色）。

```
a:active {  
  color: red;  
  background-color: #ffd9d9;  
}
```

把它们放在一起

Web设计师通常会提供所有链接状态的样式，因为这是一个简单的方法，可以为单击链接过程中的每种状态都提供一些反馈提示。事实上，用户也期待此类反馈提示：他们可以一目了然所触动的链接，当你指向这些链接时，链接会有一些提示；当成功地单击链接时，会收到确认信息。

当你将样式应用到含有五个伪类的元素时，它们的出现顺序对于正常运行是很重要的。例如，如果你把链接`:link`或者`:visit`放到最

注意：虽然无法在一个元素上悬停，但是在iOS的Safari和一些安卓设备中，可以通过单击来展示`:hover`状态样式。要打开链接，用户必须再次单击。这个方法使得在触摸设备上，依然可以使用CSS驱动的下拉菜单。但是这样可能会与其他悬停状态的对象混淆。因此一些开发者为触摸设备创建了一个不包含`:hover`的单独的样式表。

后，那么将覆盖其他状态并阻止它们的出现。链接伪类所需的顺序是：`:link`、`:visit`、`:hover`、`:active`（你可以简记为LVFHA）。

建议你为使用键盘而不是鼠标来触动链接的用户提供`:focus`样式。为`:hover`应用相同的样式也是常见的，虽然这不是必需的。

下面总结一下，已经展示的链接样式看起来应该是样式表中所示那样。图13-12显示了结果。

```
a { text-decoration: none; } /* 为所有链接关闭下划线*/
a:link { color: maroon; }
a:visited { color: gray; }
a:focus { color: maroon; background-color: #ffd9d9; }
a:hover { color: maroon; background-color: #ffd9d9; }
a:active { color: red; background-color: #ffd9d9; }
```

Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:link`
Links are maroon and not underlined.

Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:focus`
`a:hover`
While the mouse is over the link or when the link has focus, the pink background color appears.

Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:active`
As the mouse button is being pressed, the link turns bright red.

Samples of my work:

- Pen and Ink Illustrations
- Paintings
- Collage

`a:visited`
After that page has been visited, the link is gray.

图13-12：使用伪类选择器改变链接的颜色和背景

更多伪类选择器

除了动态伪类，CSS3选择器模块包含基于浏览器的状态的其他类型的选择器。

需要注意的是，IE 8或更早的版本都不支持这些选择器。为了使用JavaScript来支持，可以试着使用Selectivizr (selectivizr.com)，这是为了支持早期的IE浏览器而添加到文件中的脚本。Selectivizr是polyfill的一个例子，也是一个使老版本的浏览器能支持当今的Web功能的脚本（polyfill将在第20章中讨论）。

结构化伪类

这可以使伪类基于元素在文档结构（文档树）中的位置。它们的名字非常直观地说明了它们的作用。

<code>:root</code>	<code>:only-child</code>	<code>:nth-child()</code>
<code>:empty</code>	<code>:first-of-type</code>	<code>:nth-last-child()</code>
<code>:first-child</code>	<code>:last-of-type</code>	<code>:nth-of-type()</code>
<code>:last-child</code>	<code>:only-of-type</code>	<code>:nth-last-of-type()</code>

其他的伪类选择器

现在5个 CSS3伪类已经讲完，还剩下18个！无论你是谁，这都听起来有点单调乏味。我希望你知道还有哪些选择，所以我把它放到侧栏“更多伪类选择器”中，我们可以看看其他一些选择器类型。此外，你可以在附录B中找到说明和例子。当你准备好选择更复杂的选择器时，你可以将附录作为参考。

伪元素选择器

伪类不是唯一的一种伪选择器，还有4个用于在文档结构中插入虚构元素来完成样式的伪元素。在CSS3

中，伪元素由一个双冒号 (::) 表示，但为了更好地向后兼容，在 CSS2 中使用单冒号 (:) 来定义。

首字母和首行

下面这两个伪元素用于选择浏览器中元素里文本的首字母或首行。

:first-line

这个选择器在特定元素的首行应用样式规则。但是，能应用的属性只有：

color	font	background
word-spacing	letter-spacing	text-decoration
vertical-align	text-transform	line-height

:first-letter

这个选择器在特定元素的首字母应用样式规则。能应用的属性只有：

color	font	text-decoration
text-transform	vertical-align	padding
background	margin	line-height
border	float	
letter-spacing	word-spacing	

注意：列表中有一些没见过的属性。我们将会在第14章中涉及这些盒子相关的属性（margin、padding和border）。将会在第15章中介绍float属性。

图13-13 显示了:first-line和:first-letter伪选择器的例子。

```
p:first-line {letter-spacing: 8px;}
```

Snow White was banished for being most beautiful, fell in with seven dwarves, ate a poison apple, and fell asleep in a glass coffin until the handsome prince kissed her, married her, and they lived happily ever after.

```
p:first-letter {font-size: 300%; color: orange;}
```

Snow White was banished for being most beautiful, fell in with seven dwarves, ate a poison apple, and fell asleep in a glass coffin until the handsome prince kissed her, married her, and they lived happily ever after.

图13-13: first-line 和 :first-letter伪选择器的例子

用:before和:after生成内容

CSS2引入了:before和:after伪元素，用于在特定元素之前或之后插入内容，而不需要在源文档中真实地添加字符（这个在CSS中叫做生成内容）。生成内容这个功能可以用于在引号中插入有关语言的引用，插入自动计数器，或者文档打印时在链接附近显示URL。

UI（用户界面）选择器

这些选择器应用于表单窗体小部件。

:enabled :disabled :checked

还有更多

其他的伪类还有：

:target（选择URL的片段标识符指向的元素）
:lang()（可以基于两种语言编码进行选择）
:not()（除了圆括号中所列的元素，可以选择其他所有元素）

下面是一个简单的例子，在段前插入一些单词“Once upon a time:”，在段后插入“The End.”，如图13-14所示。

样式表：

```
p:before {
  content: "Once upon a time: ";
  font-weight: bold;
  color: purple;
}
p:after {
  content: " The End.";
  font-weight: bold;
  color: purple;
}
```

标记文本：

```
<p>Snow White was banished for being the most beautiful, ... and they
lived happily ever after. </p>
```

Once upon a time: Snow White was banished for being the most beautiful, fell in with seven dwarves, ate a poison apple, and fell asleep in a glass coffin until the handsome prince kissed her, married her, and they lived happily ever after. The End.

图13-14：在Firefox浏览器（Macintosh操作系统）用:before和:after伪选择器生成内容

生成内容在IE 6和IE 7中无法使用，但在其他浏览器中没有问题。在你的第一个项目中，没必要使用这个功能。但是，如果你有兴趣学习更多，我推荐Smashing杂志的教程：coding.smashingmagazine.com/2011/07/13/learning-to-use-the-before-and-after-pseudo-elements-in-css/。如果你想对这个技术刨根问底，可以阅读 W3C Generated and Replaced Content Module (www.w3.org/TR/css3-content/)。

属性选择器

马上就要进入选择器的最后一部分了。最后一种选择器是与属性相关的选择器。你可以匹配属性的名字或者属性的值，它使你无须添加很多class或者id标记，而能灵活地选取元素。

下面是CSS2引入的属性选择器，除了IE 6，其他浏览器都可以很好地支持这些选择器。

元素[属性]

简单属性选择器选择有特定属性的元素，而不管它的值。下面的例子中，选择所有带title属性的图像。

```
img[title] {border: 3px solid;}
```

元素[属性=“精确值”]

精确属性值选择器选择某个属性值为特定值的元素。在IE 7中，值是区分大小写的，因此必须输入正确，否则无法识别。下面的例子中，选择title值精确等于“first grade”的图像。

```
img[title="first grade"] {border: 3px solid;}
```

元素[属性~=“值”]

部分属性值选择器允许指定属性值的一部分。下面的例子中，在title属性里寻找词“grade”，所以title属性值为“first grade”和“second grade”的图像都会被选中。

```
img[title~="grade"] {border: 3px solid;}
```

元素[属性|=“值”]

连字符分隔属性选择器标记用连字符分开的值。这个选择器会匹配所有指向英文变种文档的链接，其中属性值可能为en-us（美式英文）、en-in（印度英文）和en-au-tas（澳大利亚英文）等。（注意*是通配选择器，可以选中“任意元素”。）

```
*[hreflang|=“en”] {border: 3px solid;}
```

接下来的内容出现在新的CSS3属性选择器中，所以它们也是刚刚起步。IE 6或IE 7无法支持这些新特性，老版本的Safari Opera和Firefox也只能部分支持。

元素[属性^=“值的第一部分”]

起点子串属性值选择器匹配属性值以子串开始的元素。下面的例子中，这个选择器对目录/images/icons下所有的图像都应用样式（）。

```
img[src^="/images/icons"] {border: 3px solid;}
```

元素[属性\$=“值的最后一部分”]

结尾子串属性值选择器匹配属性值以子串结束的元素。在下面的例子中，你可以对指向PDF文件的链接应用样式。

```
a[href$=".pdf"] {border: 3px solid;}
```

元素[属性*=“值的任意一部分”]

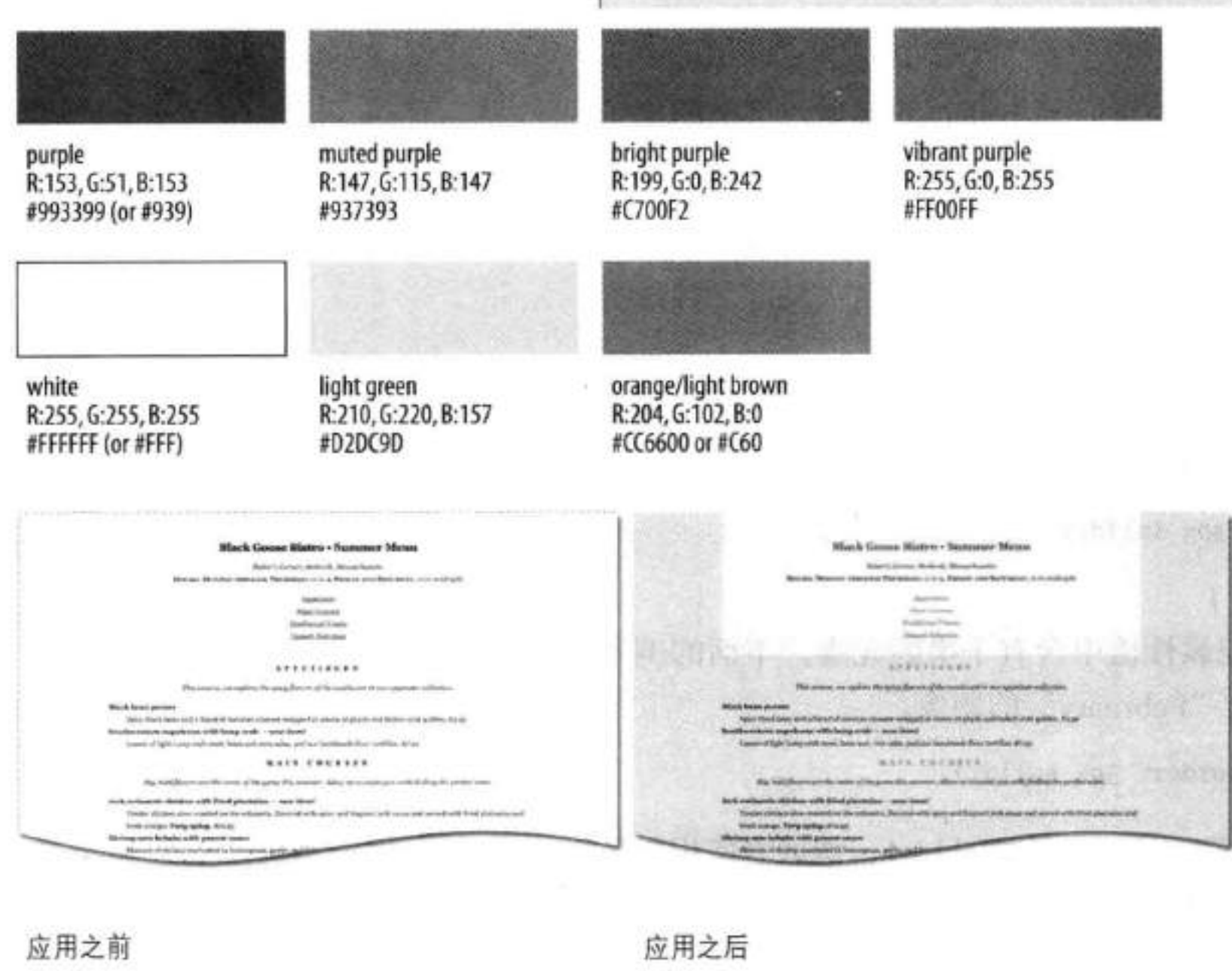
任意子串属性值选择器匹配属性值中含有子串的元素。下面的例子中，会选中所有标题中包含“February”的图像。

```
img[title*="February"] {border: 3px solid;}
```

好了，现在已经讲完了所有选择器。接下来是练习13-1，请试着使用前景和背景色，还有一些新的选择器类型。

警告：不要忘记在十六进制值前加#字符。否则这条规则将失效。

图13-15：应用色彩之后的Black Goose Bistro菜单页面



练习13-1 给文档添加颜色

本练习中，我们将以一个简单的黑白菜单开始，通过设置前景和背景提供一些个性（图13-15）。现在你应该有足够的经验来编写样式规则了，所以我不再像前面的练习一样，手把手教你了。你可以对照附录A中提供的样式表进行检查。


在文本编辑器中打开文件**bistro.html**（可以在www.learningwebdesign.com/4e/materials获得）。你会发现已经有一个提供基本文本格式化的内嵌样式表，它甚至包含第14章将要讨论的margin和padding属性。利用设置好的文本，你只需要操作颜色。你随时可以保存文本，并在浏览器中查看。

1. 通过为h1已有的规则添加声明，设置h1标题为紫色（R:153，G:51，B:153，或者#993399）。注意，因为所有的值数字成双，所以可以（也应该）使用简写（#939），从而节省了样式表中的字节。
2. 设置h2标题为黄褐色（R:204，G:102，B:0，或#cc6600）。
3. 设置整页背景为浅绿色（R: 210，G:220，B:157，或#d2dc9d）。现在可以保存一下，在浏览器中查看，如果标题和背景颜色没有显示，就先找找问题。
4. 将“头部”div的背景设置为白色，且赋予50%的透明度（R:255，G:255，B:255，.5），这样背景色就会透过来。
5. 我已经添加了一条规则，关闭链接下方的下划线（text-decoration:none），所以需要使用颜色来凸显链接。编写一个规则，使得链接与h1一样都为紫色（#993399）。

6. 设置访问过的链接为淡紫色（#937393）。
7. 当鼠标放在链接上时，使文本呈亮紫色（#c700f2），并且添加一个白色背景（#fff）。当鼠标指向链接的时候，链接看起来就像被点亮一样。当链接获取焦点时，也使用相同的样式规则。
8. 当鼠标单击时（或者在触摸设备上触动后），设置链接文本为鲜亮的紫色（#ff00ff），并且添加白色背景。确保链接伪类顺序正确。

完成以后，网页就会看起来像图13-15中的一样。之后将给网页添加背景图像，所以如果你想试着继续给不同元素设置不同的颜色，就先对文档进行复制，然后重命名。

背景图像

我们已经知道如何使用元素来给文档添加图像，但是现在，很多装饰图片都作为背景，使用CSS技术来添加到页面和元素。毕竟，装饰性图片，如瓷砖式的背景样式只是外观的一部分。这也允许设计者通过编辑.css文件来改变站点的外观。现在看来，过去依靠巨幅的图片裁剪和堆叠表格来做网站的日子真是不堪回首。

本节中将介绍用来放置和摆弄背景图像的属性集合，从基本的background-image属性开始。

添加背景图像

background-image属性用于给元素添加背景图像。它的主要工作是提供图像文件的位置。

background-image

属性值： URL (图像的位置) | none | inherit
 默认值： none
 适用对象： 所有元素
 是否可继承： 否

background-image属性值是一种包含图像位置的URL容器（见“注意”）。

URL与CSS规则的位置相关。如果规则在嵌入式样式表中（HTML文档的style元素），那么URL的路径名就与HTML文件的位置相关。如果CSS规则在外部的样式表中，那么路径就与.css文件的位置相关。可以看看另一种方法的提示。

下面的例子和图13-16显示了应用于整个网页（body）的背景图像和应用过padding和border属性的单个blockquote元素。

```
body {
  background-image: url(star.gif);
}

blockquote {
  background-image: url(dot.gif);
  padding: 2em;
  border: 4px dashed;
}
```

快速浏览

背景图像相关属性包括：

background-image
 background-repeat
 background-position
 background-attachment
 background-clip (CSS3)
 background-size (CSS3)
 background

注意：从严格意义上说，“URL容器”只是一个功能符号。十进制数和百分比的RGB值有相同的语法。

资源

Standardista站点有非常详细的浏览器所支持的、与背景属性相关的图表。非常值得阅读：
www.standardista.com/css3/css3-background-properties/。

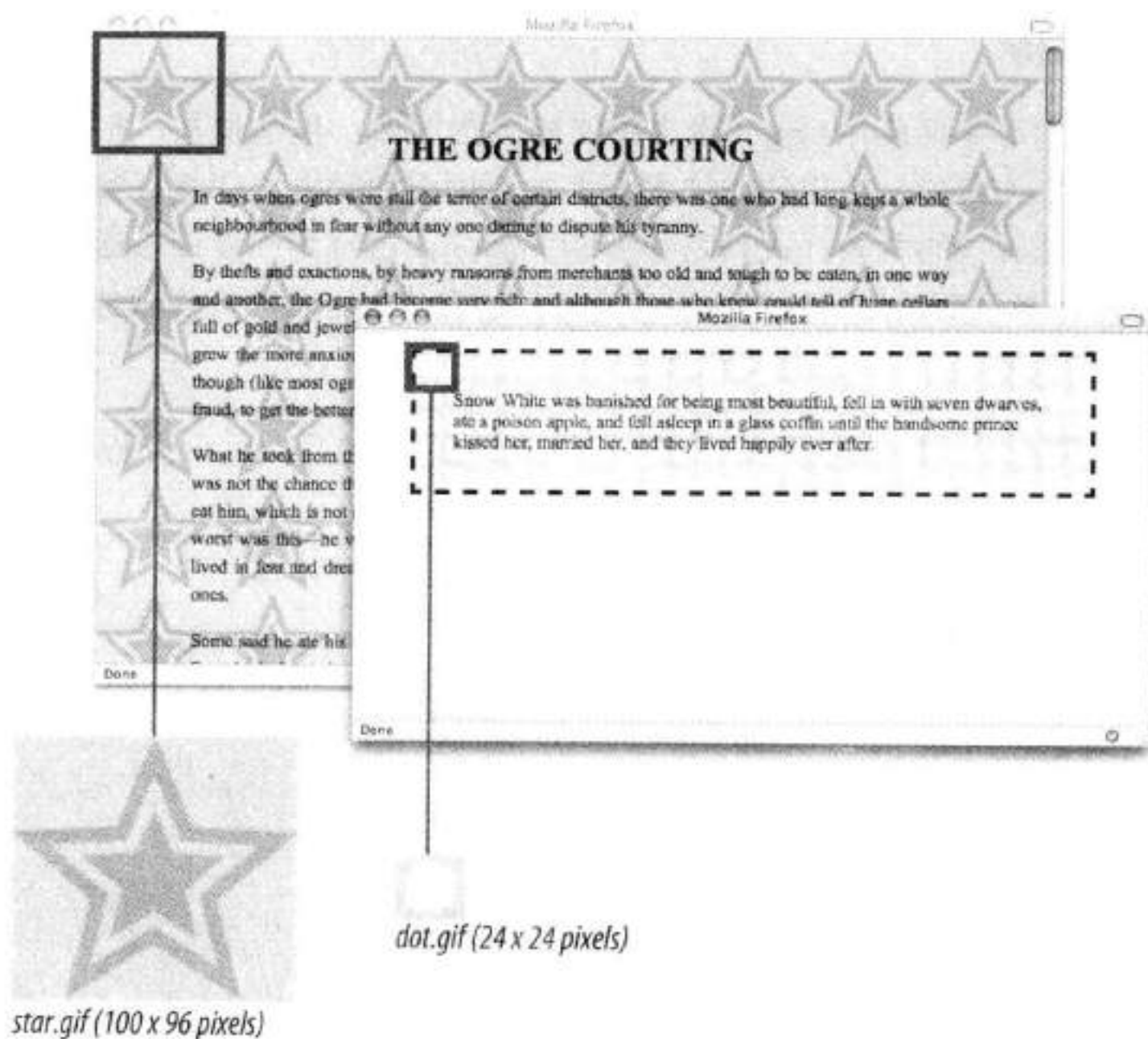


图13-16：使用background-image属性添加拼贴背景图像的例子

在这里可以看到background-image属性的默认行为，图像起始于左上角，并沿着水平和垂直方向拼贴，直到整个元素被填满（待会儿你将学习如何改变它）。就像背景颜色、背景图像填充内容区域以及内容周围扩展的padding区域，并延伸到边框的外边缘（如果存在）。

如果给元素同时提供background-color和background-image属性，那么，图像就贴在了颜色上面。事实上，我建议提供一个与背景色类似色调的备份颜色，在图像下载失败时使用。

设计提示

拼贴背景图像

当使用背景图像时，记住这些指导方针和提示：

- 使用一个简单的图像，不要干扰到文本的清晰度。
- 提供一个与背景主色调相配的background-color属性值。如果图像不能显示，至少网页的整体设计会相似。如果文本颜色在浏览器默认的白色背景下看不清楚，这一点就尤其重要。
- 对于网页，通常保持背景图像的文件大小尽可能小。

练习13-2 添加拼贴背景图像

在练习中，我们将给菜单添加拼贴图像背景。本练习所用的图像可以在`images`目录找到。

给`body`样式规则添加一个声明，使得图像`bullseye.png`在页面的背景上拼贴。确保包含样式表的路径名（在这个练习中，也就是当前这个HTML文档的路径名）。

```
background-image: url(images/bullseye.png);
```

很简单不是吗？当你保存并在浏览器中打开这个网页时，它看起来将如图13-17所示。

我想指出，`bullseye.png`是一个略微透明的PNG图像，所以它会与背景色有些混合。你将在第21章学习如何制作透明PNG。暂时先通过添加第二个`background-color`声明，它就会通过覆盖来改变`body`元素之前的`background-color`属性。可以试着使用不同的颜色，留意有什么变化。当你完成了试验，删除第二个声明，使背景依然为绿色，以便进行后面的练习。



图13-17：带有简单拼贴背景图像的文章

控制拼贴方向

就像在图13-17看到的，图像是从上到下、从左到右拼贴的。你可以用`background-repeat`属性来改变这种行为。

background-repeat

属性值： `repeat` | `repeat-x` | `repeat-y` | `no-repeat` | `inherit`

默认值： `repeat`

适用对象： 所有元素

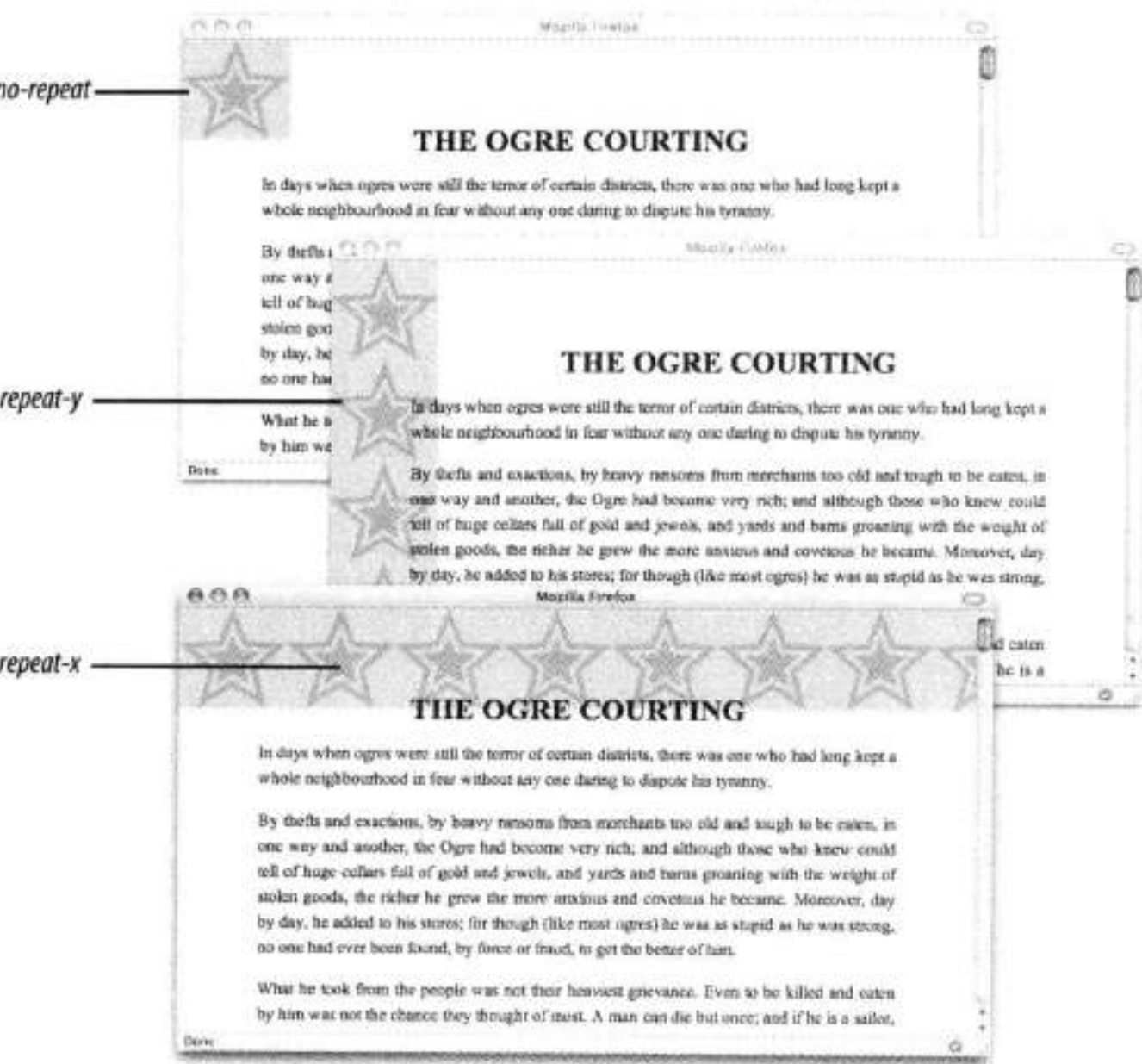
是否可继承： 否

如果你想让背景图像仅出现一次，那么使用`no-repeat`值，如下所示：

```
body {
  background-image: url(star.gif);
  background-repeat: no-repeat;
}
```

也可以像下面的例子一样，限制图像只能沿水平方向（`repeat-x`）或垂直方向（`repeat-y`）拼贴。

```
body {
  background-image: url(star.gif);
  background-repeat: repeat-x;
}
body {
  background-image: url(star.gif);
```

```
background-repeat: repeat-y;
}
```

图13-8展示了这些值的例子。注意在这些例子中，拼贴都是从元素（如果图片应用在body元素上，就从浏览器窗口）左上角开始。下面将告诉你如何改变。

图13-18：用no-repeat(上)关闭自动拼贴，用repeat-y (中)设置沿垂直轴拼贴，用repeat-x (下)设置沿水平轴拼贴

练习13-3 控制拼贴方向

现在，在文章网页的例子上一试更精妙的拼贴方式。这次添加拼贴图像，且只沿着“header”div的顶部边缘来拼贴。

- 1. 在#header规则中，添加图像purpledot.gif，并且设置它为水平拼贴。

```
#header {
  margin-top: 0;
  padding: 3em 1em 2em 1em;
  text-align: center;
  background-color: rgba(255,255,255,.5);
  background-image: url(images/purpledot.png);
  background-repeat: repeat-x;
}
```

- 2. 保存文件，并在浏览器中查看。它应该如图13-19所示。我建议调整浏览器窗口的大小，注意背景

样式的位置随着窗口的变宽或变窄有什么变化。为什么它始终固定在左边？接下来会学习如何调整位置。

- 3. 试着改变样式规则，使purpledot.gif沿垂直拼贴，然后再设置为不拼贴。完成后，再设置回repeat-x。



图13-19：给#header div元素添加水平拼贴图像

背景位置

`background-position`属性指定背景中原图 (origin image) 的位置。你可把原图想象成背景中放置的第一个图，从原图开始延伸拼贴。下面是这个属性和它的可变属性值。

`background-position`

属性值:	长度计量值 百分比值 left center right top bottom inherit
默认值:	0% 0% (同left top)
适用对象:	所有元素
是否可继承:	否

通常，应该提供水平和垂直两个方向的值来描述原图的位置，但是还有很多种不同的方法。

关键字定位法

键值 (left、right、top、bottom和center) 定义了原图相对于元素边缘的位置。例如，left位置表示原图在背景区域的左边缘。默认的原图位置相当于“left、top”

关键字是成对出现的，如下：

```
background-position: left bottom;
background-position: right center;
```

如果只提供一个关键字，遗失的关键字会假定为center。所以，`background-position: right`与`background-position: right center`有相同的效果。

长度计量法

也可以指定到元素左上角的距离的位置，使用像素作为单位。当提供长度值时，水平计量通常放在前面。

```
background-position: 200px 50px;
```

百分比值法

百分比值以水平/垂直成对的方式提供，0% 0%对应左上角，100% 100%对应右下角。注意百分比值应用于画布区域和图像。例如，100%值把图像的右下角放到可用区域的右下角。使用这些关键字，如果你只提供一个百分比值，另一个会假定为50%（居中）。

```
background-position: 15% 100%;
```

图13-20显示上面列出的每一个`background-position`例子的效果，它们清楚地设置`background-repeat`属性值为no-repeat。可以定位原图，然

CSS提示

为确保在现代浏览器中有最好的表现，请给所有的值类型都首先提供水平方向的计量。

后从这里开始拼贴，在两个方向，或者只从水平方向或只从垂直方向拼贴。当图像拼贴时，原始图像的位置不明显，但是可以用background-position属性从不是图像左边缘的一个点开始拼贴图案。

注意：在图13-20中，注意当原图置于一个元素的角落时，它会放置在边框内部。只有重复的拼贴图案会越过边框，延伸到边框的外边缘。

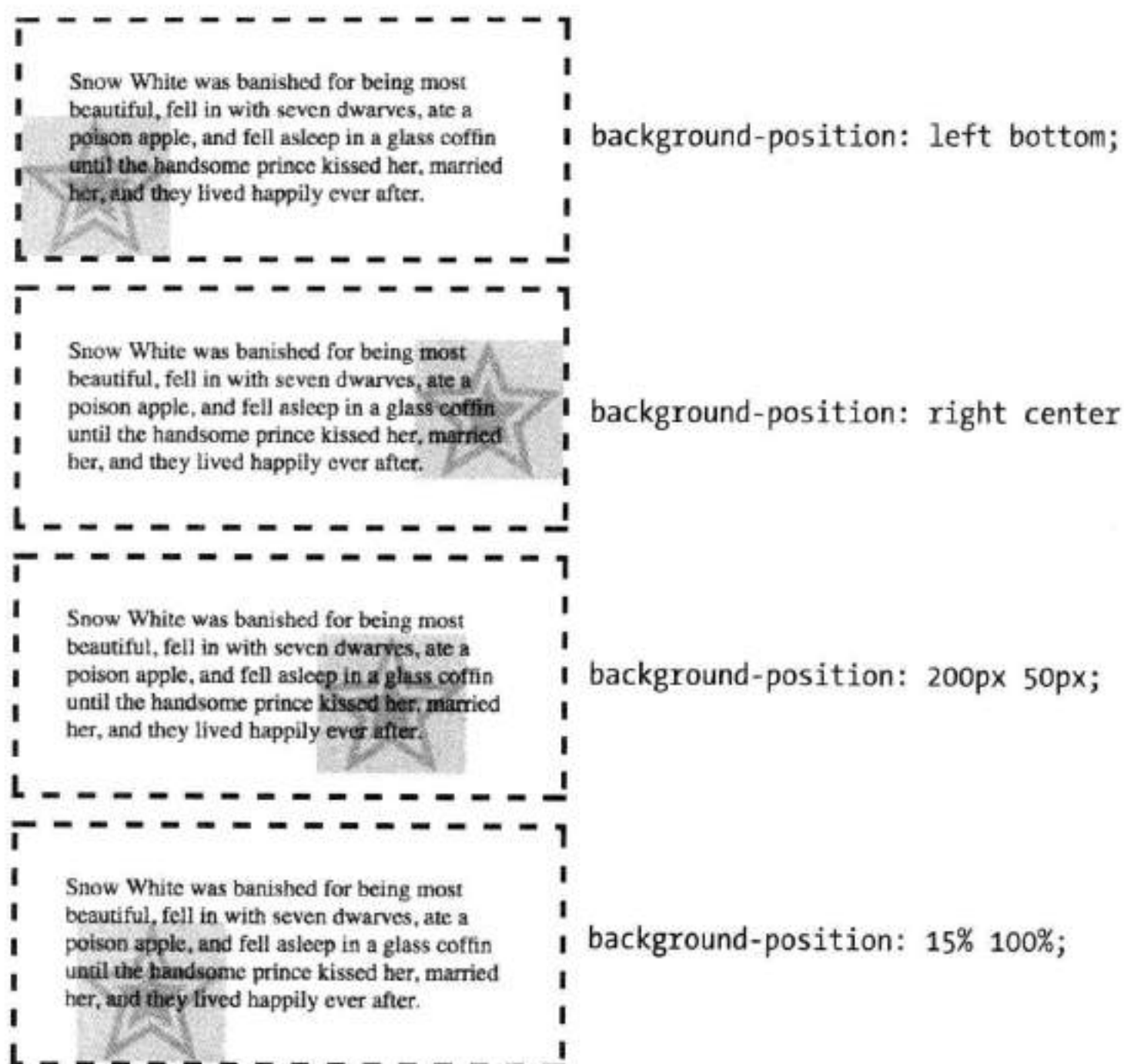


图13-20：放置无重复的背景图像

练习13-4 放置背景图像

现在对菜单中的背景图像位置进行一些有趣的尝试。首先，对已有的背景图像进行一些微妙的调整，然后换用完全不同的背景，并进行更多尝试。依然使用**bistro.html**文档，这个文档的body和#header元素中有拼贴样式。

1. 既然菜单的主元素在中央，如果背景样式也放在中央，应该会美观一些。给body和#header规则添加这个声明，保存文件并在浏览器中查看。调整浏览器窗口的宽窄来查看效果，否则你看不出分别。现在样式已经固定在中央，或多或少也延伸到了边缘外，而不是像以前一样只到边缘。

```
background-position: center top;
```

2. 接下来，改变background-position值，以使紫色点图沿着标题的底部边缘div (center bottom) 拼贴。(由于看着不好看，我又放回了原处。) 然后试着把bullseye.png下移200像素 (center 200px)。注意样式仍然填满整个屏幕——把原图下移，但是背景依然沿所有方向拼贴。图13-21显示了这些改变的结果。
3. 现在看起来不错，但是现在需要放弃body的背景。我想教你一些技巧。随着设计的进行，我倾向于隐藏注释中的样式，而不是完全删除它们。这样，我就没必要记住这些样式，或者重新输入，我只需要移除注释提示符即可。当完成设计后，该发布时，我会去掉无用的样式，以减少文件的尺寸。下面展示了如何隐藏注释中的声明：

```
body {
```

```
font-family: Georgia, serif;
font-size: 100%;
line-height: 175%;
margin: 0 15%;
background-color: #d2dc9d;
/* background-image: url(images/bullseye.png);
background-position: center 200px; */
}
```

4. 现在，给页面的背景添加blackgoose.png图像 (也是一个半透明的PNG图像)。设置为非重复，并把它放在页面中央。

```
background-image: url(images/blackgoose.png);
background-repeat: no-repeat;
background-position: center top;
```

在浏览器中看看，滚动页面时，注意观察背景的滚动。

5. 我希望你能了解多位置关键字和数字值。试试下面的规则，并在浏览器中查看。在滚动页面时，观察会发生什么。注意，当你为垂直位置提供一个百分比值或者关键字时，它参照的是整个文档的高度，而不是浏览器窗口的高度。你也可以随意试试。

```
background-position: right top;
background-position: right bottom;
background-position: left 50%;
background-position: center 100px;
```

6. 将图像放在center 100px的位置，为下一个练习做好准备。你的页面现在看起来应该如图13-21中的右图。

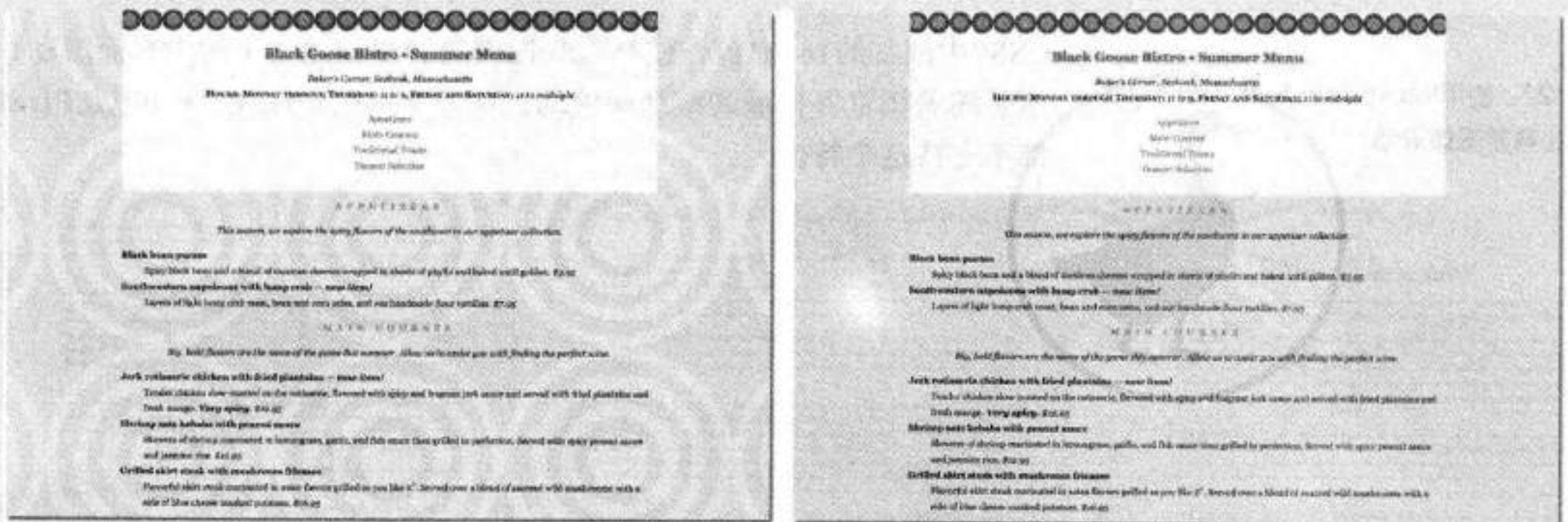


图13-21：将原图放在拼贴背景样式中（左）和放置单独的背景logo（右）

背景附加方式

在练习13-4中，我要求你们滚动网页，去看背景图像发生的变化。正如所料，它随着文档滚动，并且消失在浏览器的顶部，这是默认的行为。然后，你可以用background-attachment属性将背景从内容中解脱出来，使得当其他内容滚动时，背景固定在一个位置。

background-attachment

- 属性值: scroll|fixed|local|inherit
- 默认值: scroll
- 适用对象: 所有元素
- 是否可继承: 否

使用background-attachment属性时，可以选择背景图像是滚动的还是固定的。当图像是固定时，它留在浏览器视域的同一个位置（相对于它所在的元素，正好相反）。你马上你会知道我讲的是什么意思。

注意：你可以固定任何元素的背景图像的位置，但是遗憾的是，对于IE 6和IE 7用户是无效的。这也是当做“冷冻”使用的另一个特征。

在下一个例子中，一个大的、非拼贴的图像将放置在整个文档（body元素）的背景上。默认情况下，当文档滚动时，图像也滚动，在网页中上下移动，如图13-22所示。然而，如果你设置background-attachment属性值为fixed，那么，图像就待在它开始所在的地方，而文本却在它上面滚动。

```
body {
  background-image: url(images/bigstar.gif);
  background-repeat: no-repeat;
  background-position: center 300px;
  background-attachment: fixed;
}
```

图13-22：使用background-attachment属性防止背景图像滚动

CSS3中添加的local值，使得背景图像随着滚动元素中的内容而滚动，而不是随着浏览器视域的滚动而滚动。在写本书时，IE 6~IE 8以及Firefox都不支持这个特性。



文档的body中有一个大的、不重复的背景图片

background-attachment: scroll;

默认情况下，背景图片附加在body元素中，并且在页面内容滚动时，图像也滚动

background-attachment: fixed;

当background-attachment被设置为“fixed”时，图像不随着内容滚动，而是保持在浏览器可视区域的相关位置

练习13-5 固定位置

在练习13-4中，我们给页面的背景添加了一个大的、非重复的logo图像。这次依然要达到这样的效果，但是我们会使用background-attachment属性让它留在相同的位置上，即便滚动页面，它的位置也不变。

```
body{
  ...
  background-attachment:
  fixed;
}
```

保存文档，在浏览器中打开，然后滚动。背景图像在浏览器的视域中位置不变，很酷吧？

如果你有兴趣，可以看看当在div#header中固定点图时会发生什么。（提示：它仍在相同的位置上，但只是在div中。div滑出视窗时，它的背景也随之滑出。）

CSS3背景属性

CSS3背景和边框模块引入了更多的属性来控制背景。这个模块还是草案，所以这里的信息可能会改变。这在IE 6~IE 8中无法支持。

background-clip

值： border-box|padding-box|content-box

这个属性可以精确指定背景图像可以延伸多远。默认情况下，它可以延伸到边框的边缘（border-box），但是你需要分别使用padding-box或者content-box属性来在盒子边缘停止图像的延伸。在第14章将讨论盒子模块组件。

background-size

值： [length|percentage|auto]|cover|contain

这个属性允许设计者控制属性中的背景图像的尺寸。你可以提供特定的宽度和高度。如果你只提供一个值，另一个就假定为auto。你也可以将图像设置为contain，这会使图像调整尺寸以适应包含它的元素，即使还留下一些空白；或者可以设置为cover，这会使图像调整尺寸，以便覆盖整个元素，即使背景图像覆盖了边缘，超出了视域。

background-orig

值： border-box|padding-box|content-box

这个属性决定了background-position如何计算，或者换句话说，从哪里开始计算位置。你可以从边框、填充或内容的边缘开始。

快捷背景属性

你可以使用便捷的background属性来在一个声明里指定所有的背景样式。

background

属性值: *background-color background-image background-repeat background-attachment background-position|inherit*

默认值: 看各个独立属性

适用对象: 所有元素

是否可继承: 否

就像快捷font属性, background属性的值是上面列出的所有独立背景属性的值的列表。例如, 下面这个背景属性规则:

```
body { background: white url(arlo.png) no-repeat right top fixed; }
```

可以取代下面五个独立声明的规则:

```
body {
  background-color: white;
  background-image: url(arlo.png);
  background-repeat: no-repeat;
  background-position: right top;
  background-attachment: fixed;
}
```

background的所有属性值都是可选的, 可以以任何顺序出现。唯一的限制是, 当用坐标值表示background-position属性时, 横坐标要放在前面, 紧跟着的是纵坐标。记住, 如果一个值被忽略了, 它会被设置为默认值 (见“小心覆盖”)

小心覆盖

background属性很有效, 但是使用时要小心。之前给它编过址, 但它支持重复。因为它是一个快捷属性, 当忽略一个值时, 属性会被重置为默认值。所以要小心, 不要意外地让后面的快捷样式属性覆盖了前面的样式规则, 这会让你的设置都恢复为默认值。

在这这里的例子中, 背景图像dots.gif不能应用于h3元素, 因为如果忽略background-image的值, 它的值实际上会成为none。

```
h1, h2, h3 { background: red url(dots.gif) repeat-x; }
h3 {background: green;}
```

要重写特殊的属性, 可以使用你想改变的、特定的background属性。比如, 如果在上面的例子中想改变h3元素的背景色, 最好选择background-color属性。

练习13-6 使用快捷属性

这个练习很简单。使用单独的background属性替换bistro菜单的body中所有背景相关的声明。

```
body {
  font-family: Georgia,
  serif;
  font-size: 100%;
  line-height: 175%;
  margin: 0 15%;
  background: #d2dc9d
  url(images/blackgoose.
  png) no-repeat center 100px
  fixed;
}
```

对div做同样的事, 这个由你来做。

多背景

目前对一个元素只能应用一个背景图像。在过去要实现多背景，唯一的解决办法是添加额外的div标记，并为每一个标记添加一个图像。值得庆幸的是，CSS3允许多个背景图像应用到单一的元素，并且浏览器也都开始支持这个特性。

为background-image应用多个值，需要把它们列成以逗号分隔的表。其他背景相关的属性值也在逗号分隔的列表中；列出的第一个值，适用于所述第一图像，第二个值适用于第二图像，以此类推。第一值所定义的图像，将出现在前面，其他图像则按照顺序排在后面。

```
body {
  background-image: url(image1.png), url(image2.png), url(image3.png);
  background-position: left top, center center, right bottom;
  background-repeat: no-repeat; no-repeat; no-repeat;
  ...
}
```

或者，你可以利用background快捷属性使规则更简单。现在的background属性有三组值，都以逗号分隔：

```
body {
  background:
    url(image1.png) left top no-repeat,
    url(image2.png) center center no-repeat,
    url(image3.png) right bottom no-repeat;
  ...
}
```

图13-23显示了结果。

对于新的CSS3技术，IE 6~IE 8还不支持，它们会忽略有一个以上值的background声明。解决方案是选择一个background-image作为IE或其他不能支持的浏览器的后备属性，然后指定多个background规则。由于多个background规则列在后面，所以支持这个特性的浏览器会用它来替代单独图像的规则。对于所有的背景图像，提供一个background-color是不错的主意。把它放在最后，这样快捷背景属性就不会覆盖它。

当浏览器可以普遍支持时，你可以使用多重背景来“锦上添花”。

注意：虽然CSS声明往往是“最后一条有效”，对于多背景图像而言，列在后面的图像显示在底部，而前面的图像则在上层，以此类推。你可以把它们当做Photoshop的层，它们按列表中的顺序来堆叠显示。列表前面的图像显示在上层。

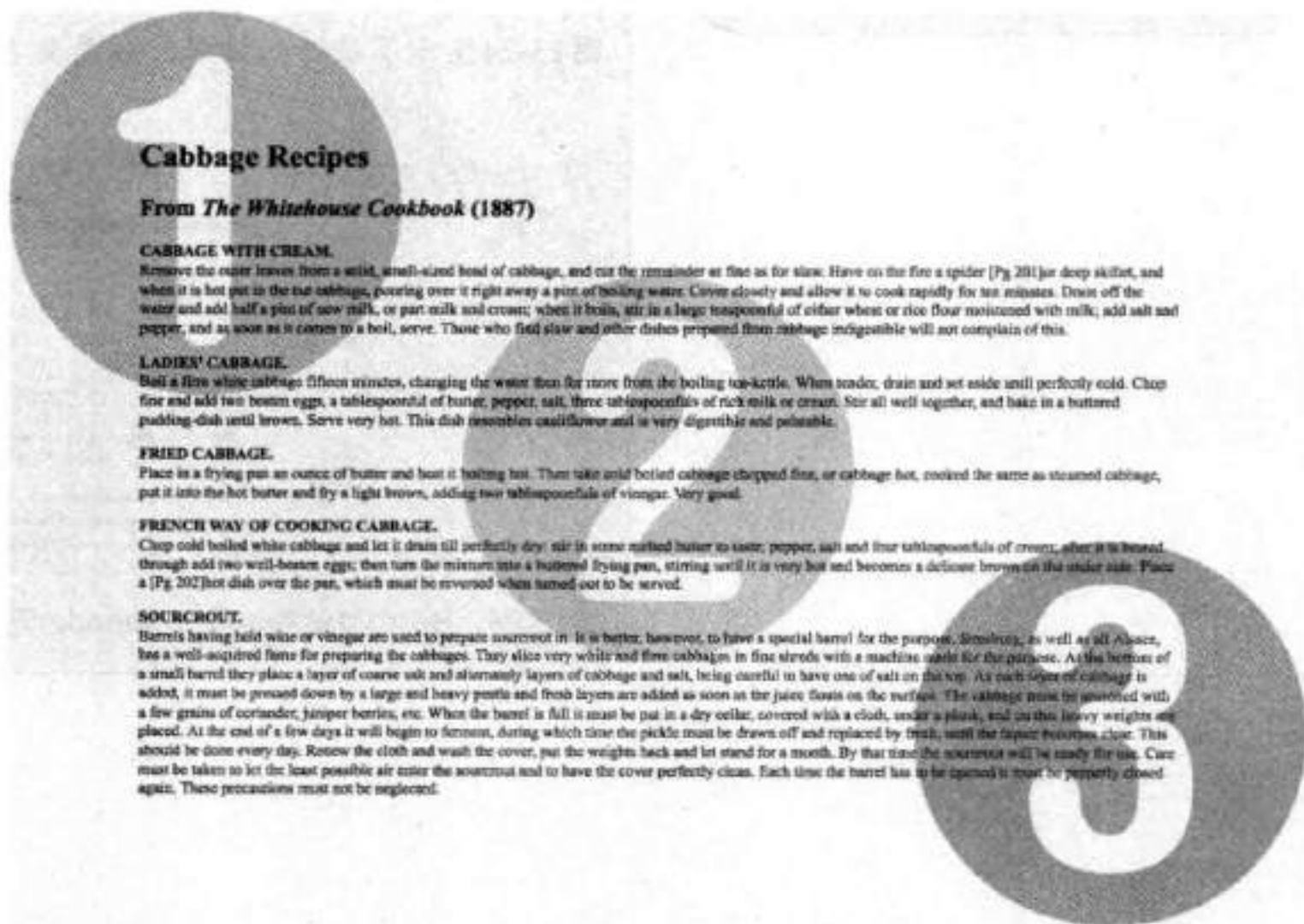


图13-23：三个不同添加到body元素的背景图像

多背景图像的滚动视差

运动视差是指相互靠近的对象在视觉上看起来比相离较远的对象移动得快。重复动画中的近场、中场、远场的速度可以有3D的效果。

一些设计者使用多背景图像来创建滚动视差效果。当你调整浏览器窗口尺寸，或者移动水平的滚动条时，背景的错列移动会创建一种视差和3D效果。由于你不能水平地调整或滚动移动浏览器，所以在手机和平板电脑上没有这样的效果。

要学习这种效果的比较好的起点是Chris Coyier写的学习教程“Starry Night: Incredible 3D Background Effect with Parallax”，(css-tricks.com/3d-parallax-background-effect/)”。也可以看看Paul Annett的“How to Recreate Silverback’s Parallax Effect”。(thinkvitamin.com/design/how-to-recreate-silverbacks-parallax-effect/)，这个文章提到了Silverback的app页面(silverbackapp.com)。

```
body {
  background: url(image_fallback.png) top left no-repeat;
  /* for non-supporting browsers */
  background:
    url(image1.png) left top no-repeat,
    url(image2.png) center center no-repeat,
    url(image3.png) right bottom no-repeat;
  background-color: papayawhip; /* background color */
}
```

练习13-7 多背景图像

在这个练习中，我们会试着使用多个背景图像。注意，如果你使用IE 6~IE 8，你就看不到多图像，所以要使用Chrome、Safari或者Firefox（这些浏览器都是免费的）。

我想让#header div的圆点图沿着左右边缘来拼贴。我还有一个鹅的图像([gooseshadow.png](#))，如果它能漫步在标题的底部，看起来一定很酷。完成下面的练习，我已经编写了后备的background-image规则（我们已经使用过水平方向的圆点图），并且用背景色来结束。

```
#header {
  ...
  background-image: url(images/purpledots.png) center top repeat-x;
  background:
    url(images/purpledots.png) left top repeat-y,
    url(images/purpledots.png) right top repeat-y,
    url(images/gooseshadow.png) 90% bottom no-repeat;
  background-color: rgba(255,255,255,.5);
}
```

图13-24显示了最终的结果。我是挺喜欢的，你感觉如何呢？



图13-24: Bistro菜单的头部div#header元素中有两列圆点图，还有一个鹅的图像

像彩虹一样（渐变）

渐变是指从一种颜色过渡到另一种、或者多种颜色。在过去，只有使用CSS创建一个图像编辑程序，并添加生成的图像，才能在一个Web页面上设置渐变。

CSS3引入了单靠CSS符号来指定颜色渐变的功能，而把渲染颜色融合的任务留给浏览器。渐变适用于图像的任何应用：`background-image`、`border-image`和`list-style-image`。在本章将使用`background-image`。

有两种类型的渐变：

- 线性渐变：沿着一条线改变颜色，从该元素的一个边缘到另一个边缘。
- 径向渐变：在一个点开始，呈向外扩散的圆形或椭圆形。

渐变是浏览器在移动时产生的图像。像背景图像一样使用它吧。

线性渐变

`linear-gradient()`符号提供了线的角度，以及沿着纯色所在线的一个或多个点的起止色位置（color stop）。你可以使用在本章前面讨论的颜色的名称或颜色值的数值。线的角度使用度（`ndeg`）或使用关键字来指定。如果使用度，0度表示向上，正角度沿着顺时针方向，因此90度刚好指向右边。因此，你想要从顶部边缘的黄色，到达底部边缘的绿色，需要设置旋转角度为180度。

```
#banner
  background-image: linear-gradient(180deg, yellow, green);
}
```

关键词描述方向，每个方向都相差90度（`to top`、`to right`、`to bottom`和`to left`）。因此，旋转180度的渐变也能使用关键词`to bottom`表示。结果显示在图 13-25（最上图）中。

```
#banner {
  background-image: linear-gradient(to bottom, yellow, green);
}
```

在下面的例子中，渐变从左到右，并包括第三种颜色——橙色，橙色以25%的方式横跨渐变线（图13-25的中图）出现。你可以看到color stop显示在颜色值之后。起止色位置值如果为0%和100%，则可以省略。

```
#banner {
  background-image: linear-gradient (90deg, yellow, orange 25%, blue);
}
```

这些例子都相当显眼，但如果你正确地选择颜色和起止色位置（color

stop)，渐变是一个很好的方式来给元素提供细微的阴影和3-D外观。底部的按钮使用背景渐变实现了3D效果，而无须使用图形（图13-25的下图）。

```
a.button-like {
  background: linear-gradient(to bottom, #e2e2e2 0%, #dbdbdb 50%,
    #d1d1d1 51%, #fefefe 100%);
}
```



```
linear-gradient(to bottom, yellow, green);
```



```
linear-gradient(90deg, yellow, orange 25%, blue);
```



```
linear-gradient(to bottom, #e2e2e2 0%, #dbdbdb 50%, #d1d1d1 51%, #fefefe 100%);
```

图13-25：线性渐变的例子

径向渐变

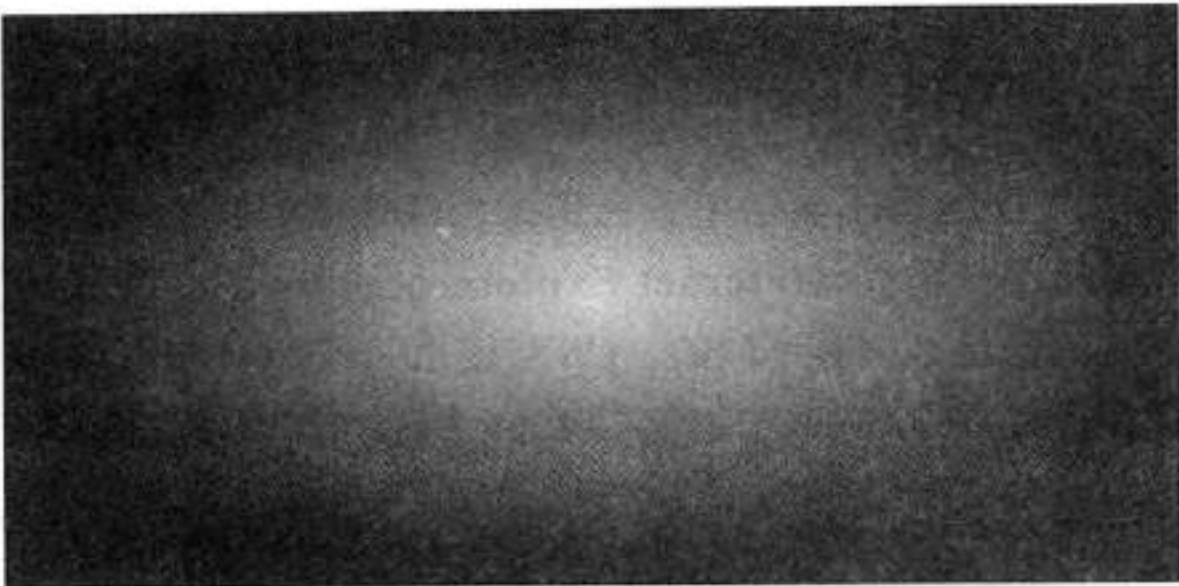
径向渐变，正如这个名字，意思是从一个圈中的一个点开始渐变。在CSS3规范中，`radial-gradient()`符号描述了形状（圆形（`circle`）或椭圆形（`ellipse`），如果没有指定形状，默认情况下是`circle`），渐变中心的位置（与`background-position`有相同的语法）和尺寸，可以用半径长度来精确指定，或使用一个关键字来描述颜色应停止的边角（`closest-side`、`farthest-side`、`closest-corner`、`farthest-corner`、`cover`和`contain`）。

下面的规则使用径向渐变填充元素框（图13-26）。

```
#banner {
```

注意：想要了解更多的径向渐变信息，可以看看John Alsopp写的这篇比较透彻的文章，“CSS3 Radial Gradients”（www.webdirections.org/blog/css3-radial-gradients/）。

```
background-image: radial-gradient(center contain yellow green);
}
```



```
radial-gradient(center, yellow, green);
```

图13-26：径向渐变的例子

引入供应商的前缀

这里，事情变得有趣的地方，其实也并不是那么有趣。CSS渐变的例子中，我们已经了解了CSS3规范中的语法描述。但是在规范完全确定之前，浏览器开始主动融合渐变技术。对于像渐变这样的先进技术，浏览器往往用自己的解决方法进行实验，并且在规范完全确认下来之前执行。

这是很多浏览器制造商在20世纪90年代做的事，这造成许多不兼容的灾害，但是这一次，它们可以用供应商前缀来标注其专有的特性，这种前缀清楚地标明其具有专有解决方案。图13-1列出了目前正在使用的浏览器前缀。

表13-1：浏览器供应商前缀

前缀	组织	最流行的浏览器
-ms-	Microsoft	Internet Explorer
-moz-	Mozilla基金会	Firefox、Camino、SeaMonkey
-o-	Opera Software	Opera、Opera Mini、Opera Mobile
-webkit-	最初由Apple负责开发，现已开源	Safari、Chrome、Android、Silk、BlackBerry、WebOS等
-khtml-	Konqueror	Konqueror

进一步阅读

为了更好地解释浏览器前缀，我非常推荐我的好伙伴Eric Meyer的文章“Prefix or Posthack” (www.alistapart.com/articles/prefix-or-posthack/)。

对于所有浏览器前缀属性，其中一些将永远不会进入标准，可以看看Peter Beverloo编辑的绚丽的图表：peter.sh/experiments/vendor-prefixed-css-property-overview。

注意：想了解更多关于渐变在不同浏览器的语法信息，以及渐变相对于图像的优势，可以阅读Chris Coyier的“Speed Up with CSS3 Gradients” (css-tricks.com/css3-gradients/)。

这对于设计人员和开发人员意味着，一些较新的CSS3的功能，我们需要使用浏览器前缀为每个浏览器列出属性，以适应不同的实现。虽然这意味着额外的工作和额外的编码，但其实并不是一件坏事。它可以让浏览器厂商以不干扰的标准过程的方式实现创新。

还是继续讨论渐变，下面的例子显示了黄色到绿色线性渐变，它可以用来解决所有现代浏览器的问题（与Internet Explorer过滤器相当，都是一种好办法）。注意它的语法是略有不同的。CSS3使用“to bottom”关键字，其他大部分使用“top”，Webkit使用“left top和left bottom”。

```
background: #ffff00; /* Old browsers */
background: -moz-linear-gradient(top, #ffff00 0%, #00ff00 100%);
/* FF3.6+ */
background: -webkit-gradient(linear, left top, left bottom, color-stop(0%,#ffff00), color-stop(100%,#00ff00));
/* Chrome,Safari4+ */
background: -webkit-linear-gradient(top, #ffff00 0%,#00ff00 100%);
/* Chrome10+,Safari5.1+ */
background: -o-linear-gradient(top, #ffff00 0%,#00ff00 100%);
/* Opera 11.10+ */
background: -ms-linear-gradient(top, #ffff00 0%,#00ff00 100%);
/* IE10+ */
background: linear-gradient(to bottom, #ffff00 0%,#00ff00 100%);
/* W3C */
filter: progid:DXImageTransform.Microsoft.gradient(
  startColorstr=' #ffff00' , endColorstr=' #00ff00' ,GradientType=0 );
/* IE6-9 */
```

好消息是，随着新的兼容的浏览器出现和旧版本离我们远去，一些特性，如文字阴影，不再要求使用浏览器前缀了。可能当你读本书的时候，浏览器的前缀可能已经是一种老套的办法了（我希望是这样的）。但更可能的是，了解供应商前缀以及它们适用的属性仍然会是很有用的。

在接下来的章节中，每当一个属性要求供应商前缀，我会一定会做提示。否则，你可以认为只是标准的CSS。

设计渐变

最后的代码示例非常绝妙！不用说供应商前缀，单是描述渐变的任务就很艰巨。这不可能靠手工来编写代码，我建议你像我一样使用在线的渐变工具！我最喜欢的是来自ColorZilla的The Ultimate CSS Gradient Generator “ (www.colorzilla.com/gradient-editor/)”，如图13-27所示。只需输入你想输入的多种起止色位置，滑动滑块直到得到你想要的，然后复制代码。这正是在刚刚讲过的例子里做的事。

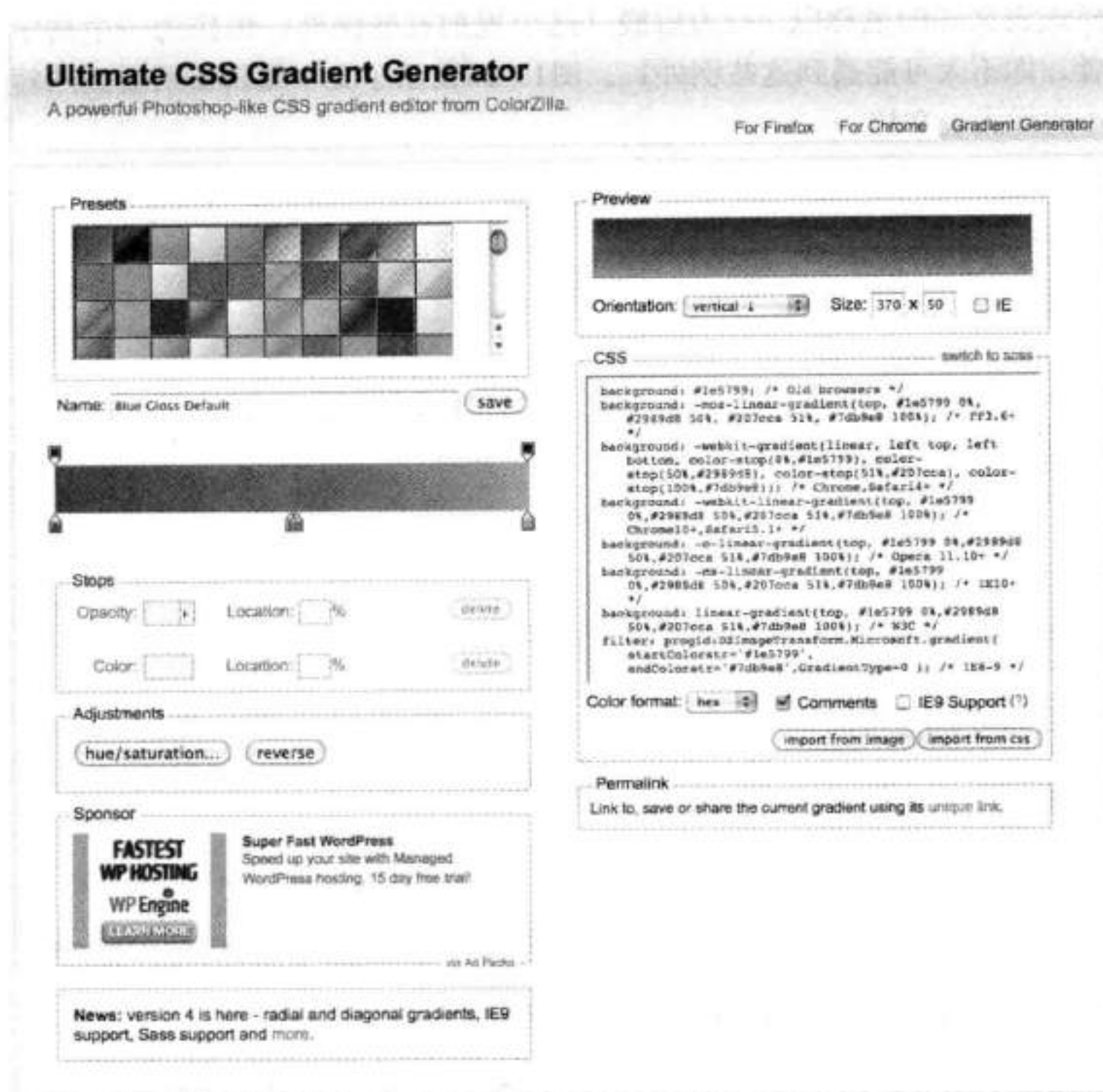


图13-27: CSS渐变生成器 (The Ultimate CSS Gradient Generator (www.colorzilla.com/gradient-editor)), 它使创建CSS渐变轻而易举

最后，外部样式表

回到第11章，已经讲了三种将样式表关联到HTML文档中的方法：使用style属性内联，嵌入style元素，和作为一个外部.css文件链接或导入文档中。本节来讲第三个方式。

外部样式表是CSS至今最强大的方法，因为可以通过编辑一个样式表来改变整个网站的样式。这就是把所有样式信息放在一个地方，而不是把样式信息与源文档混在一起的好处。

首先，需要知道一些关于样式表文档本身的信息。外部样式表是一个纯文本文档，它至少含有一条样式规则。它不能包括任何HTML标签（没有为什么，就是不可以）。它可以包含注释，但它只能使用你已经见过的CSS注释语法：

```
/* This is the end of the section */
```


样式表文件的名称以.css为后缀（这个规则有些例外，但作为一个初学者，你不太可能遇到这些例外）。图13-28显示了文本编辑器所看到的短小的样式表文档。

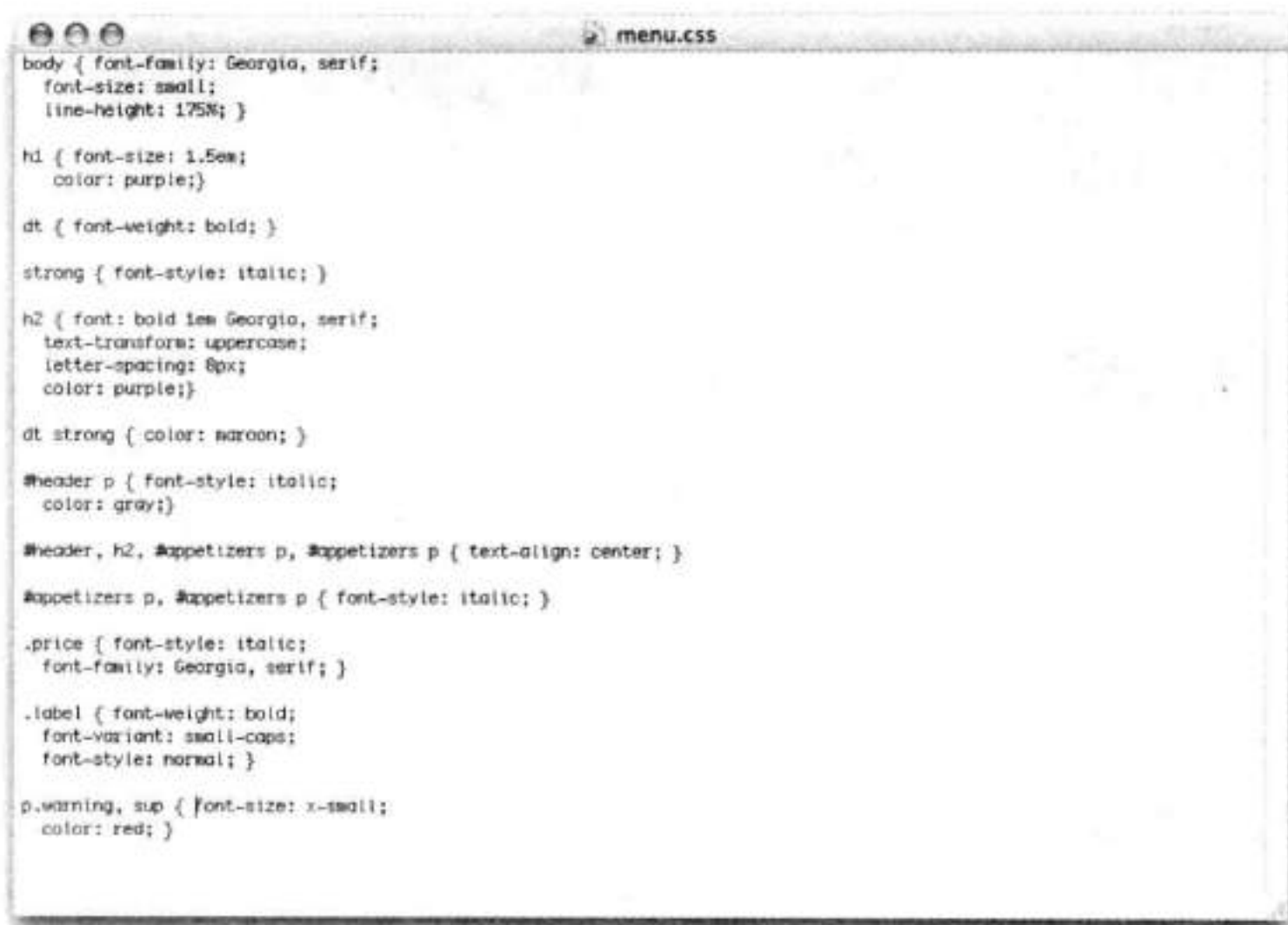


图13-28：外部样式表仅包含纯文本的CSS规则和注释

有两种引用外部样式表的方法：通过链接（link）元素和@import规则。下面介绍这两种附加方法。

使用链接元素

最受支持的方法是在文档head部分，使用link元素创建一个指向.css文档的链接，如下所示：

```
<head>
  <title>Titles are required.</title>
  <link rel="stylesheet" href="/path/stylesheet.css">
</head>
```

在这个link元素里必须包括两个属性（attribute）：

rel="stylesheet"

定义被链接的文档与当前文档的关系。链接到样式表时，rel属性的值往往是stylesheet。

href="url"

提供.css文件的位置。

注意： Link元素是空的，所以你需要在XHTML文档里用一个紧跟着的斜线（<link/>）结束它，如例子所示。在HTML文档里会忽略这个紧跟着的斜线。

注意： 在HTML4.01和XHTML1.0中，link元素必须包含设置为text/css的type属性：

```
type="text/css"
```

HTML 5中不再需要type属性了。

你可以把多个link元素包含到不同的样式表中，并且它们都会发挥作用。如果有冲突，根据规则的顺序和层叠关系，列在后面的规则会覆盖前面的规则。

使用@import导入

另外一个把样式表附加到文档的方法是使用@import规则。@import规则是添加样式表的另一种方式，无论是在外部的.css样式表文档中，还是在style元素中，如下例所示：

```
<head>
  <style>
    @import url("/path/stylesheet.css");
    p { font-face: Verdana;}
  </style>
  <title>Titles are required.</title>
</head>
```

本例中使用了相对URL，但是也可以用绝对URL（以http://开始）。@import规则必须在样式表开始出现，在所有选择器之前。你可以导入多个样式表，并且它们都会发挥作用，但是后面文件的样式规则优先于前面的规则。

你可以在练习13-8中试着使用link和@import方法。

注意： 也可以不用url()符号提供URL：

```
@import "/path/style.css";
```

这里的绝对路径名又是从根目录开始，以保证.css文档总是能被找到。

组件样式表

因为可以从多个样式表中汇编信息，所以作为样式管理工具，组件样式表已经变得很受欢迎。很多开发者都喜欢重复使用样式表，例如，排版处理、布局规则以及表单相关的样式都在不同的样式表中，使用@import规则来把它们混合搭配起来。@import规则需要在使用选择器的规则之前。

clientsite.css的内容：

```
/* basic typography */
@import url("type.css");

/* form inputs */
@import url("forms.css");

/* navigation */
```

练习13-8 创建外部样式表

现在设计网页时可以使用嵌入式样式表，但最好在设计完成后，移动到外部样式表，这样它就可以被站点中的多个文档重复使用。我们将会为bistro菜单样式表这么做。

1. 打开最新版本的**bistro.html**文件。选择并剪切style元素里的所有规则，只留下<style>...</style>标签，因为过会儿会用到它们。
2. 创建一个纯ASCII的文本文档，并把所有的样式规则粘贴进去。确保没有意外的元素标签在里面。
3. 把文档保存为**menustyles.css**，放在与**bistro.html**文档相同的目录下。
4. 现在，返回**bistro.html**，添加一个@import规则用来附加一个外部样式表，如下所示：

```
<style>
@import url(menustyles.css);
</style>
```

保存文件，在浏览器中重新载入。应该看起来和样式表嵌入的结果一模一样。否则，返回确定你所有的做法都和我例子中说明的一样。

5. 删除整个style元素，这一次，在文档的head部分，使用link元素来添加一个样式表。

```
<link rel="stylesheet"
href="menustyles.css">
```

再次，通过保存文档并在浏览器中查看来测试你的作品。

注意：虽然模块化样式表是很有用的，但是在性能和缓存上还有一些问题。如果你使用这个方法，建议你在传到浏览器之前，把所有的样式都编辑到一个单独的文档中。不要发愁，你没必要手动去做：有一些工具可以帮你。LESS和SASS CSS Framework（在第18章会正式介绍）是可以提供编辑功能的两个工具。

```
@import url("list-nav.css");

/* site-specific styles */
body { background: orange; }

...more style rules...
```

记住，在设计网站的经验中，这是一个很好的技术。你会发现，正好有一些解决问题的方法，并且没必要在做新网站时都重新忙活一遍。模块化样式表是一个节省时间的好方法（见侧栏“注意”）。

自我测验

这一次用匹配题和选择题来测试你对重点内容的掌握情况：

- 背景色填充哪个区域？
 - 内容后面的区域
 - 内容周围添加的padding区域
 - 边框下
 - 边框周围的margin区域
 - 以上所有
 - a和b
 - a、b和c
- 哪个不是在CSS中指定颜色为白色的方法？
 - #FFFFFF
 - #FFF
 - rgb(255, 255, 255)
 - rgb(FF, FF, FF)
 - white
 - rgb(100%, 100%, 100%)
- 将下面的伪类与它的目标元素配对。

a. a:link	1. 已经单击过的链接
b. a:visited	2. 突出并可输入的元素
c. a:hover	3. 父元素的第一个子元素
d. a:active	4. 鼠标指针在上面的元素
e. :focus	5. 尚未访问的链接
f. :first-child	6. 正在单击的链接
- 将下列规则与图13-29(右边)的例子分别配对。图中所有的例子使用相同的源文档，其中包含已经应用过一些padding和border属性的paragraph元素。

- `body {background-image: url(graphic.gif);}`
- `p {background-image: url(graphic.gif); background-repeat: no-repeat;background-position: 50% 0%;}`
- `body {background-image: url(graphic.gif); background-repeat:repeat-x;}`
- `p {background: url(graphic.gif) no-repeat right center;}`
- `body {background-image: url(graphic.gif); background-repeat:repeat-y;}`
- `body { background: url(graphic.gif) no-repeat right center;}`

CSS回顾：色彩和背景属性

下面是本章讲过的属性的总结，按照字母表顺序排列：

属性	介绍
<code>background</code>	联合所有背景属性的快捷属性
<code>background-attachment</code>	指定背景图像是滚动的还是固定的
<code>background-clip</code>	指定背景图像延伸多远
<code>background-color</code>	指定元素的背景色
<code>background-image</code>	提供用作背景的图像的文件位置
<code>background-origin</code>	指定background-position如何计算（从边缘到边框、填充或者内容盒）
<code>background-position</code>	指定原图的位置
<code>background-repeat</code>	背景图像是否重复（拼贴）或者如何重复
<code>background-size</code>	指定背景图像的尺寸
<code>color</code>	指定前景（文本和边框）色
<code>opacity</code>	指定前景和背景的透明级别



图13-29：问题4的例子

第14章 盒子思想（填充、边框和空白边）

第11章介绍了CSS的基本概念盒子模型。根据盒子模型，文档中的每一个元素都产生一个盒子，很多属性，比如width、height、padding、border和margin属性都可以应用到这个盒子。通过给元素添加背景，你可能已经对盒子如何工作有了感知。本章讲述与盒子相关的所有属性。一旦学完这些基本知识，将准备讲述第15章。

以元素盒子的组件的概要开始，然后从里到外分析盒子属性：内容尺寸、填充、边框和空白边。

元素盒子

如我们所看到的，文档中的每一个元素都是块级别的内联的，这些元素形成矩形元素盒子。元素盒子的组件见图14-1中的图表。注意这些新的术语——这对接下来的学习会很有帮助。

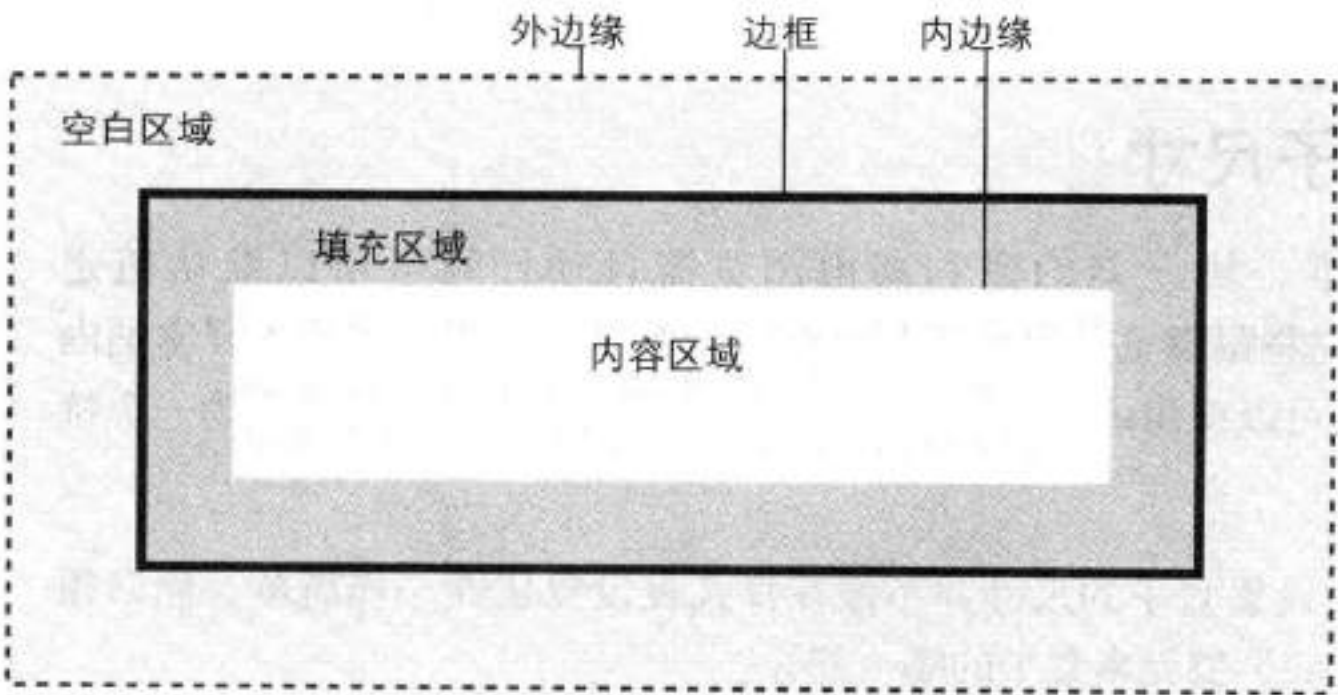


图14-1：根据CSS盒子模型，元素盒子的组成部分

本章内容

- 元素盒子的组件
- 设置盒子尺寸
- 在内容周围
- 添加填充
- 添加边框
- 添加空白边
- 指派显示角色
- 添加阴影

注意： 一个元素盒子的总尺寸包含元素的内容，加上所有的填充、边框和空白。

内容区域

元素盒子的核心是内容本身。图14-1中，白盒子里的文本表示内容区域。

内边缘

内容区域的边缘属于元素盒子的内边缘。虽然在图14-1中可以通过颜色突变使内边缘明显，但是在实际网页中，内容区域的边缘将不可见。

填充

填充是处于内容区域和可选的边框之间的区域。在图14-1中，橙黄色表示填充区域。填充是可选择的。

边框

边框是环绕元素和它的填充的线（或者样式化的线）。边框是可选的。

空白边

空白边是添加到边框之外的可选空间。在图14-1中，浅蓝色表示空白边，但实际上，空白边总是透明的，允许父元素的背景穿透它而显示。

外边缘

空白边的外边缘构成了元素盒子的外边缘。这是元素在网页中占的总面积，并且它的宽度是内容的宽度，以及应用到元素的填充、边框、空白边的宽度的总和。在图14-1中点划线表示外边缘，但是在实际网页中，空白区域的外边缘不可见。

所有元素都有这些盒子组件；然而，如你所见，基于元素是块还是内联的，一些属性表现各异。事实上，马上可以从盒子区域看到一些不同之处。

指定盒子尺寸

默认情况下，块元素的宽和高由浏览器自动计算（所以默认值是auto）。它将跟浏览器窗口或其他块元素容器一样宽，高度正好容纳内容。然而，可以使用width和height属性使元素内容区域宽度为一个特定值。

遗憾的是，设置盒子的尺寸并不像在样式表设置属性一样简单。你必须明确知道到底调整元素盒子的哪一部分。

CSS3提供了两种方法来指定一个元素的大小。默认的方法——CSS1中介绍的方法——对content box设定宽度和高度值。这意味着所得到的元素

的大小将是指定的尺寸，加上适量的填充和边框尺寸。另一种方法——CSS3中的`box-sizing`属性——对border box指定宽度和高度值，其中包括内容、填充和边框。有了这个方法，可见元素盒子（包括填充和边框），将被精确地指定。有些人觉得这是一个更直观的方法。在这一节里将熟悉这两种方法。

无论选择哪一种方法，都可以为块级元素和非文本内联元素（如图像）只指定宽度与高度。`width`和`height`属性并不适用于内联文本（又名非替换）元素，会被浏览器忽略。换句话说，不能指定锚（a）或者strong元素的宽度和高度（见“注意”）。

注意：事实上，有一种办法可以对内联元素（如锚a）应用`width`和`height`属性：使用`display`属性，强制它们像块元素一样，本章最后会讲述这个方法。

box-sizing

属性值： `content-box|border-box`
 默认值： `content-box`
 适用对象： 所有元素
 是否可继承： 否

width

属性值： 长度值|百分比值|`auto`|`inherit`
 默认值： `auto`
 适用对象： 块级元素和可替代内联元素（如图像）
 是否可继承： 否

height

属性值： 长度值|百分比值|`auto`|`inherit`
 默认值： `auto`
 适用对象： 块级元素和可替代内联元素（如图像）
 是否可继承： 否

调整内容盒子的尺寸（默认）

默认情况下（如果在你的样式中不包括`box-sizing`规则），`width`和`height`属性就会应用于内容盒子。目前所有的浏览器都是这样解释宽度和高度值的，但也可以通过设置`box-sizing:content-box`明确地指定该行为。

在下面的示例和图14-2中给出了一个简单的盒子，宽度为500像素，高度为150像素，20像素的填充，2像素的边框，20像素的边距（margin）。使用内容盒子模型，`width`和`height`值只应用于内容区域。


```
p {
  background: #c2f670;
  width: 500px;
  height: 150px;
  padding: 20px;
  border: 2px solid gray;
  margin: 20px;
}
```

最终可见的元素盒子的宽度是544像素（内容加上40像素的填充和4像素的边框）。当赋予边距40像素时，整个元素盒子的宽度是584像素。知道了元素的大小是至关重要的，这样才能了解并合理安排布局。

$20\text{px} + 2\text{px} + 20\text{px} + 500\text{px width} + 20\text{px} + 2\text{px} + 20\text{px} = 584$ 像素

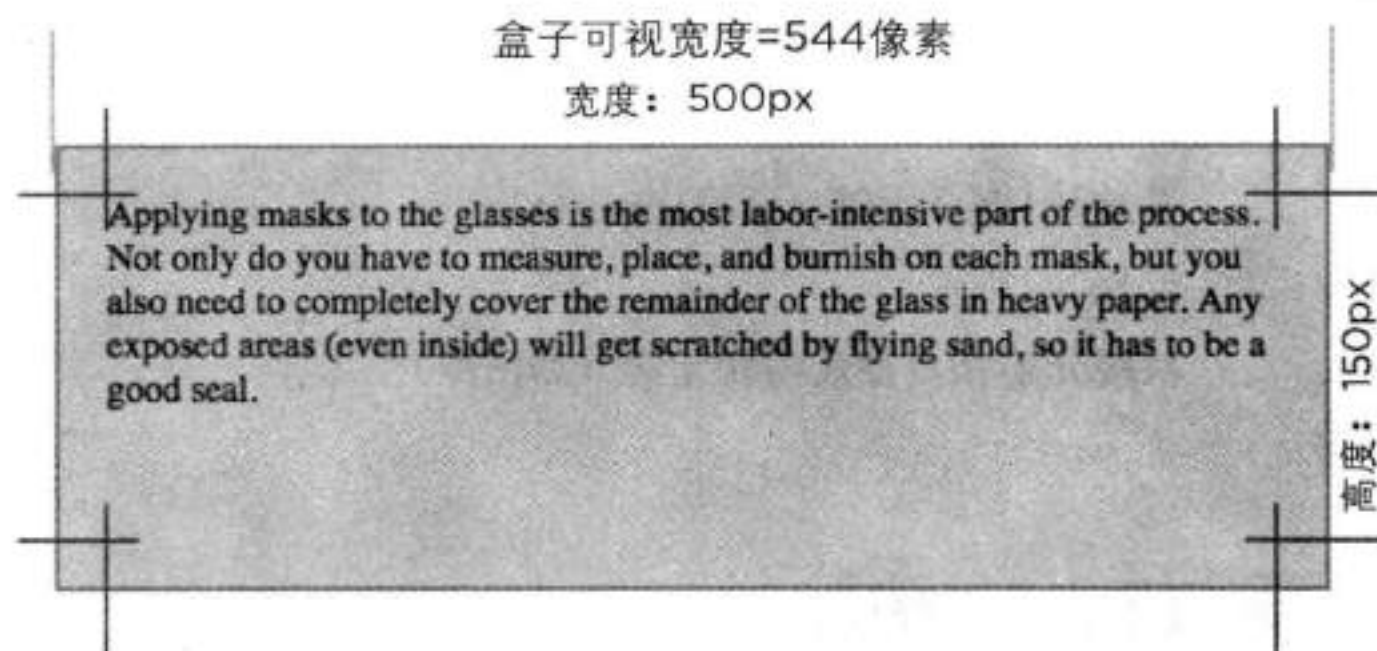
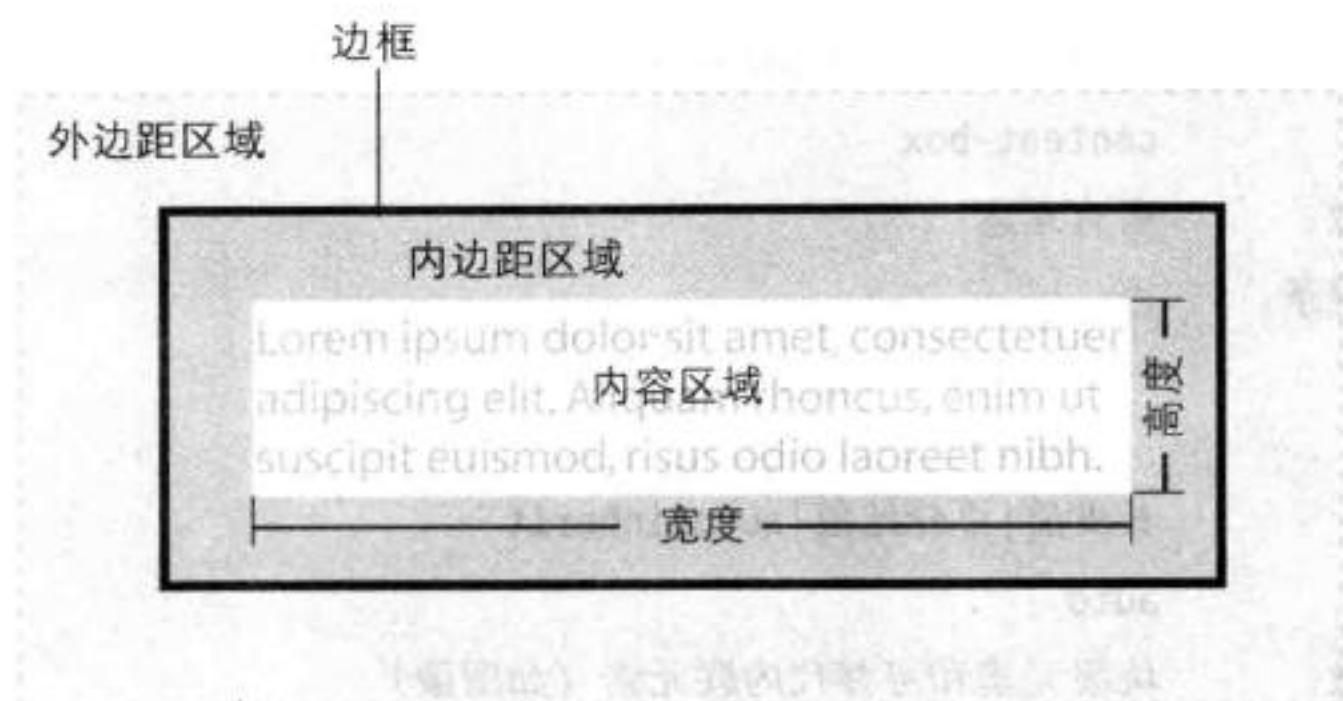


图14-2: 使用content-box模型指定width和height

边框盒模型

指定元素大小的另一种方法是对整个可见盒子应用宽度和高度，包括填充和边框。因为这不是浏览器默认的行为，所以你需要在样式表中显式地设置`box-sizing: border-box`。

警告： 避免对border-box模型使用max-和min-宽度与高度。众所周知，这样浏览器会出现问题。

让我们来看看上一节的同一个例子：使用border-box方法，指定盒子宽500像素，看看会如何（见图14-3）。请注意，在写本书时，你需要提供-webkit-和-moz-的供应商前缀，才能在Safari、Chrome和Firefox浏览器中生效。Opera和Internet Explorer8及其更高版本无须后缀就可以支持（见“注意”）。

```
p {  
  ...  
  -webkit-box-sizing: border-box;  
  -moz-box-sizing: border-box;  
  box-sizing: border-box;  
  width: 500px;  
  height: 150px;  
}
```

最大和最小尺寸

CSS2引入了属性为块元素设置最小和最大的高度与宽度。如果你想限制元素尺寸，它们应该是很有用的。

max-height, max-width,

min-height, min-width

属性值： 百分比|长度|none|inherit

这些属性仅对块级元素和可替代元素（如图像）使用。当使用内容盒子模型时，这些值只能适用于内容区域，所以如果你对填充、边框或者空白边距应用这些属性时，即便已经指定max-width或者max-height属性，也会使整个元素盒子变得很大。IE 6及其之前的版本不支持这些属性。

注意：IE 6和IE 7不支持`box-sizing`属性，但是也有一些办法。可以使用条件注释为IE 8之前的版本提供一个单独的样式表。一个更简洁的方法是使用Christian Shaefer的`box-sizing polyfill`（可以使不支持的浏览器模拟支持的脚本），可以从github.com/Schepp/box-sizing-polyfill获取该脚本。把脚本放在服务器上，并且按Github页面上的说明去做就可以。

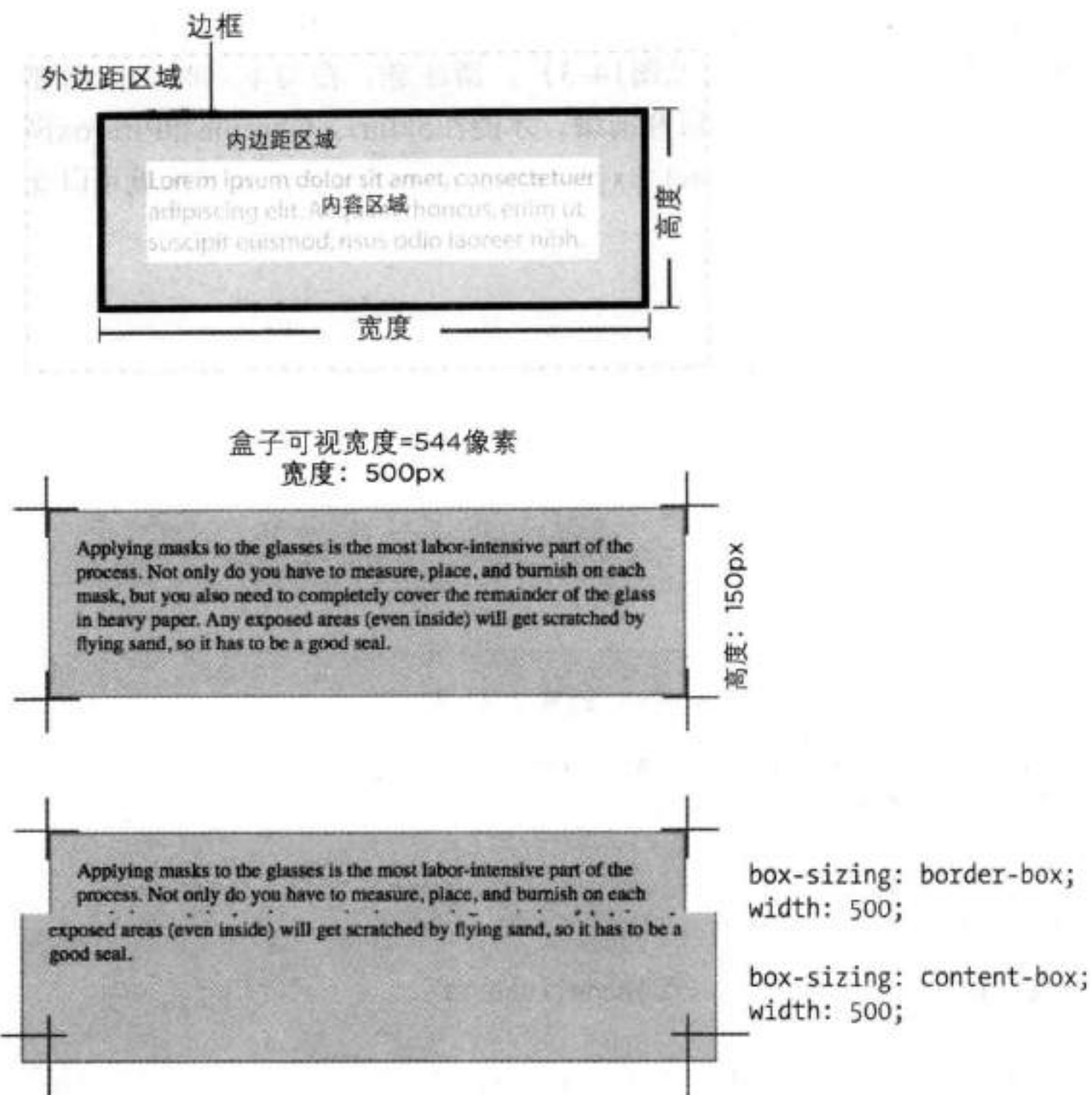


图14-3：使用border-box方法调整元素尺寸。最下面的图是各种调整尺寸方法的对比

许多开发人员发现边界盒子模型是调整元素尺寸更直观的方式。对于以百分比来指定宽度来说，它是特别有用的方法，它也是响应式设计的基石。例如，可以使两列宽50%，就可以让它们一个接着一个，而不必浪费时间来计算填充和边框的宽度。事实上，通过使用通用选择器，一些设计人员在文档中仅设置`everything`来使用border-box模型：

```
* {box-sizing: border-box;}
```

阅读Paul Irish的文章来详细了解这项技术：paulirish.com/2012/box-sizing-border-box-ftw/。

IE盒子模型的问题

过去的网页设计者应当记得border-box方法曾经是IE的Bug。在1996年，CSS1规约以标准方式描述了内容盒子模型，这样浏览器才可以计算元素的尺寸。但是这并不能阻止微软在IE 5中实现它自己的border-box模型，引起的差异让开发者头疼了好几年。

IE 6最终转向了标准内容盒子，但是也只是在标准模式(Standards Mode)下如此。当文档不以合法的DOCTYPE声明开始时，IE 6和IE 7还是会回到Quirks Mode，并且使用过去的IE border-box模型，这也是包含DOCTYPE的一个原因。幸运的是，这些浏览器已经时日无多了。

要想详细了解盒子模型的发展历史，可以阅读Jeff Kaufmann的文章“The Revenge of the IE Box Model” (www.jeffik.com/news/2012-02-18.html)。

指定高度

一般来说，不常指定元素的高度。更多的是，使用中等的尺寸，允许根据文本和其他内容来自动计算高度。这就允许它根据字体尺寸、用户设置或其他因素而变。如果指定包含文本的元素的高度，请确保内容与之相适应。幸运的是，CSS给了我们一些选项，将在下一节介绍。

处理溢出

当元素设置得太小而容不下内容时，就可能需要使用overflow属性来指定如何处理内容不适合的问题。

overflow

属性值: visible|hidden|scroll|auto|inherit
 默认值: visible
 适用对象: 块级别元素和可替换的内联元素（比如图像）
 是否可继承: 否

图14-4 给出了overflow的预定值。其中，不同的值应用于同一个元素，元素是150像素宽的正方形。背景色使内容的边缘很明显

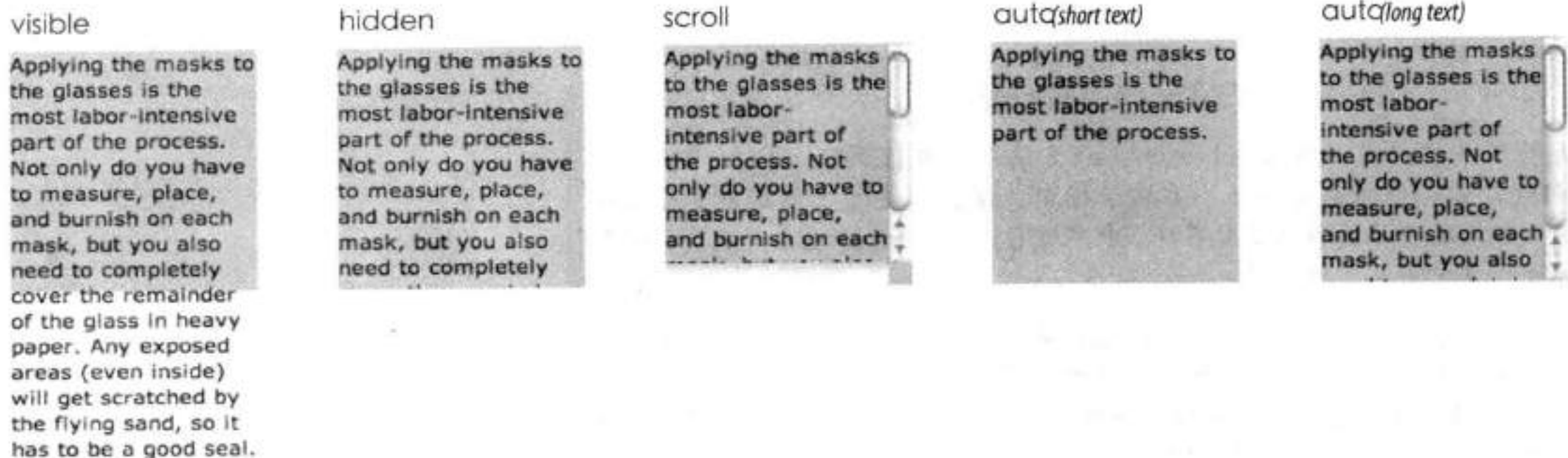


图14-4：处理内容溢出的选项

visible

默认值是`visible`，这个值允许内容在元素盒子外悬浮，从而内容可以全部可见。

hidden

当`overflow`属性值为`hidden`时，不适用的内容将被忽略，并且不会在元素内容区域边缘之外出现。

scroll

指定`scroll`时，滚动条将添加到元素盒子中，让用户可以滚动内容。请注意，当将值设置为滚动时，滚动条将始终存在，即使内容适合指定高度。

auto

`auto`值允许浏览器来决定如何处理溢出。在大多数情况下，只有当内容不适合并且必要的时候才添加滚动条。

填充

填充是内容区域和边框（如果没有指定就是边框应该在的地方）之间的空间。我发现用背景色或者边框来添加填充很有用。它可以给内容一点呼吸的空间，防止背景的边缘或边缘正好与文本冲撞。

可以分别给（块级别或内联）元素的每条边添加填充。同样有一个快捷`padding`属性，让你一次添加所有边的`padding`属性。

警告

移动设备中的滚动区域

在写本书的时候，`overflow`属性在一些移动设备（大多数是过时的）会引起问题，虽然页面中的可滚动区域其实是节省空间的一个好办法。一些移动浏览器简单的隐藏`overflow`文本。其他添加一个滚动条，但是需要使用两个手指才能控制。

一个解决办法是在有问题的浏览器上使用Scott Jehl的“`overthrow`”脚本来模拟支持。可以详细了解`Overthrow`: filamentgroup.com/lab/overthrow。

padding-top、padding-right、padding-bottom、padding-left

属性值： 长度计量值|百分比值|inherit

默认值： 0

适用对象： 除了table-row、table-row group、table-header-group、table-footer-group、table-column和table-column-group的所有元素

是否可继承： 否

padding

属性值： 长度计量值|百分比值|inherit

默认值： 0

适用对象： 所有元素

是否可继承： 否

使用padding-top、padding-right、padding-bottom和padding-left属性，可以指定元素每条边的填充值，见下面的例子，如图14-5所示(注意，我还添加了背景色使填充区域的边缘显而易见)。

```
blockquote {
  padding-top: 1em;
  padding-right: 3em;
  padding-bottom: 1em;
  padding-left: 3em;
  background-color: #D098D4;
}
```

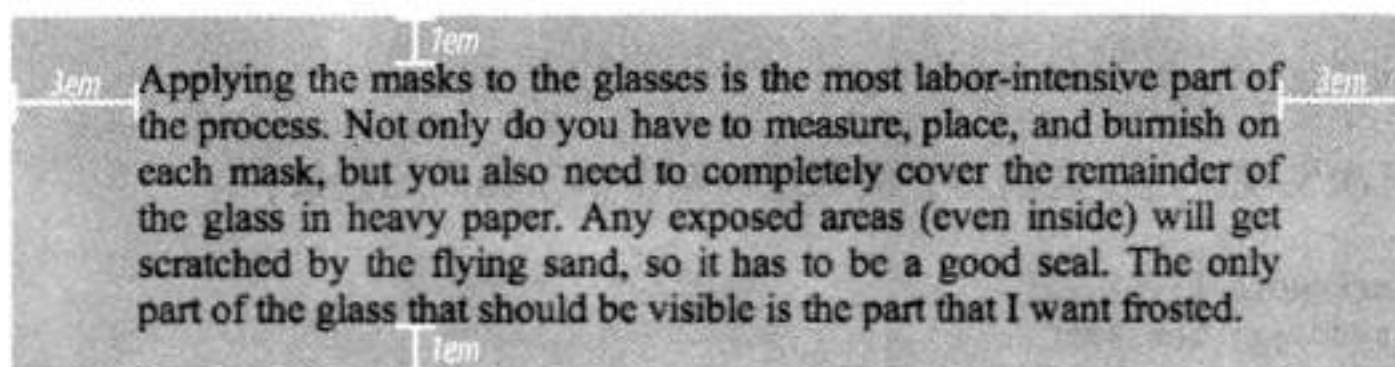


图14-5：在元素内容周围添加填充

可以用CSS的长度单位（一般就是指em和px）来指定padding值，也可以用父元素宽度的百分比值来指定。是的，父元素的宽度是参考，即便对于top和bottom填充也是如此。如果父元素的宽度改变，子元素各个方向的填充值也将发生变化，这也使得百分比值管理起来有些棘手。

快捷padding属性

作为按边填充的替代方案，我们可以使用快捷padding属性来添加元素周围的所有填充。语法非常有趣：可以指定给padding属性四个、三个、两个或一个值。接下来介绍它的工作方式，以四个值开始。

快速浏览

快捷值

1个值

`padding: 10px;`

应用于所有方向

2个值

`padding: 10px 6px;`

第一个值用于上和下;

第二个值用于左和右

3个值

`padding: 10px 6px 4px;`

第一个值用于上

第二个值用于左和右

第三个值用于下

4个值

`padding: 10px 6px 4px 10px;`

依次顺时针应用于上, 右, 下, 左边 (TRBL)。

当你提供四个padding值时, 它们延顺时针应用到每一个方向, 从向上开始。一些人使用记忆法” TrouBLE” 表示顺序上 (Top)、右 (Right)、下 (Bottom) 和左 (Left)。

```
padding: top right bottom left;
```

使用这个padding属性, 可以重新完成上面那个例子。如下所示:

```
blockquote {
  padding: 1em 3em 1em 3em;
  background-color: #D098D4;
}
```

如果左边和右边的padding值相同, 可以简写为三个值。右值 (串中的第二个值) 可以作为左值的镜像。这就好像左值丢失了, 只能在两个方向都用右值。三个值的语法如下:

```
padding: top right/left bottom;
```

这个规则与上个例子是等效的, 因为元素左边和右边的填充都将设置为3em。

```
blockquote {
  padding: 1em 3em 1em;
  background-color: #D098D4;
}
```

继续这个话题, 如果只提供两个值, 那么第一个值用于上下边, 第二个值用于左右边:

```
padding: top/bottom right/left;
```

我们再次发现, 这个规则将完成与上面两个例子相同的效果:

```
blockquote {
  padding: 1em 3em;
  background-color: #D098D4;
}
```

注意, 前面所有的例子都有相同的视觉效果, 如图14-5所示。

最后, 如果你只提供一个值, 它会被应用到元素的四条边。下面的例子为div元素的所有边都赋予了15像素的填充。

```
div#announcement {
  padding: 15px;
  border: 1px solid;
}
```

练习14-1 添加一些填充

本练习中，我们将运用基本的盒子属性来改进一个虚拟网店（Jenware.com）的外观。我将给你做好标记化源文档，创建处理文本格式、颜色和背景的风格表。文档jenware.html可以在材料目录（www.learningwebdesign.com/4e/materials）中得到。

图14-6显示了Jenware主页改进前后的快照。我们将采取一些步骤，使这个网页有个像样的形状，填充仅是个开始。

哇！导航部分太丑了！别急，我们将在第15章给它一个漂亮的水平导航菜单。

在浏览器和文本编辑器中打开jenware.html，看看你的工作内容。文档被分成两个主要div元素（“intro”，和“content”），#content div也被分成了“products”和“testimonials”。背景色添加给body元素，#nav、#products和#testimonials。我还给页面的顶部添加了一个渐变，给“testimonials”div的背景添加了感叹号图像。剩下的规则都是用来格式化文本的。

1. 我们要做的第一件事是给products的div元素添加填充。周围1em的填充应该很合适。找到#products选择器并且添加padding声明。

```
#products {
  background-color: #FFF;
  line-height: 1.5em;
  padding: 1em;
}
```

2. 然后，将对testimonials部分应用一些奇异的效果。我们将清空div的左边的一些空间，这样漂亮的感叹号背景图就可见了。有许多方法来应用不同的padding值到各个方向，但是，我将故意覆盖以前的声明，来让你练练手。

使用padding快捷属性给testimonials div元素的所有方向都添加1em的填充。再写第二个声明，只给左padding添加55像素。因为padding-left声明第二次出现，所以它将覆盖padding快捷属性中应用的1em。



图 14-6: Jenware 首页改进前后的快照

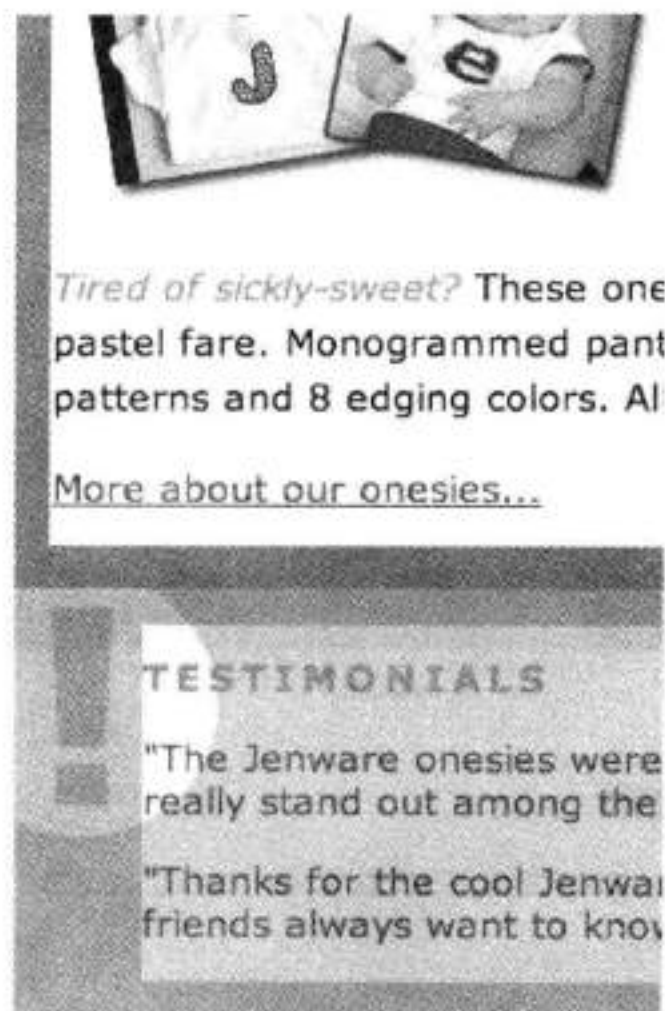


图14-7：粉色的区域表示填充添加到了 testimonials 部分。蓝色表示 products 部分的填充。

```
#testimonials {
  background: #FFBC53 url(images/ex-circle-corner.gif) no-repeat
  left top;
  color: #633;
  font-size: .875em;
  line-height: 1.5em;
  padding: 1em;
  padding-left: 55px;
}
```

3. 保存你的工作，在浏览器中查看。testimonials 和 products 描述在各自的盒子中，看起来应该更舒适。图14-7突出了padding的填充部分。

边框

边框就是环绕内容区域和它的（可选的）填充的一条线。幸运的是，它并不怎么麻烦。你可以从8个边框样式中选择，并且设置成你喜欢的任何宽度和颜色。你可以在元素的四周，或者仅某一个方向使用边框。CSS3 引入了属性，可以使边框角柔化，甚至可以给边框应用图像。我们将以不同的边框样式来开始边框探索之旅。

边框样式

样式是border属性中最重要的，因为根据CSS规范，如果没有指定边框样式，边框将不存在。换句话说，你必须总是声明边框的样式，否则其他的边框属性都会被忽略。

边框样式可以使用快捷border-style属性，每次给一边应用边框样式。

border-top-style、border-right-style、
border-bottom-style、border-left-style

属性值： none | dotted | dashed | solid | double | groove | ridge | inset |
outset | inherit

默认值： none

适用对象： 所有元素

是否可继承： 否

设计提示

底部边框，而不是下划线

关闭链接的下划线，并且使用经典的底部边框替代，这是现代Web设计中很常用的设计技巧。它会使链接看起来显眼，也使得它们在普通文本中更突出。

border-style

属性值: none|dotted|dashed|solid|double|groove|ridge|inset|outset|inherit

默认值: none

适用对象: 所有元素

是否可继承: 否

border-style属性值是边框样式的十个关键字之一，如图14-8所示。

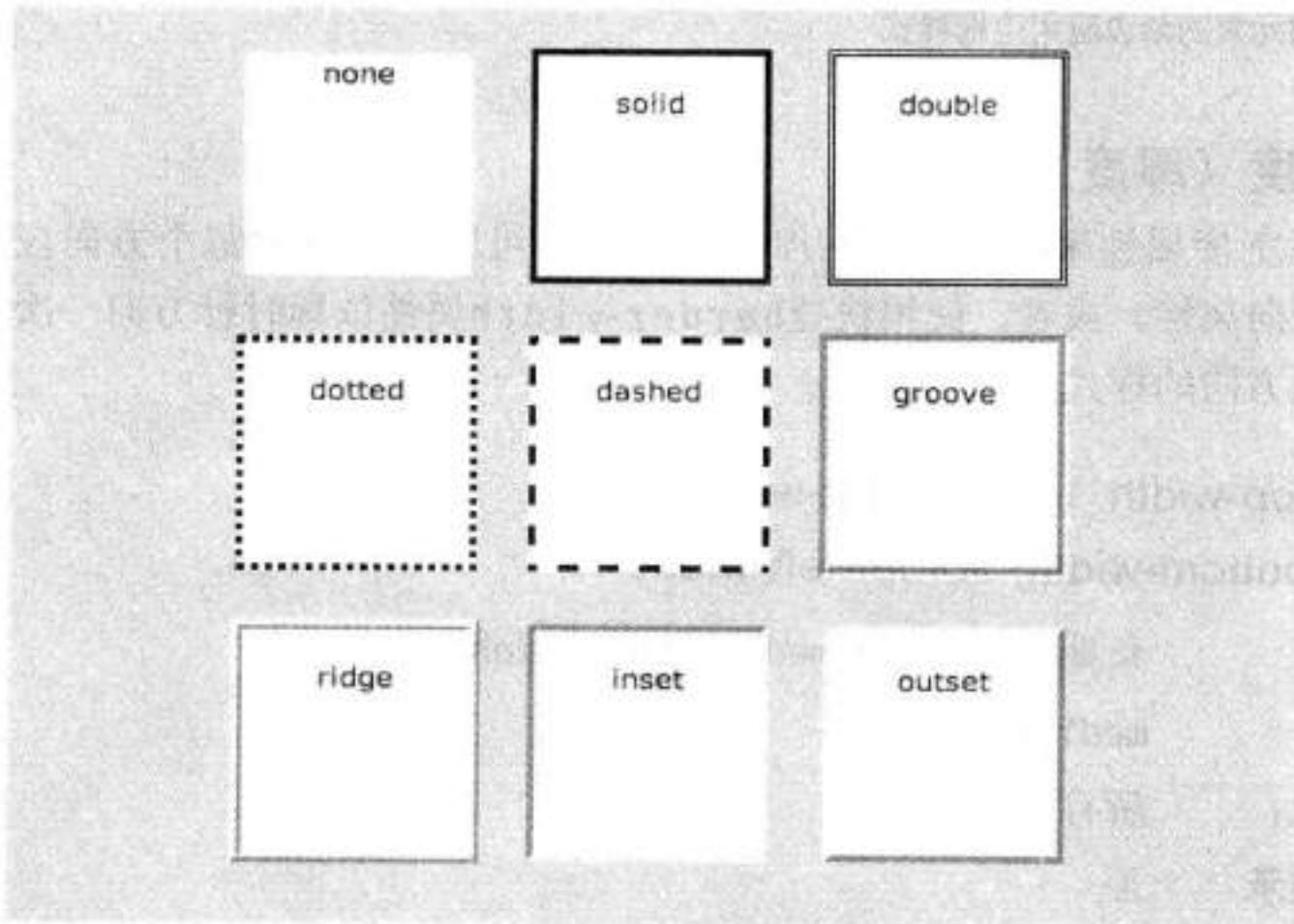


图14-8: 可用边框样式（显示为默认的中等宽度）

可以使用定向的边框样式属性（border-top-style、border-right-style、border-bottom-style和border-left-style），对元素的一边应用样式。如果没有指定宽度，将使用默认的中等宽度。如果没有指定颜色，边框将使用元素的前景色（同文本）。

在下一个例子中，我将应用不同的样式到元素的各个边，实际演练单向边框样式属性（如图14-9所示）。

```
div#silly {
  border-top-style: solid;
  border-right-style: dashed;
  border-bottom-style: double;
  border-left-style: dotted;
  width: 300px;
  height: 100px;
}
```

border-style快捷属性与前面的padding快捷属性一样，都是按顺时针

(TRouBLe) 方式工作。我们可以为四个方向提供四个值，当左/右边框和上/下边框相同时可以提供更少的值。前面例子所示的边框效果也可以用border-style属性指定，如下所示，结果将与图14-9中的相同。

```
border-style: solid dashed double dotted;
```

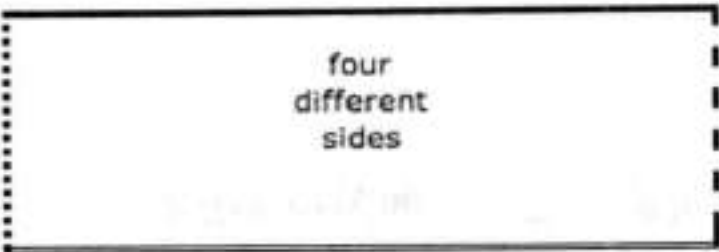


图14-9：对元素的单边应用边框样式

边框宽度（厚度）

使用边框宽度属性来指定边框的厚度。同样，可以给元素的每个方向应用一个单向属性，或者，使用快捷border-width属性依顺时针方向一次指定多个方向的样式。

border-top-width, border-right-width,
border-bottom-width, border-left-width

- 属性值： 长度单元|thin|medium|thick|inherit
- 默认值： medium
- 适用对象： 所有元素
- 是否可继承： 否

border-width

- 属性值： 长度单元|thin|medium|thick|inherit
- 默认值： medium
- 适用对象： 所有元素
- 是否可继承： 否

指定边框宽度的最常用方法是使用像素计量值。然而，也可以使用其中一个关键字（thin、medium或thick），留给浏览器解析。

这个例子包含一个混合值（见图14-10），注意，还包含border-style属性，因为如果不包含，边框就不会显现。

```
div#help {  
  border-top-width: thin;  
  border-right-width: medium;  
  border-bottom-width: thick;  
  border-left-width: 12px;  
  border-style: solid;
```

```
width: 300px;
height: 100px;
}
```

或者:

```
div#help {
border-width: thin medium thick 12px;
border-style: solid;
width: 300px;
height: 100px;
}
```

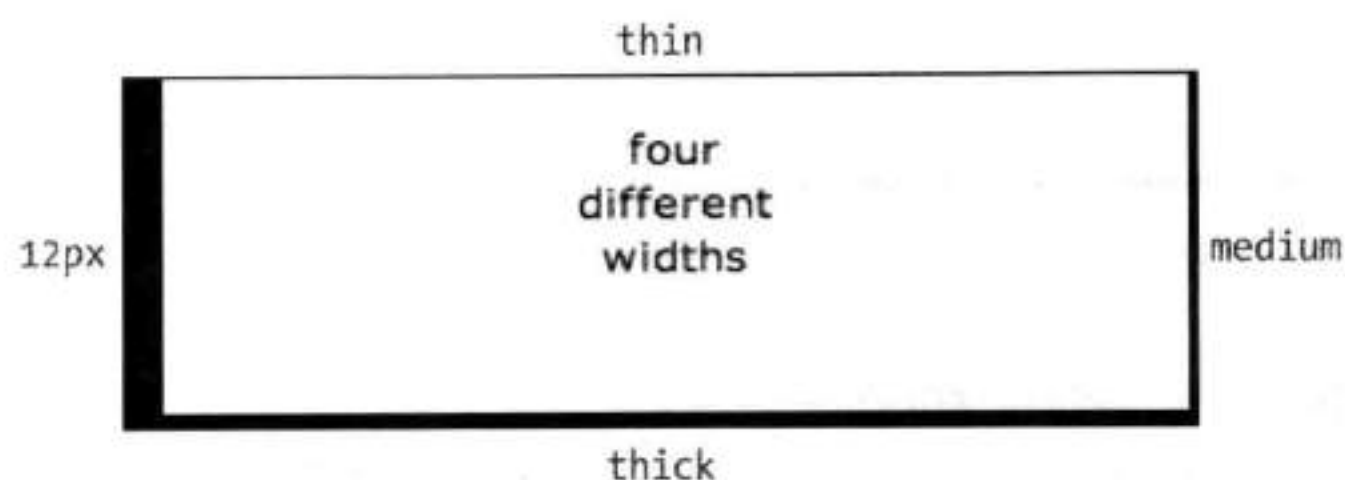


图14-10: 指定边框宽度

边框颜色

边框颜色用相同的方法指定: 使用专向属性或border-color快捷属性。当指定边框颜色时, 它将覆盖color属性设置的元素的前景色。

border-top-color、border-right-color、
border-bottom-color、border-left-color

属性值: 颜色名或RGB值|transparent|inherit
默认值: 元素的color属性值
适用对象: 所有元素
是否可继承: 否

border-color

属性值: 颜色名或RGB值|transparent|inherit
默认值: 元素的color属性值
适用对象: 所有元素
是否可继承: 否

你已经知道指定颜色值的所有知识了, 并且你可能也习惯于使用快捷属性, 所以, 使这个例子保持短小而美好 (如图14-11所示)。这里, 我已

设计提示

将这个规则的颜色设置得与背景色相近, 并且将宽度设置得细一些 (thin), 我们会得到比较舒服的效果。

注意: CSS2添加transparent关键字值, 允许父元素的背景透过边框, 即使指定过边框宽度也是如此。当CSS创建悬停 (:hover) 效果时, 这个很有用, 因为当鼠标不在元素上面时, 边框出现的空间还存在。透明值不受IE 6的支持。

经给快捷border-color属性提供两个值，来使div元素的上边和下边为栗色，左边和右边为浅绿色。

```
div#special {  
  border-color: maroon aqua;  
  border-style: solid;  
  border-width: 6px;  
  width: 300px;  
  height: 100px;  
}
```

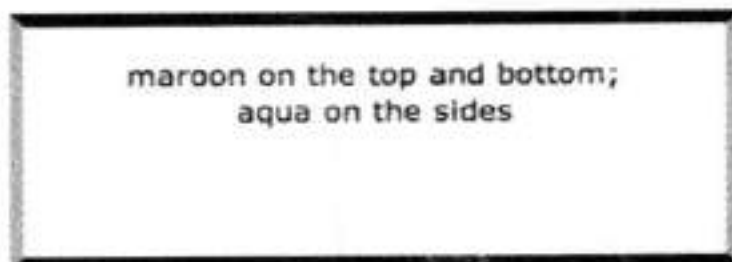


图14-11：指定边框颜色

联合使用style、width和color

CSS的作者并不吝惜使用边框的快捷属性。他们同样创建了在一个声明中同时提供style、width和color的属性。同样地，可以专门为每个方向指定外观，也可以使用border属性一次改变所有四个方向的外观。

border-top, border-right, border-bottom, border-left

属性值： *border-style border-width border-color|inherit*

默认值： 每个属性的默认值

适用对象： 所有元素

是否可继承： 否

border

属性值： *border-style border-width border-color|inherit*

默认值： 每个属性的默认值

适用对象： 所有元素

是否可继承： 否

border属性值和某一方向的border属性值可以包括任何顺序的style、width和color值。不需要声明所有的三个值，但是要记住，如果边框样式值遗漏了，那就不会显示边框。

border快捷属性与我们学习过的其他的快捷属性工作方式有点不同，它有一个值集，并且总是应用到元素的四个方向。换句话说，它并不是使用在其他快捷属性中看到的顺时针“TRBL”规则。

这是为数不多的几个可用的border快捷属性的例子，它可以让我们对它们如何工作有个大概的认识。

```
h1 { border-left: red .5em solid; }    /*left border only*/
h2 { border-bottom: 1px solid; }       /*bottom border only*/
p.example { border: 2px dotted #663; } /*all four sides*/
```

使用border-radius提供圆角

最近几年，圆角的盒子已经成为一种潮流样式。原先，页面的圆角只能使用图像或者其他标记来完成。现在很幸运，所有当今浏览器都可以使用CSS的border-radius属性来使元素成为圆角。这也就意味着不用从服务器下载图片，设计者也不用做太多的Photoshop工作了。本节将从CSS3规约中的代码开始，学习一些例子，最后说明浏览器的支持状态。

就像之前的其他属性一样，border-radius既有一些单独的属性，也有快捷属性。

border-top-left-radius, border-top-right-radius,
border-bottom-right-radius, border-bottom-left-radius

属性值： 长度值|百分比值

默认值： 0

适用对象： 所有元素

是否可继承： 否

border-radius

属性值： 1、2、3或4长度或者百分比值

默认值： 0

适用对象： 所有元素

是否可继承： 否

要使元素的直角成为圆角，可以简单地使用一个border-radius属性，但要注意，如果你想看到效果，元素必须有边框，或者元素的背景色必须不同于页面的背景色。属性值是以em或者像素为单位的。也可以使用百分比，而且百分比可以保持元素的曲线比例，但是你可能会遇到一些不一致的浏览器。

可以单独设置某个角，也可以使用快捷属性border-radius属性。如果你为border-radius提供一个值，它会应用于四个角。如果是四个值，会按照顺时针方向来应用（top-left、top-right、bottom-right和bottom-left）。当你指定两个值时，第一个值会应用于左上角和右下角，第二个值适用

于其他两个角落。

图14-12列出了不同border-radius值的结果。通过设定不同的值，可以获得不同的效果：略微柔化的角，长菱形。

p { width: 200px; height: 100px; background: darkorange; }



图14-12：使用border-radius属性，使元素盒子的角成为圆角

椭圆角

到目前，我们完成的都是圆角，但是也可以通过指定两个值创建椭圆角：第一个值是水平半径，另一个是垂直半径（如图14-13 A和B所示）。



A border-top-right-radius: 100px 50px;



B border-top-right-radius: 50px 20px;
border-top-left-radius: 50px 20px;

如果你想使用快捷属性，水平半径和垂直半径需要用斜线分隔（否则，它们会被误以为是圆角的值）。下面的例子将水平半径设置为60px，垂直半径设置为40px（如图14-13C所示）。

C border-radius: 60px / 40px;



如果你想得到比较怪异的效果，border-radius快捷属性可以为四个角指定不同的椭圆度。所有的水平半径按顺时针方向排列在斜线的左边，对应的垂直半径列在斜线的右边（如图14-13D所示）。

D border-radius: 36px 40px 60px 20px / 12px 10px 30px 36px;



图14-13：椭圆角盒子

浏览器支持

就像之前提到的，当前主流的浏览器支持CSS3规约语法的border-radius，这是个好消息！早期的Safari、Chrome和Firefox本来就可以支持圆角，但是它们使用浏览器前缀来完成了自己的语法（可以看看侧栏“过去的半径前缀”）。而IE 9之前的版本无法支持border-radius。

那么IE 6~IE 8怎么办呢？其实你的网站的成功和可用性并不取决于圆角，所以只需要不断做得更好就可以了：IE上提供完美的方形盒子，其他浏览器上再用圆角。如果由于某些原因，即便在老版本的IE中，你的盒子也必须是圆角，那么就需要依靠JavaScript补丁了，比如Curvy Corners (www.curvycorners.net)。

过去的半径前缀

老版本的Firefox和Webkit浏览器只能通过它们自己的供应商前缀属性来支持边框圆角。不同于IE顽固了十年，其他浏览器自动进行了升级，或者至少更符合标准。因此，很多开发者已经停止为border-radius使用前缀。但是在一些老版本的移动浏览器上，它们可能仍然有用，所以如果在早前的Android版本中，圆角对你的产品很重要，那么就可能需要包含-webkit-前缀。

Webkit browsers

(Safari <5, Chrome <10.5, Android <2.2, iOS < 4)

```
-webkit-border-top-left-radius
-webkit-border-top-right-radius
-webkit-border-bottom-left-radius
-webkit-border-bottom-right-radius
-webkit-border-radius
```

Firefox (Mozilla)

(Firefox <4)

```
-moz-border-radius-topleft
-moz-border-radius-topright
-moz-border-radius-bottomleft
-moz-border-radius-bottomright
-moz-border-radius
```

完美的图片边框

本节已用大量篇幅讨论CSS边框……盒子周围的线居然都要说这么多！我把边框最神奇也最棘手的部分放在最后来讲。在本节中会使用border-image属性用你所选择的图像填充边框盒子的边和角。这个属性不需要切分图像文件，也没必要添加一堆无用的标记来包含图像。现在，使用CSS就可以将一个单独的图像应用到元素上。

应该注意到，在写本书时，IE的所有版本（甚至IE 9和IE 10）都不支持边框图像，所以你最多能做的就是提供一种舒适的后备色和边框样式。支持图像的浏览器（Safari、Chrome、Firefox和Opera）需要它们各自的供应商前缀来支持这个功能。

border-image

- 属性值: *border-image-source border-image-slice border-image-width border-image-outset border-image-repeat*
- 默认值: 每个属性的默认值
- 适用对象: 除了border-collapse重叠的table元素的所有元素
- 是否可继承: 否

让我们以图像来直观地告诉你这里讨论的是是什么。图14-14显示了两个元素和各自用来填充边框的图像。注意图像的角精确地填充了元素的角。元素的边可以拉伸（如图14-14上图所示），也可以拼贴（如图14-14下图所示）。

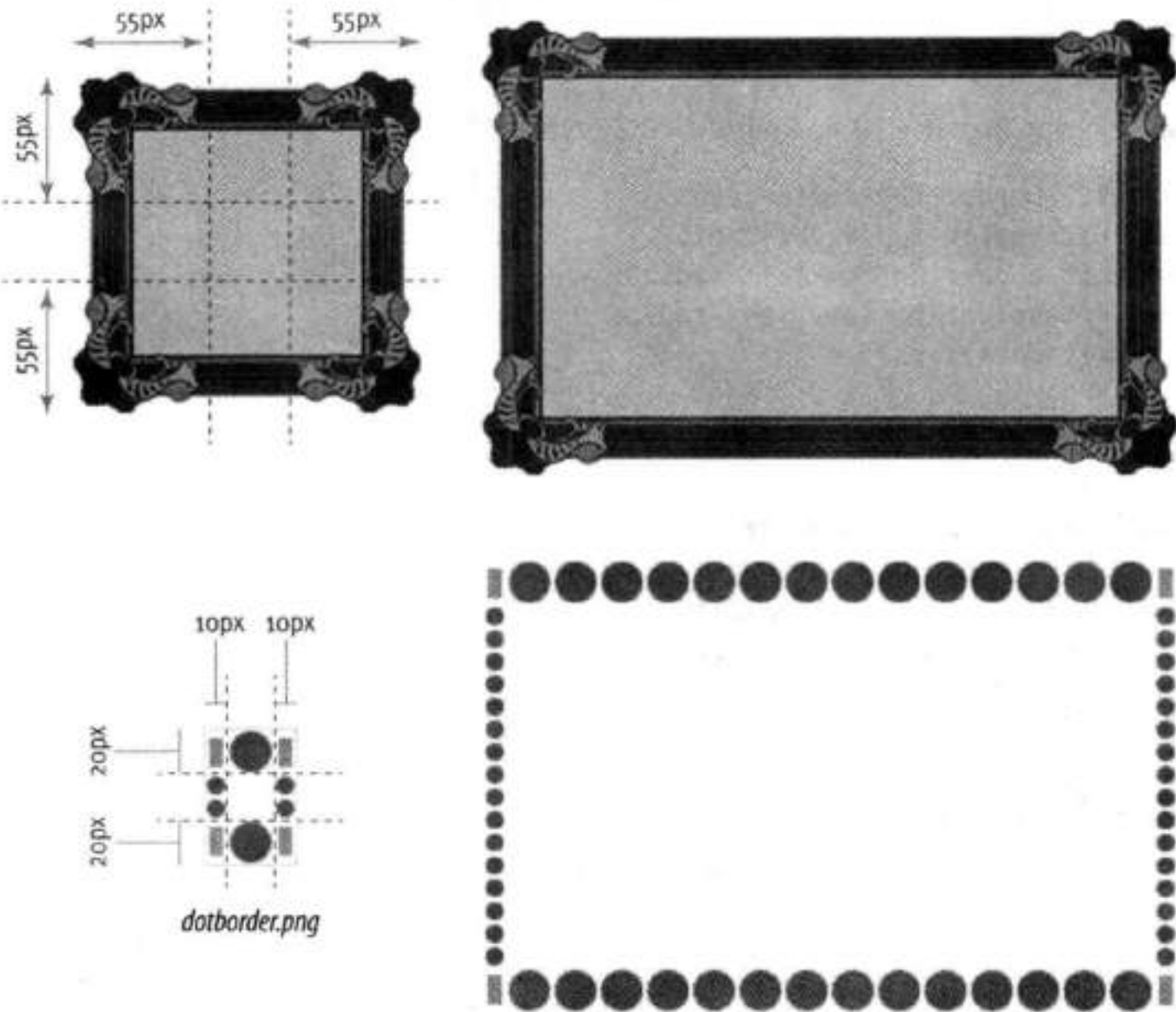


图14-14：使用拉伸边的边框图像和使用重复拼贴边的边框图像

下面该看看代码了！在写本书的时候，border-image快捷属性的三部分得到了支持（如图14-15所示）。详见侧栏“边框图像规约”。

```
border-image: url(fancyframe.png) 55 55 55 55 stretch;
```

A B C

图14-15：border-image规则的组成部分

A中的URL符号包含边框图像文件的位置。

下一个值表示切边的位置，它将图形分成了9部分B。这些值是图像每条边的偏移量，还是按照TRouBLe的样式，以顺时针方向来列（top、right、bottom、left）。可以使用快捷值，比如提供一个值，使所有的切边线都与四条边距离相同。当指定像素值时，可以忽略单位“px”。当然也可以指定百分比值。

最后一个关键词描述了如何填充边框的边C。值是stretch（自然会把图片拉伸），repeat（重复拼贴）和round（重复图像，但是会拉伸或者压扁图像，以便使图像刚好适应，不多不少）。round值当前不被Safari或者Chrome支持——可以用repeat替代——尽管在未来的版本中，可能会有变化。

下面是图14-14（顶部）的样式规则，它创建了奇特的边框图像。为了让它看起来简单，我没有使用供应商前缀。

```
.framed {
  ...
  background-color: #fec227; /* bright yellow-orange */
  border-color: #fec227; /* bright yellow-orange */
  border-style: solid;
  border-width: 55px;
  border-image: url(fancyframe.png) 55 stretch;
}
```

边框的源图像是fancyframe.png。由于四条边的切点都是相同的（55像素），我只需要指定一次55即可（注意，像素没必要使用单位）。最后，stretch关键词表示盒子的边会用图像的拉伸来填充。作为后备，我已经将背景色和边框色指定为黄色-橙色（从边框图像的中心开始）。IE图像将获得相同大小和颜色的盒子，但是没有框架图（详见“注意”）。

注意：不同类型的边框图像需要不同的后备方案，由于这个例子的边框很粗，所以我觉得最好的后备办法是用纯色来填充。

下面是添加了浏览器前缀的规则。如果你想使用边框图像，你的规则看起来应该就是这个样子。确保将标准的、没有前缀的属性放在最后。

```
.framed {
  ...
  background-color: #fec227; /* bright yellow-orange */
  border-color: #fec227; /* bright yellow-orange */
  border-style: solid;
  border-width: 55px;
  -moz-border-image: url(fancyframe.png) 55 stretch;
  -webkit-border-image: url(fancyframe.png) 55 stretch;
}
```

边框图像规约

根据CSS3规约，border-image是一个快捷属性，包含下面5个单独属性。

border-image-source
指定将要使用的图像的URL。

border-image-slice
给四条边提供测量值，以顺时针来列出。

border-image-width
以顺时针方向的TRouBLe方法，指定边框的宽度。

border-image-outset
指定图像超出边框的距离。

border-image-repeat
指定图像如何填充边（拉伸、重复拼贴或者围绕）。

现在，没有浏览器支持这些单独属性，所以你总是要使用快捷border-image属性来添加边框图。

此外，因为border-image-width会使浏览器出现问题，而且border-image-outset不被任何浏览器支持，我已经在border-image讨论中忽略了这些属性，只留下如图14-15所示的三个部分。


```

-o-border-image: url(fancyframe.png) 55 stretch;
border-image: url(fancyframe.png) 55 stretch;
}

```

下面是点线边框图的样式规则。它的不同在于顶部和侧边的不同宽度（因此需要border-image-slice和border-width两个属性值），而且我已经将重复性设置为round，使得可以用调整大小后的图来完美拼贴。注意，Webkit浏览器当前简单地把round当做repeat来处理。

```

.dotted {
background-color: white;
border-color: #0063a4;
border-style: dotted;
border-width: 20px 10px;
-moz-border-image: url(dotborder.png) 20 10 round;
-webkit-border-image: url(dotborder.png) 20 10 round;
-o-border-image: url(dotborder.png) 20 10 round;
border-image: url(dotborder.png) 20 10 round;
}

```

注意：根据CSS3规约，浏览器默认不能渲染图像的中心，但是今天所有的浏览器都把图像的中心显示在元素中心。根据指定，中心区域会为边框stretch或repeat。如果你想给内容盒子的背景提供不同的背景色和图像，可以使用中心透明来创建border-image图像，这样背景就可以透过来显示了。

现在是你使用边框一试身手的时候了。练习14-2不只是一个练习，也会让你了解如何使用边框来添加视觉设计效果。

练习14-2 边框技巧

在本练习中，将对Jenware主页作一些有趣的边框应用。除了在网页内容周围添加细微的边框外，还将使用边框替代链接下划线来加强产品标题的效果。

1. 如果还没准备好，就在文本编辑器中打开jenware.html文档。开始，我们就简单地使用快捷border属性在products部分周围添加淡橙色（#FFBC53）double规则。给“products”div添加一个新的规则声明。

```

#products{
...
border:double #FFBC53
}

```

2. 接着，再给“testimonials”区域添加圆角。在IE 6~IE 8和其他老版本的浏览器上无法显示，但是这也并不会影响什么。

```

#testimonials{
...
border-radius:20px;
}

```

3. 为了更有趣，我们将在products部分的h3标题的两个方向添加装饰边框。我想让边框与文本的颜色相同，所以不必指定border-color属性。在“products”的div中查找h3元素的已有规则，并在标题上方添加1像素solid规则的声明。再添加另一个声明，在左方添加3像素solid规则的声明。最后，为防止文本冲撞左边框，在标题内容的左边添加一些填充（1em）。

```
#products h3 {
  font-size: 1em;
  text-transform: uppercase;
  color: #F26521;
  border-top: 1px solid;
  border-left: 3px solid;
  padding-left: 1em;
}
```

4. 我们要做的最后一件事是用装饰底边框替换标准的链接文本下划线。开始，通过将每个a元素的text-decoration属性值设为none来取消所有链接元素的下划线。在样式表的“link styles”中添加这条规则。

```
a {
  text-decoration: none;
}
```

5. 接着，通过以下规则，给链接的底边框添加1像素点线边框：

```
a {
  text-decoration: none;
  border-bottom: 1px dotted;
}
```

注意，因为我们想要所有的边框颜色和链接的一样，所以不必指定颜色。然而，如果你想在自己的网页上尝试，你可以很容易地改变底边的颜色和样式。

通常，给元素添加边框时，添加一些填充来防止冲撞是一个好办法。我们只给底边添加一些填充，如下所示：

```
a {
  text-decoration: none;
  border-bottom: 1px dotted;
  padding-bottom: .1em;
}
```

网页现在看起来如何可以看看图14-16。

图14-16：边框添加规则的效果

[More about custom barware...](#)

BABYWARE



Tired of sickly-sweet? These onesies with oversized fare. Monogrammed pants and baby shoes are also available. and 8 edging colors. Allow 2 to 4 weeks for delivery.

[More about our onesies...](#)

TESTIMONIALS

"The Jenware onesies were the hit of the baby show. I got cookie-cutter clothes from the department store."

"Thanks for the cool Jenware barware. I couldn't wait to know where I got them so I send them to your site."

CSS提示

浏览器默认空白边

你可能已经注意到，空白被自动添加到标题、段落和其他块元素的周围。这是浏览器工作时默认的风格表，应用空白边到这些元素的上面和下面。

最好记住，浏览器在后台应用自己的空白边和填充。这些值会一直起作用，直到你用自己的风格表来覆盖它们。

如果你在设计时遇到了不是你添加的神秘的空白，那就是浏览器做的。一个解决方案是将所有元素的填充和空白边都重置为0，将在第18章中讨论。

空白边

元素剩下的最后一个组件是空白边（margin），它是添加在边框外面数量可选的空间。空白边保证了元素间互不冲撞或不冲撞浏览器窗口的边线。甚至可以用空白边来为内容的另一列创造空间（工作原理参见第16章）。因此，空白边是基于CSS网页布局的一个重要工具。

专向和快捷的margin属性工作方式类似于我们见过的padding属性，然而，我们还需要知道margin的一些特殊行为。

margin-top, margin-right, margin-bottom, margin-left

属性值： 长度值 | 百分比 | auto | inherit

默认值： auto

适用对象： 所有元素

是否可继承： 否

margin

属性值： 长度计量值 | 百分比值 | auto | inherit

默认值： auto

适用对象： 除了有表格显示类型的元素（并不是table-caption、table和inline-table）的所有元素

是否可继承： 否

margin属性可直接使用。可以给元素的每一个方向指定一定数量的空白边，或者使用margin属性来一次指定所有方向。

速记margin属性与padding快捷属性的工作方式相同。当提供四个值的时候，它们依顺时针（上、右、下、左）应用于元素的各个方向。如果提供三个值，中间的值将应用于左右两个方向。当提供两个值时，第一个用于上下边，第二个用于左右边。最后，如果只提供一个值，将应用于元素的四条边。

对于大多数网页计量，cm、像素和百分比都是指定空白边最常用的方式。但要注意的是，如果指定百分比值，会根据父元素的宽度来计算。如果父元素的宽度改变，子元素四条边的margin值也会改变（padding也一样）。关键字auto允许浏览器使用一定量的margin来适应或填满可用的空间。

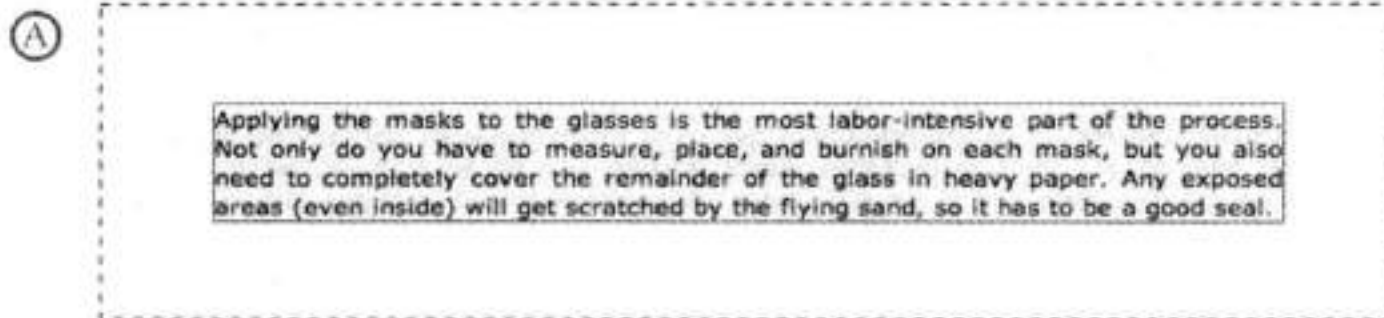
图14-17 显示了下面margin例子的结果。注意，添加一个淡淡的点划线规则只是为了清晰地表示空白边的外边缘，其实它们不会在真实网页中显示。

- ```

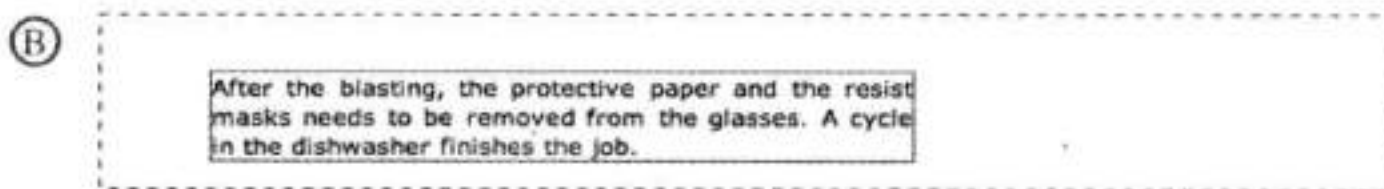
① p#A {
 margin: 4em;
 border: 1px solid red;
 background: #FCF2BE;
}
② p#B {
 margin-top: 2em;
 margin-right: 250px;
 margin-bottom: 1em;
 margin-left: 4em;
 border: 1px solid red;
 background: #FCF2BE;
}
③ body {
 margin: 0 15%;
 border: 1px solid red;
 background-color: #;
}

```

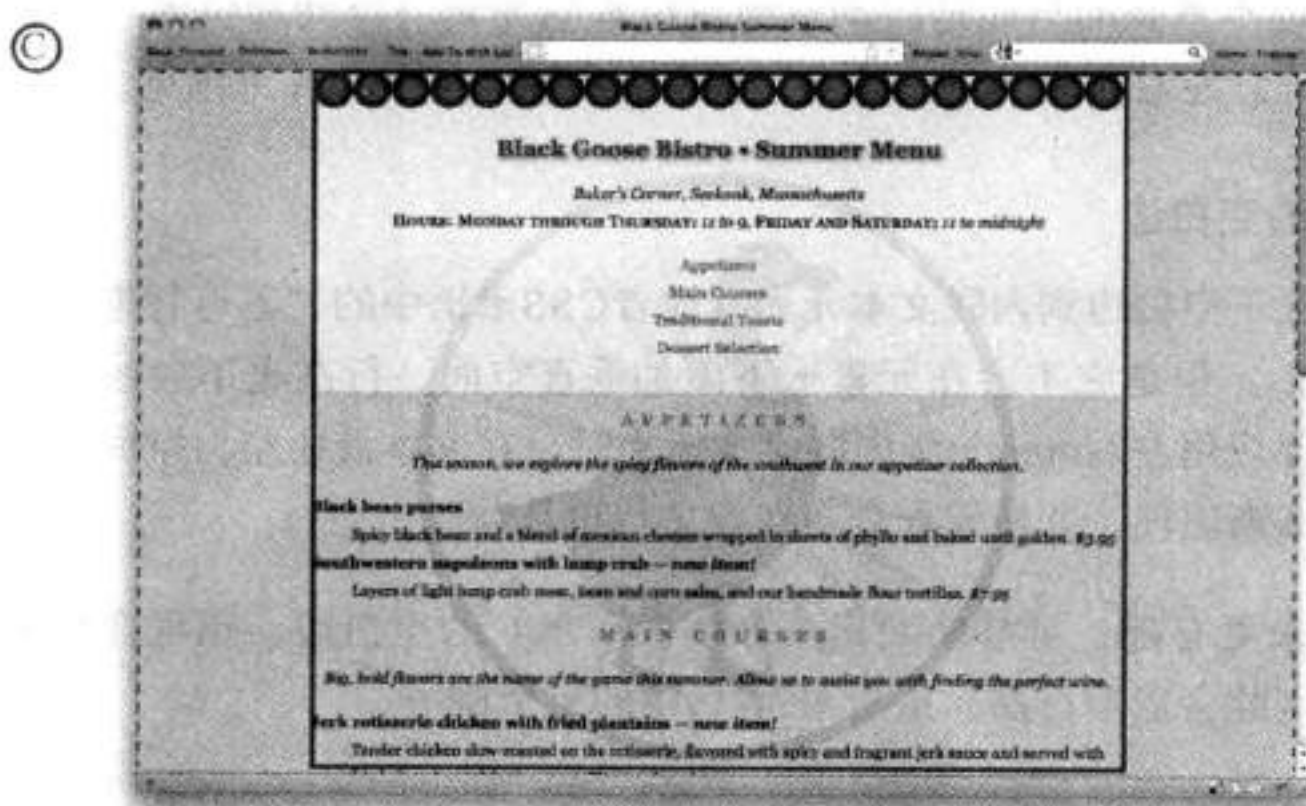
注意：给body元素添加margin属性，在网页内容和浏览器窗口边框之间添加空间。



margin: 4em;



margin-top: 2em;  
margin-right: 250px;  
margin-bottom: 1em;  
margin-left: 4em;



body: {margin: 0 15%}

给body元素添加margin属性，可以在网页内容和浏览器窗口边框之间添加空间。红边框显示了body元素的边界（没有使用填充）

图14-17：应用margin属性到body元素和其他独立元素



## 扩展阅读

## 重叠空白边

当元素间和元素周围的空白不正确时，往往是重叠空白边引起的。这里有一些深入讲解重叠空白边行为的文章。它们也许可以帮你明白布局的后台行为。

- Andy Budd的“No Margin for Error” ([www.andybudd.com/archives/2003/11/no\\_margin\\_for\\_error/](http://www.andybudd.com/archives/2003/11/no_margin_for_error/))
- Eric Meyer的“Uncollapsing Margins” ([www.complexspiral.com/publications/uncollapsingmargins/](http://www.complexspiral.com/publications/uncollapsingmargins/))

## 空白边行为

虽然在HTML元素周围应用margin属性的规则编写起来很容易，但是熟悉空白边的行为还是非常重要的。

## 重叠空白边

要知道空白边行为影响最大的是上下空白边与邻近元素的重叠。这意味着，不是累加，相邻空白重叠，只有最大值才被应用。

以图14-7中两个段落为例，如果上下的元素有一个4em的上空白边，而下面的元素的上空白边为2em，元素间的最终空白边没有达到6em，而是4em，也就是最大的指定值，如图14-18所示。

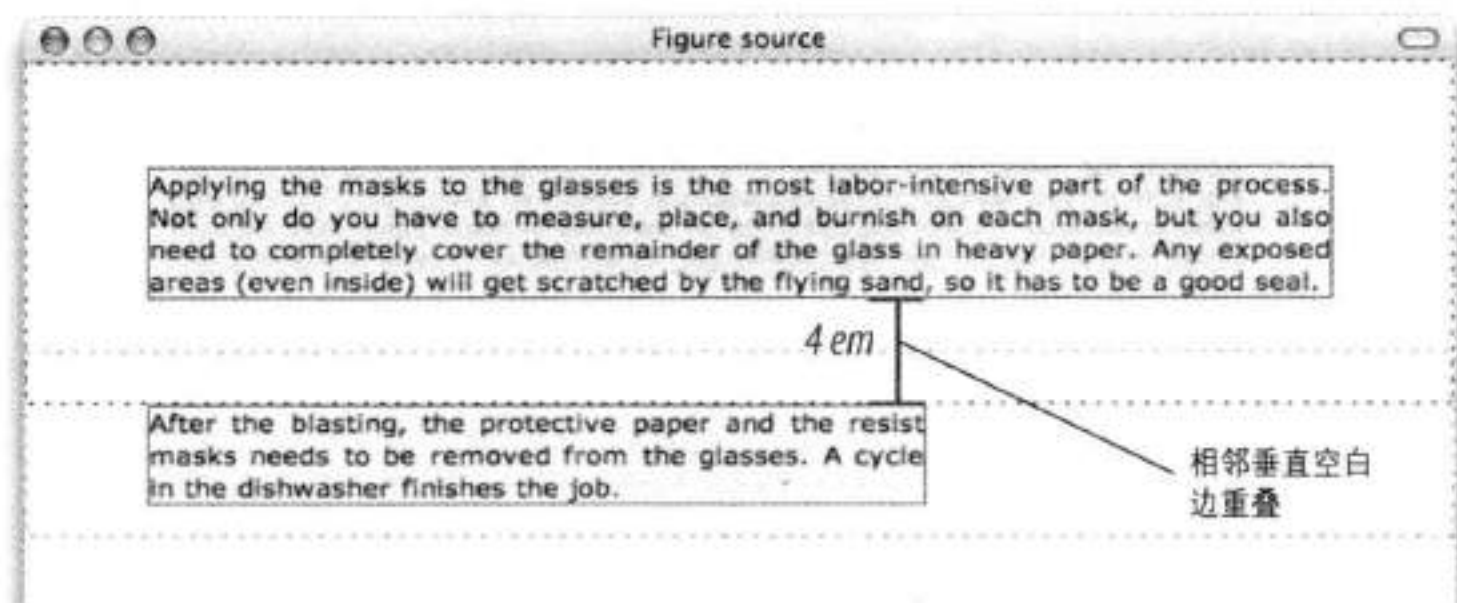


图14-18：相邻元素的垂直空白边重叠，以便应用较大值

上下空白边不重叠的只能是浮动或绝对位置的元素（浮动和位置将在第15章学习）。左右方向的空白边不会重叠，所以它们美观并可预见。

## 内联元素的空白边

可以应用上下空白边到内联文本元素（或者CSS术语中的“不可替换的内联元素”），但是它不会在元素上下添加垂直空间，行高也不会改变。然而，当你应用左右margin到内联文本元素时，在文本流中空白空间会在文本前后清晰出现，即使元素分布在很多行也是如此。

为了看起来更有趣，可替换元素（如图像所示）的空白边，出现在所有方向上，因此会影响行高。请看图14-19中的每个例子。

```
em { margin: 2em;}
```

不可替代的（文本）元素上只有水平空白边会显示

Applying the masks to the glasses is the most labor-intensive part of the process. Not only do you have to measure, place, and burnish on each mask, but you also need to completely cover the remainder of the glass in heavy paper. Any exposed areas (even inside) will get scratched by the flying sand, so it has to be a good seal.

```
img { margin: 2em;}
```

可替代元素上所有边的空白都会显示，比如图像

Applying the masks to the glasses is the most labor-intensive part of the process. Not only do you have to measure, place, and burnish on each



mask, but you also need to completely cover the remainder of the glass in heavy paper. Any exposed areas (even inside) will get scratched by the flying sand, so it has to be a good seal.

图14-19：应用于内联文本和图像元素的空白边

## 负空白边

值得注意的是，可以给margin属性指定负数值。当应用负空白边时，内容、填充和边框将向相反的方向移动，这都是负的margin属性值引起的。

用例子可以更清楚地说明。图14-20显示了相邻的两个段落，为了显示它们的边缘，我们使用了不同颜色的边。在上面的视图中，给上段添加4em的下空白边，这样就把下段推下4em。如果指定这个值为-4em，那么下段将会上移4em，并且负空白边将与上面的元素交迭。

```
p.top { margin-bottom: 4em;}
```

把下段推下4em

Applying the masks to the glasses is the most labor-intensive part of the process. Not only do you have to measure, place, and burnish on each mask, but you also need to completely cover the remainder of the glass in heavy paper. Any exposed areas (even inside) will get scratched by the flying sand, so it has to be a good seal.

Applying the masks to the glasses is the most labor-intensive part of the process. Not only do you have to measure, place, and burnish on each mask, but you also need dot to completely cover the remainder of the glass in heavy paper. Any exposed areas (even inside) will get scratched by the flying sand, so it has to be a good seal.

```
p.top { margin-bottom: -4em;}
```

下段将上移4em

Applying the masks to the glasses is the most labor-intensive part of the process. Not only do you have to measure, place, and burnish on each mask, but you also need to completely cover the remainder of the glass in heavy paper. Any exposed areas (even inside) will get scratched by the flying sand, so it has to be a good seal.

图14-20：使用负空白边

这看起来有点奇怪，事实上，你也不会像图中一样使文本块交迭。这里的重点是，你可以使用正负值的空白边来移动网页中的元素。这是许多CSS布局技术的基础。



现在，将在练习14-3中，使用margin属性来给Jenware主页的各部分之间添加一些空白。

### 练习14-3 在元素周围添加空白边

先在文本编辑器中打开jenware.html文档，在这里将设置一些空白边。练习过程中，我们会调整整个文档的空白，从上到下地改进每一部分。

1. body元素的空白往往会设置为0，这样就可以清除浏览器的默认空白设置，并且为我们自己设置页面元素的空白做好准备。

```
body {
 ...
 margin:0;
}
```

保存文件，并在浏览器中查看。我喜欢紫色的导航栏从浏览器的一边拉伸到另一边，但是我想我们需要改进其他内容区域。

2. 现在在网页的#intro div上方添加2em空白，下方添加1em的空白。我还想去掉logo和标语之间的空白，所以将h1的底部空白边设置为0，h2的顶部空白边设置为-10px，从而将标语上移到靠近logo的位置。最后，给介绍段落(p)周围添加1em的空白。

```
#intro {
 ...
 margin: 2em 0 1em;
}
#intro h1 {
 margin-bottom: 0;
}
#intro h2 {
 ...
 margin-top: -10px;
}
#intro p {
 ...
 margin: 1em;
}
```

3. 给#products四周添加1em的空白边。

```
#products {
 ...
 margin: 1em;
}
```

注意：当值为0时，不需要指定单位

4. 现在给#products的标题h3上方添加2.5em的空白。到现在，我想你可以独立完成了，但是为了确保无误，下面是给“products”的h3添加的新声明。你可以试试不同数量的空白，看看你最喜欢的是什么样的。

```
#products h3 {
 ...
 margin-top: 2.5em;
}
```

5. 最后，我们会给Testimonials盒子上方添加1em的空白，在左右边缘添加10%的空白，来使其分离出来。这一步，你自己试试看。
6. 保存并在浏览器中查看，现在看起来应该如图14-21所示的那样。现在的设计并不是最漂亮的，尤其是当浏览器窗口比较宽的时候。然而，如果你将浏览器窗口调整得很窄，你会发现，没有比网页设计在小屏幕上看起来更糟糕的了。（将它视为第18

章的铺垫。）网页最终的样式表可以从附录A中得到。

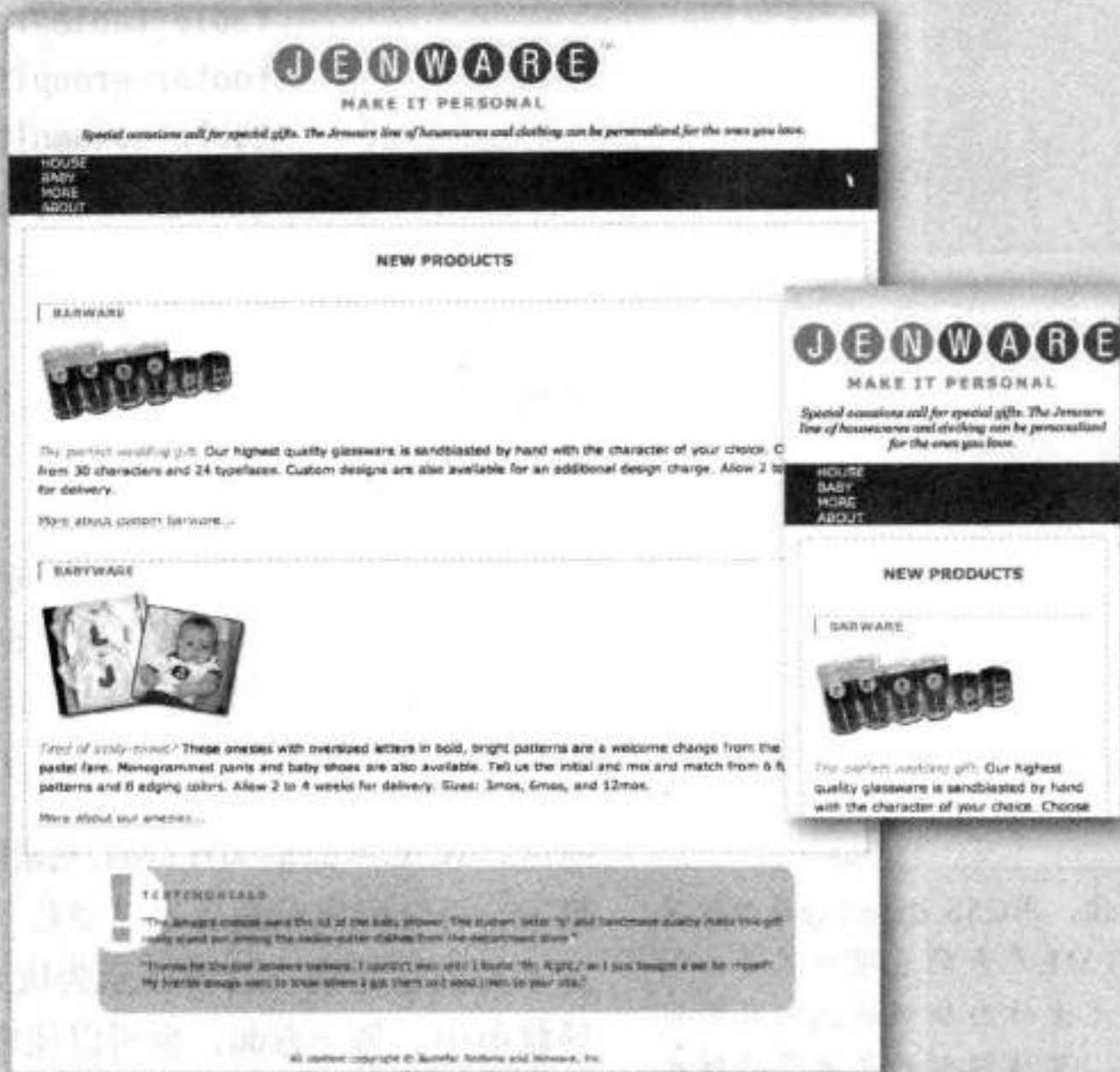


图14-21：添加padding、border和margin后的jenware主页

对填充、边框和空白边理解透彻是精通CSS布局的第一步。在第15章中，我们将学习网页中浮动和定位元素的属性。甚至将jenware网页转换为两列布局。但是在继续向前之前，我们还需学习更多的属性。

## 分配显示角色

只要讨论盒子和CSS布局模型，就需要介绍display属性。你应该已经熟悉了块和内联元素的显示行为。然而，并不是所有的XML语言都给元素分配默认显示行为（CSS规范中的术语是显示角色（display roles））。因此，需要创建display来允许作者指定元素在布局中如何表现。



## display

|        |                                                                                                                                                                                                                                                                                                             |
|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 属性值:   | <code>inline block list-item inline-block table inline-table table-row-group table-header-group table-footer-group table-row table-column-group table-column table-cell table-caption none</code><br>CSS3新引入了: <code>run-in compact ruby ruby-base ruby-text ruby-base-container ruby-text-container</code> |
| 默认值:   | 内联                                                                                                                                                                                                                                                                                                          |
| 适用对象:  | 所有元素                                                                                                                                                                                                                                                                                                        |
| 是否可继承: | 是                                                                                                                                                                                                                                                                                                           |

`display`属性定义了元素在布局中生成的元素盒子的类型。除了熟悉的`inline`和`block`显示角色，我们也可以使元素像列表项显示或以表格的不同部分来显示。就像你从属性值列表中看到的，一个元素可以有多个角色，但是日常使用中其实只是用了为数不多的角色。

通常，W3C不提倡为HTML元素随机再分配显示角色。然后，在特定场景下，再分配显示角色也已经是司空见惯的。例如，我们往往会使`li`元素显示为内联元素（通常显示为块元素），这样可以将一个列表装入水平导航条中。另一方面，也可以使内嵌元素`a`（锚）显示为块元素从而可以指定宽和高。

```
ul.navigation li { display: inline; }
ul.navigation li a { display: block; }
```

`display`属性的另一个有用的值是`none`，它从普通流实体中移除内容。不像`visibility:hidden`（它会使元素不可见），`display:none`会移除内容，它所占据的空间也将关闭。

`display:none`另一个流行的用法是以特定方式，防止某些内容在文档中显示。例如，设置一个段落在文档打印时出现，但在显示器显示时并不是网页的一部分。

## 给盒子添加阴影

本节是元素盒子的最后一部分。在第12章中，你已经学到过`text-shadow`属性，它会给文本添加阴影。`box-shadow`属性（CSS3新引入的）会给整个可见元素盒子添加一个阴影（不包括空白）。

---

**警告：** 记住，用CSS `display`属性改变HTML元素的表现形式并不影响元素作为块级别或内联的定义。置块级别的元素于内联元素中是不符合HTML语法的，不管它的显示角色如何。

---

---

**警告：** 记住，具有其自身的`display`的内容设置为`none`时，依然会随文档下载而下载。将一些内容设置为`display:none`对于小屏幕设备来说，可以使页面短小，但是并不能减少下载数据和下载时间。

---

## box-shadow

属性值: “水平偏移量” “垂直偏移量” “模糊距离” “扩散距离” 色彩 inset|none

默认值: none

适用对象: 所有元素

是否可继承: 否

box-shadow属性的值看起来很像text-shadow的值: 指定水平和纵向偏移量、阴影模糊量以及色彩。对于盒子阴影来说, 还可以指定一个扩散值spread, 可以增加阴影的尺寸(也可以使用负值来缩小阴影尺寸)。默认情况下, 阴影颜色与元素的前景色相同, 如果指定色彩, 则会覆盖前景色。

图14-22显示了下面代码例子的效果。第一个A添加了一个简单的盒子阴影: 右侧6像素, 下方6像素, 没有模糊和扩散效果。第二个B添加了一个模糊值为5像素的阴影, 第三个C显示了扩散值为10像素的效果。盒子阴影总是应用于元素边框的外部(如果边框不显示, 也显示在实际边框的外部)。如果元素有透明或半透明的背景, 在元素后面的区域, 你将看不到盒子阴影。

- ① `-webkit-box-shadow: 6px 6px #666;`  
`-moz-box-shadow: 6px 6px #666;`  
`box-shadow: 6px 6px #666;`
- ② `-webkit-box-shadow: 6px 6px 5px #666;`  
`-moz-box-shadow: 6px 6px 5px #666;`  
`box-shadow: 6px 6px 5px #666; /* 5 pixel blur */`
- ③ `-webkit-box-shadow: 6px 6px 5px 10px #666;`  
`-moz-box-shadow: 6px 6px 5px 10px #666;`  
`box-shadow: 6px 6px 5px 10px #666; /* 5px blur, 10px spread */`

可以通过添加inset关键字来使阴影显示在可视元素盒子的内部。这样就好像这个元素被按下去一样(如图14-23所示)。

```
-webkit-box-shadow: inset 6px 6px 5px #666;
-moz-box-shadow: inset 6px 6px 5px #666;
box-shadow: inset 6px 6px 5px #666;
```

对于text-shadow来说, 可以通过提供逗号分隔的值列表, 对一个元素指定多个盒子阴影。其中第一个值的阴影出现在最上面, 其他的则都按顺序排在下面。

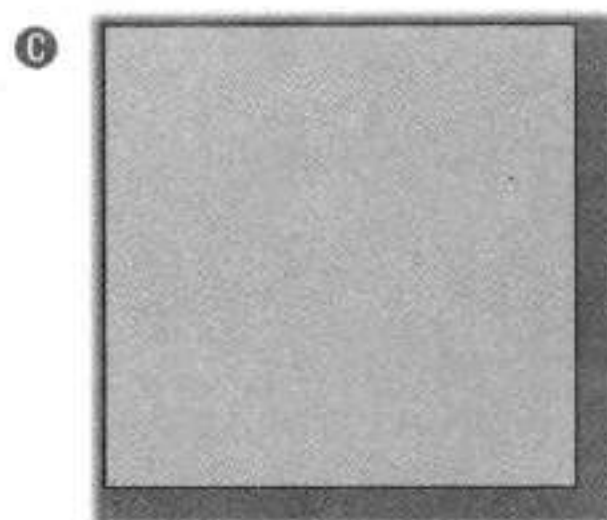
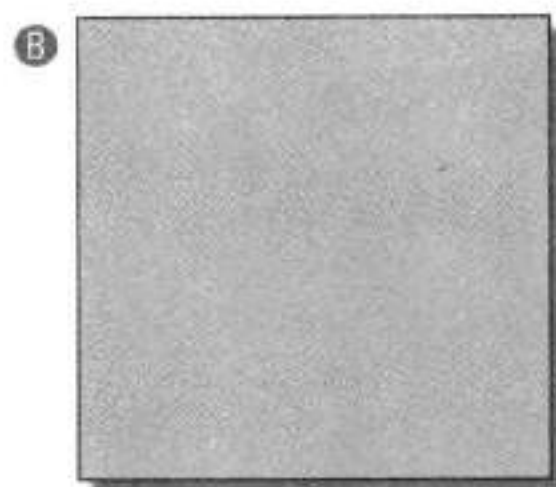
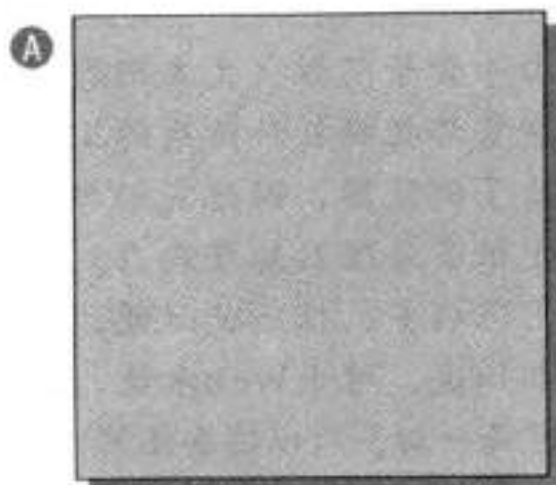


图14-22: 使用box-shadow属性给元素周围添加阴影

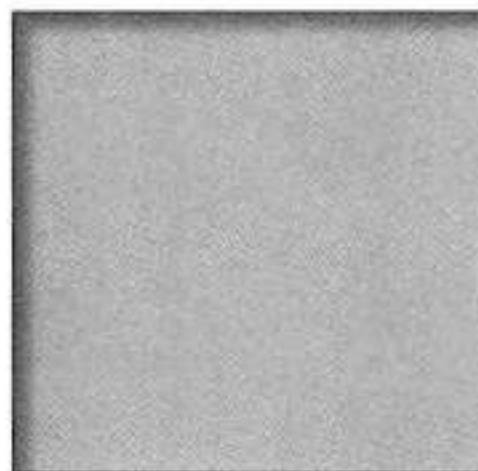


图14-23: 在元素盒子内部显示的内置盒子阴影



---

**警告：** 由于盒子阴影、文本阴影以及渐变都把解析和渲染的工作交给了浏览器，因此它们都要耗费很多处理器处理能力。你使用得越多，性能就越慢，我们都知道，对于Web来说，性能就是一切。所以慎重使用！

---

当今的浏览器都支持box-shadow属性，除了移动设备的Opera Mini。在写本书时，为了适应最新的Webkit浏览器（Safari和移动Safari、Chrome和Android），以及老版本的Firefox，最好如上面的例子中一样，使用供应商前缀。

IE 9及更高的版本支持标准的属性，但是IE 6~IE 8不支持。我的看法是，就算使用老版本浏览器的用户看不到漂亮的阴影，那也并不是世界末日。如果一定要在老版本的浏览器中显示阴影，Zoltan的文章“[How to Simulate CSS3 box-shadow in IE6-8 without JavaScript](#)”和Hawryluk的文章“[Du Lac](#)”有详细解释（[www.useragentman.com/blog/2011/08/24/how-to-simulate-css3-box-shadow-in-ie7-8-without-javascript/](#)）。

## 自我测验

现在，你应该已经对元素盒子和如何操作内外空间有了很好的感觉。这些是你将来完成CSS布局所需要的原始工具。在第15章中，将开始在网页中移动盒子，但首先，请完成下面测验中的练习，编写一些用于填充、边框和空白边的规则。

本测验中，你的任务是编写声明，来产生图14-24中每个例子的效果。这里看到的所有段落都共享一个规则，也就是设置每段的尺寸和背景色的规则。你只需要提供盒子相关属性声明即可。与前面一样，答案见附录A。

一些有用的提示：空白边的外边缘以蓝色点划线表示。所有需要的计量值都用红色表示。边框使用17个标准颜色名之一。

Lorem ipsum dolor sit amet,  
 consectetur adipiscing elit.  
 Praesent porttitor venenatis  
 mi. Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel arcu.

这个练习中所有例子的样式都在这里。它们都共享下面所列出的属性。

```

p { background-color: #C2F670;
width: 200px;
height: 200px;}

```

Lorem ipsum dolor sit amet,  
 consectetur adipiscing elit.  
 Praesent porttitor venenatis  
 mi. Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel arcu.

A

Lorem ipsum dolor sit  
 amet, consectetur  
 adipiscing elit. Praesent  
 porttitor venenatis mi.  
 Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel  
 arcu.

B

2 em  
 Lorem ipsum dolor sit amet,  
 consectetur adipiscing elit.  
 Praesent porttitor venenatis  
 mi. Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel arcu. 2 em  
 2 em

C

4 pixels  
 2 em  
 Lorem ipsum dolor sit amet,  
 consectetur adipiscing elit.  
 Praesent porttitor venenatis  
 mi. Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel arcu. 2 em  
 2 em

D

4 pixels  
 2 em  
 Lorem ipsum dolor sit amet,  
 consectetur adipiscing elit.  
 Praesent porttitor venenatis  
 mi. Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel arcu. 2 em  
 2 em

E

4 pixels  
 1 em  
 1 em  
 Lorem ipsum dolor sit amet,  
 consectetur adipiscing elit.  
 Praesent porttitor venenatis  
 mi. Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel arcu. 1 em  
 6 em  
 6 em  
 1 em  
 1 em

F

2 pixels  
 1 em  
 Lorem ipsum dolor sit amet,  
 consectetur adipiscing elit.  
 Praesent porttitor venenatis  
 mi. Nunc semper, orci a  
 adipiscing tempus, magna  
 nulla varius nisi, imperdiet  
 fermentum nisi erat vel arcu. 1 em  
 50 pixels  
 50 pixels

G

图14-24: 编写这些例子的声明



## CSS回顾：基本盒子属性

| 属性                         | 介绍                   |
|----------------------------|----------------------|
| border                     | 结合了所有边框属性的快捷属性       |
| border-top                 | 元素每个方向的边框属性          |
| border-right               |                      |
| border-bottom              |                      |
| border-left                |                      |
| border-color               |                      |
| border-top-color           | 元素的每个方向各自指定颜色        |
| border-right-color         |                      |
| border-bottom-color        |                      |
| border-left-color          |                      |
| border-image (CSS3)        | 在边框区域内添加一个图像         |
| border-radius (CSS3)       | 对可视元素盒子提供圆角的快捷属性     |
| border-top-left-radius     | 为每个单独的角指定弯曲半径        |
| border-top-right-radius    |                      |
| border-bottom-right-radius |                      |
| border-bottom-left-radius  |                      |
| border-style               | 指定所有边框样式的快捷属性        |
| border-top-style           | 元素的每个方向各自指定边框样式      |
| border-right-style         |                      |
| border-bottom-style        |                      |
| border-left-style          |                      |
| border-width               | 指定所有边框宽度的快捷属性        |
| border-top-width           | 元素的每个方向各自指定宽度        |
| border-right-width         |                      |
| border-bottom-width        |                      |
| border-left-width          |                      |
| box-sizing                 | 说明宽度和高度应用于内容盒子还是边框盒子 |
| box-shadow (CSS3)          | 给可视元素盒子周围添加阴影        |
| display                    | 定义元素产生的元素盒子的类型       |
| height                     | 指定元素内容区域的高度          |
| margin                     | 指定元素所有空白边的快捷属性       |

| 属性                                                             | 介绍                 |
|----------------------------------------------------------------|--------------------|
| margin-top<br>margin-right<br>margin-bottom<br>margin-left     | 元素的每个方向各自指定空白边     |
| max-height                                                     | 指定元素的最大高度          |
| max-width                                                      | 指定元素的最大宽度          |
| min-height                                                     | 指定元素的最小高度          |
| min-width                                                      | 指定元素的最小宽度          |
| overflow                                                       | 内容不适合于内容区域时如何处理    |
| padding                                                        | 指定内容区域与边框之间空白的快捷属性 |
| padding-top<br>padding-right<br>padding-bottom<br>padding-left | 指定各个方向的填充值         |
| width                                                          | 指定元素内容区域的宽度        |





# 第15章 浮动与定位

到目前，你已经学习了不少CSS属性来让你改变文本元素的外观和它们产生的盒子。但是到现在依然局限于仅在文档流中修饰元素。

本章将介绍浮动和定位，用于打破流并在网页上安排元素的CSS方法。浮动一个元素指左右移动元素，并允许后面的文本环绕它。定位是指以像素精读来指定元素在网页上的位置。

首先，学习浮动和定位对应的属性，这样将对CSS布局工具如何工作有良好的体会。在第16章中将拓宽视野，看这些属性如何用来创建普通的多列网页布局。

开始移动元素之前，需要确认是否理解它们在普通流中是如何表现的。

## 普通流

在前几章中学习了普通流，但有必要再复习一遍。在CSS布局模型中，文本元素依源代码中的顺序从上到下排列，然后从左到右（在从左到右来阅读的语言中）（注1）。块元素一个接一个，填满浏览器窗口或其他窗口元素的所有可用宽度。内联元素和文本字符一行接一行地填满块元素。

当窗口或窗口元素重置大小的时候，块元素随着新宽度增大或缩小，而内嵌内容会重新流动来适应宽度，如图15-1所示。

注1： 在从右到左来阅读的语言中，例如阿拉伯语和希伯来语，通常按照从上到下和从右到左的顺序。

### 本章内容

- 左右浮动的元素
- 清除的浮动元素
- 包含浮动元素
- 相对定位
- 绝对定位和包含块
- 固定定位



## 处理浏览器bug

这里讲讲浏览器的bug。本书使用的CSS方法都假定为可以工作的，但实际上，桌面或者移动浏览器有bug，这让布局成为令人头痛的问题。

过去，主要的问题就是IE 6。它有很多广为人知的问题，比如“Guillotine Bug”、“Peekaboo Bug”、“Double-Float Margin Bug”和“3-Pixel Gap Bug”等，这只是能叫出名字的一小部分。

这些bug都在IE后续的版本中解决了，而且也都不再是问题了。在写本书时，IE 6在美国仅有不到1%的人还在使用，所以大多数开发者不需要为此纠结。如果你确实需要支持老版本的IE，下面的站点应该可以解决你的问题：  
[positioniseverything.net/explorer.html](http://positioniseverything.net/explorer.html)。

在你开始开心之旅之前，不得不说我们其实处于比较糟糕的环境。不仅仅是因为在各种设备上有太多不同的浏览器，还因为bug越来越难以搞明白，而且很难简单地预测。

当一个属性确定有浏览器相关的问题时，我会明确指出。在你读到本书时，也许有问题的浏览器已经不复存在了。我能给你的最好建议是，尽可能多在不同设备和不同浏览器上测试你的设计，并且修复解决你发现的问题。

可以在Web上搜索特殊的属性或者浏览器问题，很多开发者已经发布了类似的问题甚至解决办法。你也可以检查CSS-discuss Wiki ([css-discuss.incutio.com](http://css-discuss.incutio.com))，那里除了有很多有用的CSS信息，也有很多浏览器的bug归档。

块按照它们在源文档中的顺序来布局

每个块都从新的一行开始

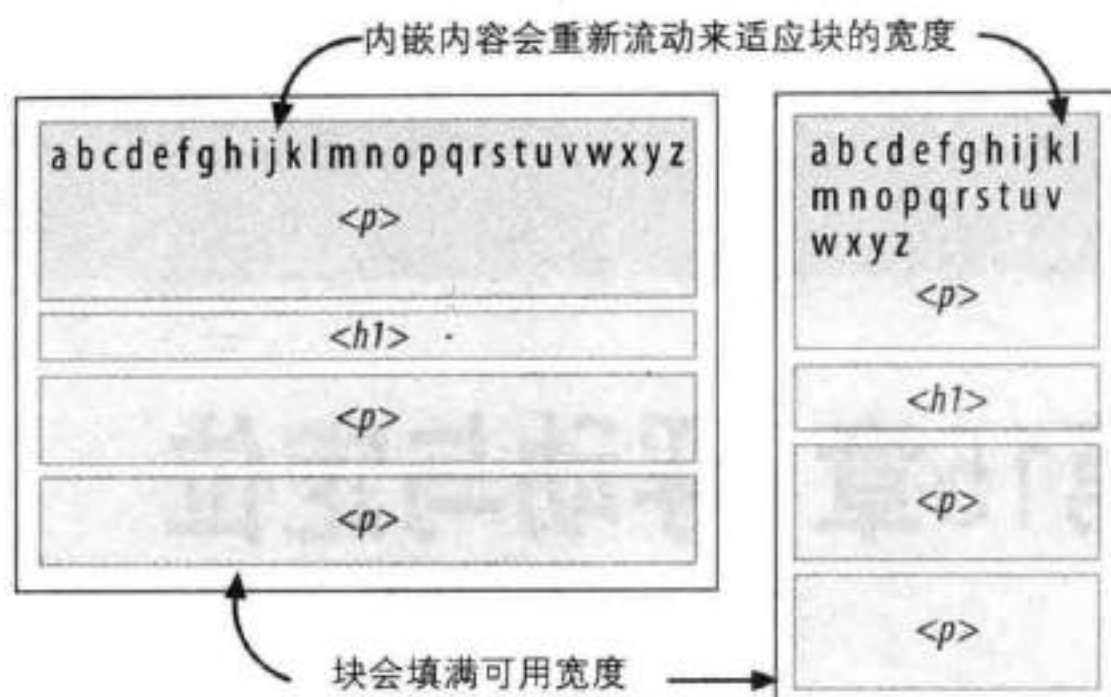


图15-1：又一个普通流行为的示例

普通流中的对象影响它们周围对象的布局。这正是你期望的网页中的行为——元素不交迭扎堆，它们互让空间。

我们以前都见过，但是本章中关注元素是在流中还是从流中移除了。浮动和定位通过不同的方法转换元素间的关系为普通流。首先看看浮动元素的特殊行为（或简称为“浮动”）。

## 浮动

简单地说，float属性尽可能远地向左或向右移动元素，允许后面的内容环绕它。它本质上并不是定位方案，却是唯一集成到包含大量有趣行为的CSS中的特性。浮动是现代基于CSS网页设计的主要工具之一，用于创建多列布局、列表中的导航工具条和无表格的类表格的排列。这些真是令人兴奋，让我们从float属性本身开始。

### Float

属性值： left|right|none|inherit

默认值： none

适用对象： 所有元素

是否可继承： 否

解释浮动的最好方法是给个范例。本例中，float属性应用于一个img元素，使之浮动到右边。图15-2展示了默认情况下，段及其包含的图像如何呈现（上图），以及应用float属性后的显示（下图）。

### 标记文本

```
<p>They went down, down,...</p>
```

## 样式表

```
img {
 float: right;
}
p {
 padding: 15px;
 background-color: #FFF799;
 border: 2px solid #6C4788;
}
```

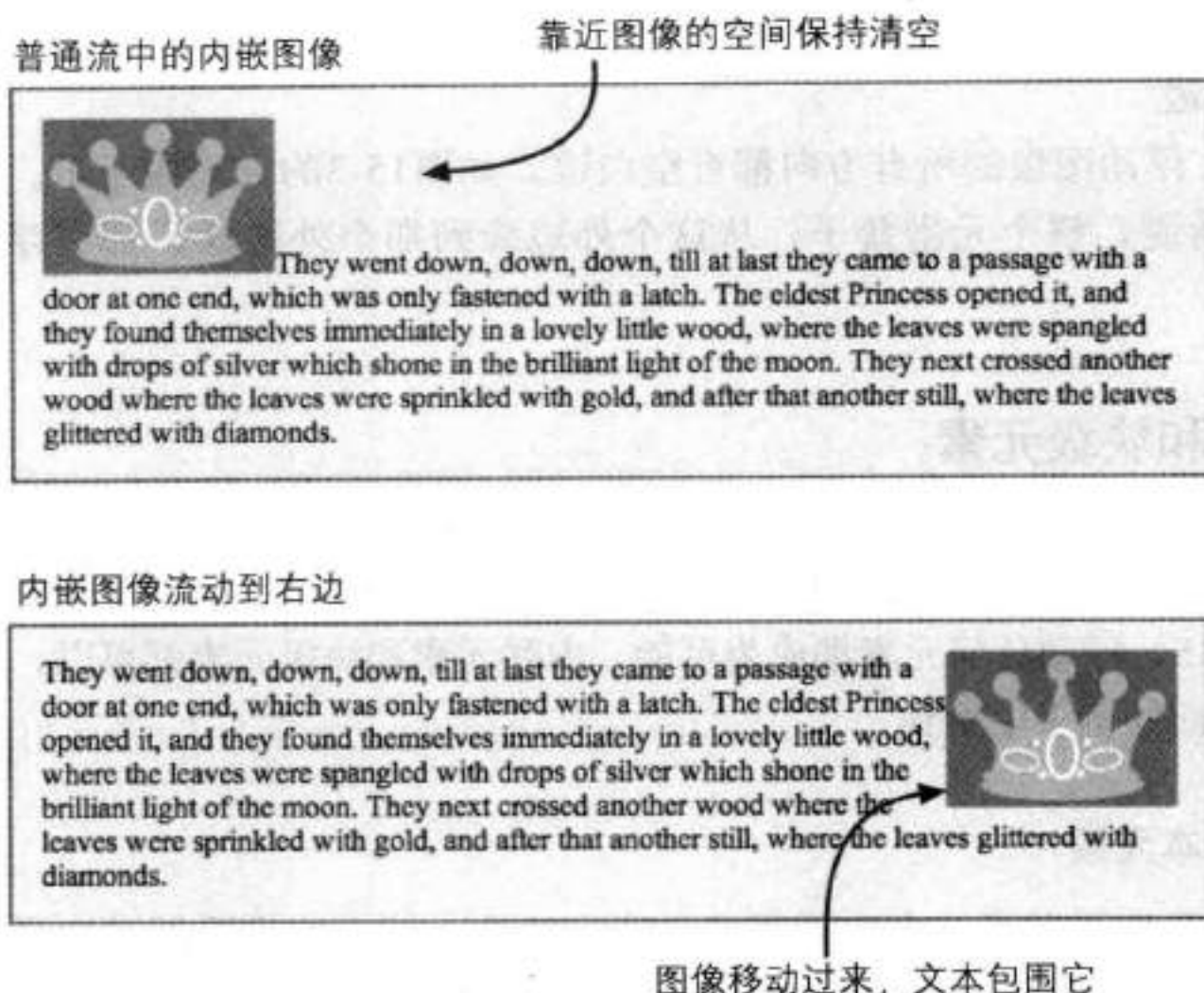


图15-2：普通流（上图）以及应用float元素后（下图）各自图像的布局

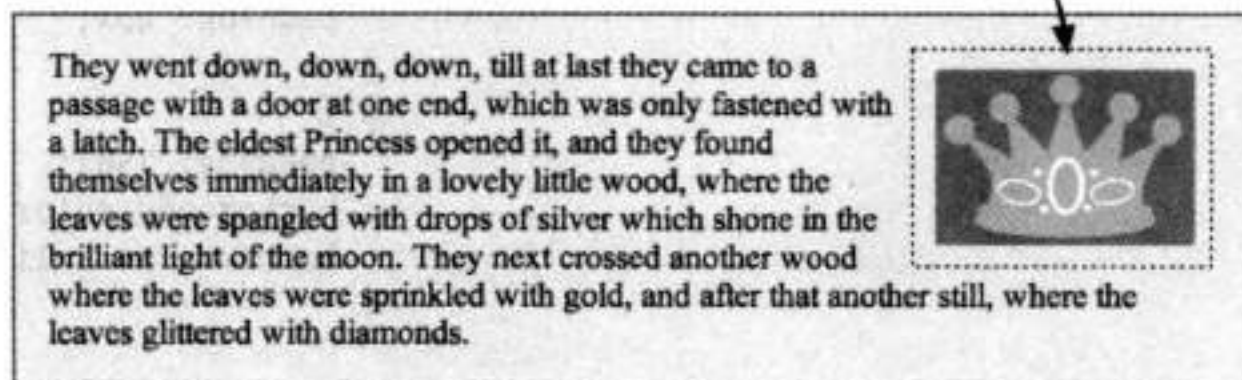
很漂亮的效果……我们减少了网页上空间的浪费，让文字向右边堆积到图像。你觉得在图像和环绕文字间加些空间如何？如果你猜到“添加空白边”，那就完全正确。使用margin属性给图像的所有方向加10像素的空间（如图15-3所示）。你会看到所有的盒子属性如何在网页布局上一起工作。

```
img {
 float: right;
 margin: 10px;
}
```

图15-3：浮动图像周围添加10像素的空白边

表示外部空白边缘  
(在实际中，虚线框不会显示)

浮动元素的一些关键行为出现在上面两幅图中：





### 浮动元素像水流中的岛。

首先，你可以看到图像从普通流的位置中移除，同时继续影响周围内容。接下来段落的文本重新流动来给浮动的img元素提供空间。一个流行的类比是把浮动元素比作水流中的岛。它们没有流动，但水从它们周围流过。这个行为是浮动元素特有的。

### 浮动元素在包含元素（containing element）的内容区域

注意到这点也很重要，浮动图像位于包含它的段落的内容区域（内边缘）中。它不能扩展到段落的填充区域。

### 也包括空白边

另外，浮动图像的所有方向都有空白边，如图15-3的点划线所示。换句话说，整个元素盒子，从这个外边缘到那个外边缘，都在浮动。

## 浮动内联和块级元素

这些都是基础。让我们看更多的例子，了解其他的浮动行为。在样式表之前，你唯一需要浮动的是图形，可以通过使用过时的align属性来完成。有了CSS，浮动任何元素都成为可能，内联元素和块级元素都可以，我们会在下面的例子中看到。

### 浮动内嵌文本元素

在上例中，我们浮动了一个内联图像元素。这次，让我们看看浮动内联文本（不可替换的）元素时会发生什么（如图15-4所示）。

### 标记文本

```
<p>Disclaimer: The existence of silver,
gold, and diamond trees is not confirmed.They went down,
down, down, till at last they came to a passage... </p>
```

### 样式表

```
span.disclaimer {
 float: right;
 margin: 10px;
 width: 200px;
 color: #FFF;
 background-color: #9d080d;
 padding: 4px;
}
p {
 padding: 15px;
 background-color: #FFF799;
 border: 2px solid #6C4788;
}
```

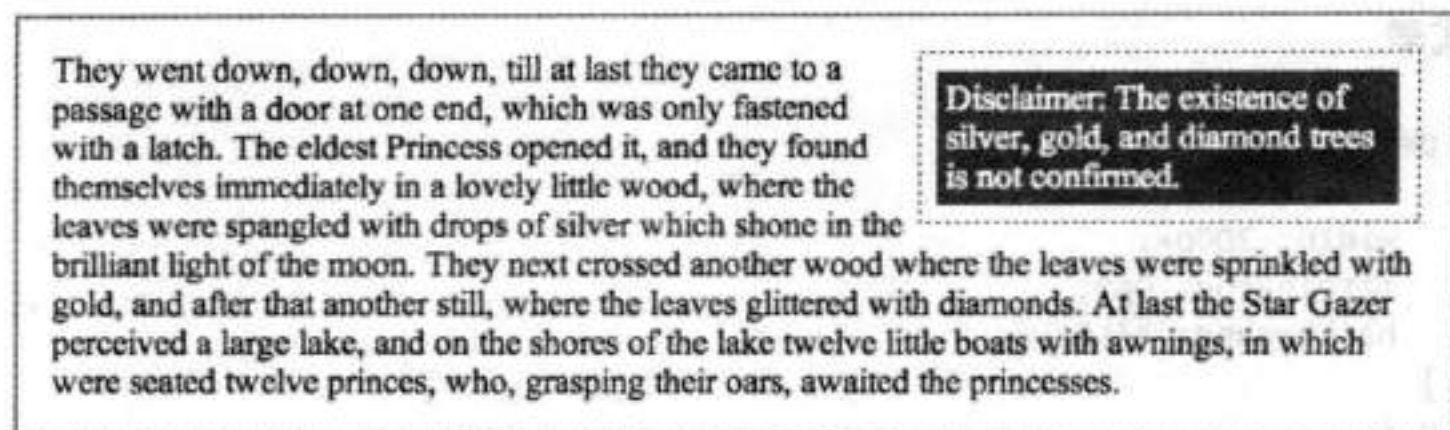


图15-4：浮动内联文本（不可替换的）元素

从这些事物的外观来看，它和浮动图像的行为一样，这正是我们所期望的。但是还有一些细微的地方需要指出。

### 总是提供浮动文本元素的宽度

首先，要注意，浮动span元素的样式规则包括width属性。事实上，必须指定浮动文本元素的宽度，否则，盒子的内容区域将扩张到最大可能的宽度（或者，在某些浏览器中，它会收缩到最小可能的宽度）。图像有一个继承来的宽度，所以在上例中不需要指定宽度（虽然可以指定）。

必须指定浮动文本元素的宽度。

### 浮动内联元素表现为块元素

注意，浮动的span文本的四周都有空白边，即使内联元素的上下空白边通常不显示（如第14章的图14-18）。这是因为所有的浮动元素都表现得像块元素。一旦浮动内联元素，它就遵循块级别元素的显示规则，所有四个方向的空白边都会呈现出来。

### 浮动元素的空白边不会重叠

在普通流中，紧挨着的顶部和底部空白边会重叠，但对于浮动元素来说，所有边的空白都按指定值来保留。

### 浮动块元素

看看在普通流中浮动块会发生什么。本例中，整段元素都浮动到左边（如图15-5所示）。

### 标记文本

```
<p>ONCE upon a time....</p>
<p id="float">As he had a white skin, blue eyes,...</p>
<p>the fact was he thought them very ugly...</p>
```



## 样式表

```

p#float {
 float: left;
 width: 200px;
 margin-top: 0px;
 background: #A5D3DE;
}
p {
 border: 1px solid red;
}

```

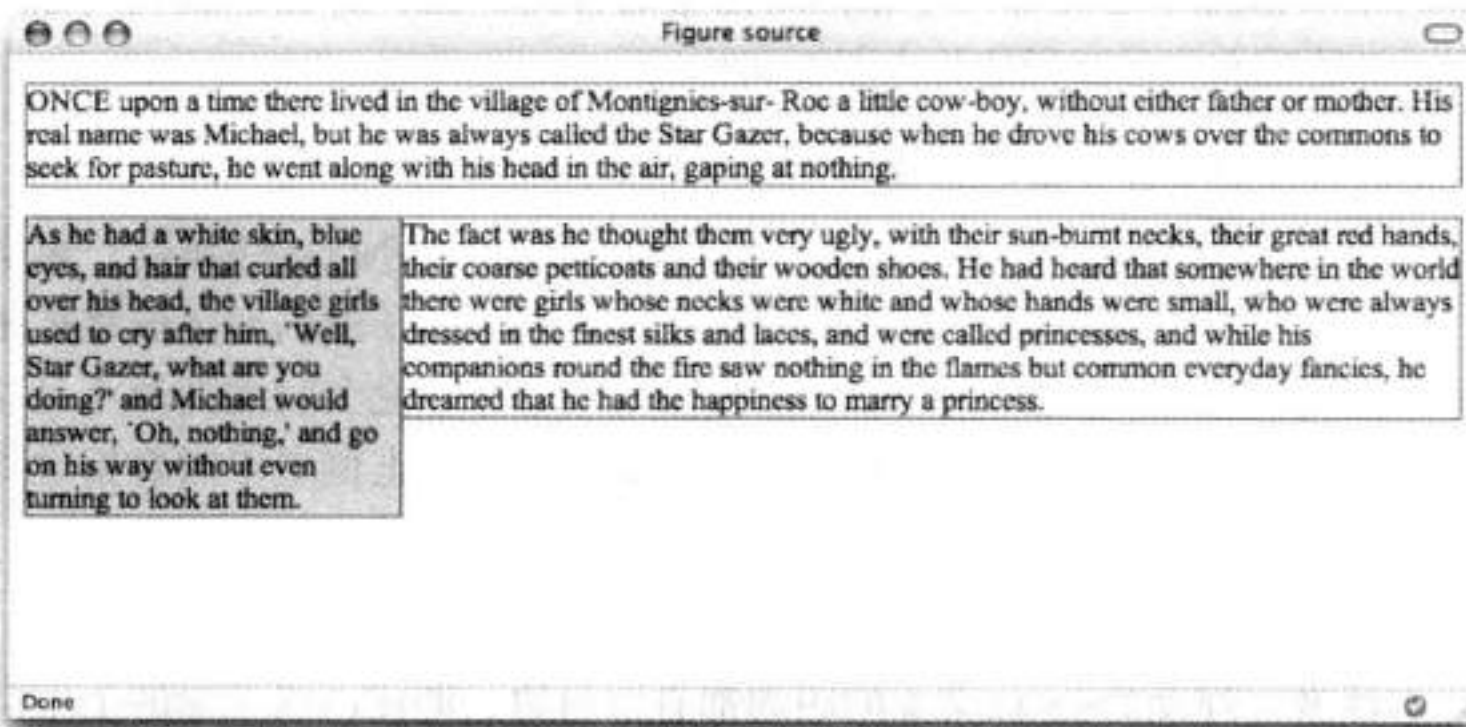


图15-5：浮动块级别元素

我添加了一条规则，使所有p元素周围为红色，以显示它们的边界。另外，设置上空白边为0来覆盖浏览器对段落设置的默认空白边。这就允许浮动段落沿着下一段的上边排列。

正如我们所看到的图像中，段落移动到边界（这次左边），下面的内容环绕它，即使文本块平常是一个接一个。本例中我要指出两件事：

## 必须提供浮动块元素的宽度

如果不能提供宽度值，那么浮动块width属性值将设置为auto，这样会填满浏览器窗口或容器元素的所有宽度。全宽度浮动盒子没有多大意义，因为我们希望文本可以环绕浮动元素，而不是从元素下面开始。

## 浮动元素不会高过其在源码中的参照物

相对于源码中环绕它的文本，浮动块会浮动到左边或右边。在流中，它在所有之前的块元素下面（它被这些块阻塞了）。这意味着不能把元素上浮到网页的上角，即使它最近的祖先是body元素。如果希望元素从网页最上面开始，那么它必须在源码中最先出现。

**注意：** CSS方法中用于摆放网页元素的绝对定位方法不考虑它们在源码中的显示情况。下面将学习绝对定位。

## 清除浮动元素

如果你想到处浮动元素，就需要知道如何取消文本环绕，恢复平常布局。这可以通过清除你想从浮动元素下面开始排列的元素来完成。应用 `clear` 属性到一个元素来防止元素紧接着浮动元素出现，并强制它从浮动元素下面的“清除”空间开始。

`clear`

属性值: `left|right|both|none|inherit`

默认值: `none`

适用对象: 仅块级元素

是否可继承: 否

记住，将 `clear` 属性应用到在浮动元素下面从左边开始的元素，而不是浮动元素自身。`left` 值使元素在任何元素下浮动到左边，类似地，右值使元素清除所包含的块右边的所有浮动元素。如果有多个浮动元素，并且你希望元素从所有这些元素下面开始布局，可以使用 `both` 值来清除两边的浮动元素。

本例中，`clear` 属性用来设置 `h2` 元素从左浮动元素下面开始布局。图15-6展示了 `h2` 标题如何从浮动元素下面的下一个可用的清除边开始布局。

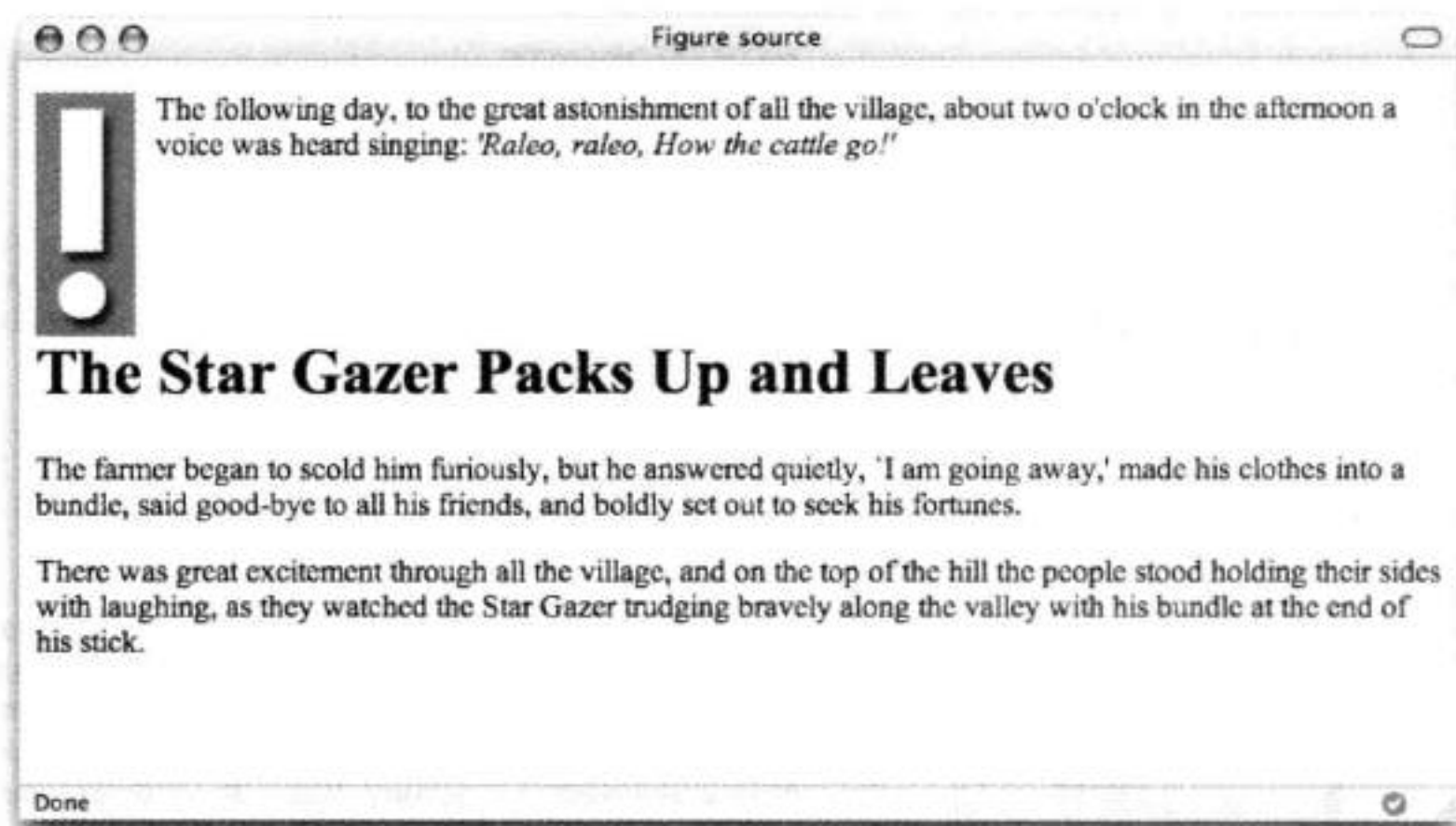
```
img {
 float: left;
 margin-right: 10px;
}
h2 {
 clear: left;
 margin-top: 2em;
}
```

注意，在图15-6中，虽然应用 `2em` 的空白边到 `h2` 元素，但并没有在标题和浮动的图像之间显现。

这是重叠垂直空白边的结果。如果你想让浮动元素和接着的文本保持距离，那就对浮动元素本身应用底部空白边。

现在是在练习15-1中试用浮动属性的时候了，快一试身手吧！

图15-6：清除左侧浮动元素





## 练习15-1 浮动元素

在本练习中，会进一步改进在第14章中修改过的Jenware主页。如果在第14章没有跟着一起做，我为你提供了一个最新的复件，文件名为jenware\_ch15.html，在第15章的材料目录中（[www.learningwebdesign.com/4e/materials](http://www.learningwebdesign.com/4e/materials)）可以找到。

1. 在文本编辑器和浏览器中打开Jenware主页文档（如第14章中的图14-21）。

开始，通过将图像浮动到左边，移除靠近产品图像的多余的垂直空间。我们将使用关联选择器确保只浮动网页“products”部分的图像。这里使用margin速记属性来在右边和底部添加一些空白边。

```
#products img {
 float: left;
 margin: 0 6px 6px 0;
}
```

保存文档，在浏览器中查看。可以看到产品介绍环绕在图像右边。

2. 接着，我想要将“More about...”链接显示在图像下面，这样它们可以在“products”部分清晰可见。这个改动需要一些额外的标记，因为我们需要一种方法来只选择包含“more about”链接的段落。向下滚动滑

动条到文档的标记部分，给包含链接的每一个段落添加类名“more”。下面是第一个：

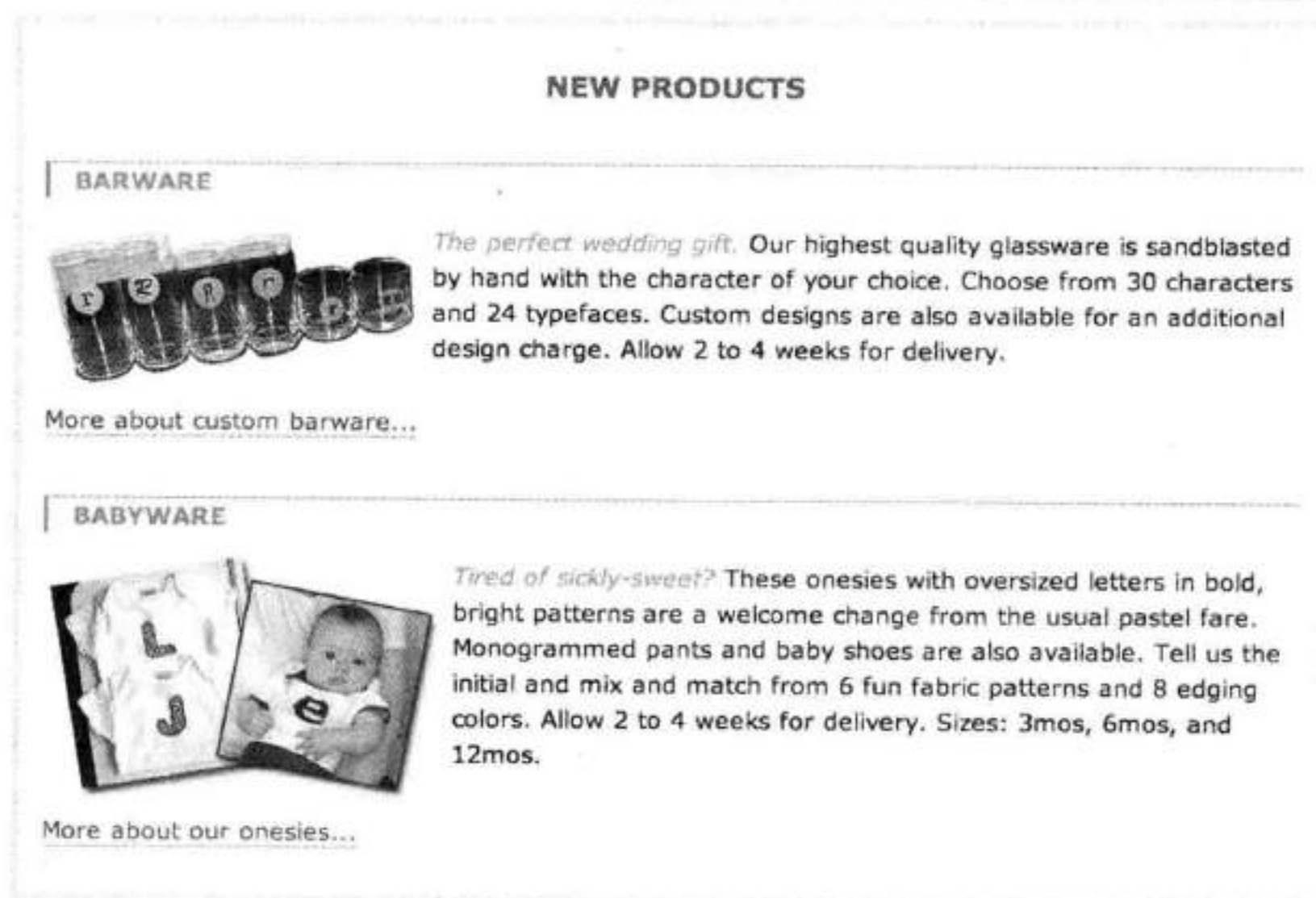
```
<p class="more"><a href=
"#">More about custom
barware...</p>
```

现在我们可以通过类型选择器使那些段落清除浮动图像。

```
#products .more {
 clear: left;
}
```

图15-7显示了新的改进后的Products部分。

图15-7：包含浮动图像和围绕文本的Product部分，浪费空间更少



## 浮动多个元素

在网页甚至单个元素中浮动多个元素，这简直太棒了！事实上，我们将看到，这也是使一系列链接成为水平菜单的一种办法。

当你浮动多个元素时，背后渲染了一个复杂的系统，以确保浮动元素不重叠。你可以参考CSS规范中的细节：浮动元素将被置于最左边或右边（可以指定），以及向上空间允许的最高处。

图15-8显示当一系列连续段落浮动到相同方向时的情景。前三个浮动元素开始从左边依次排列，但当空间不足以放第四个时，第四个移到下方并向左移动直到碰壁；本例中，那个壁就是浏览器窗口的边缘。然而，如果其中一个浮动元素（如“P2”）已经很长了，那么第四个将向上撞到长浮动元素的边缘，而不是浏览器窗口边缘。

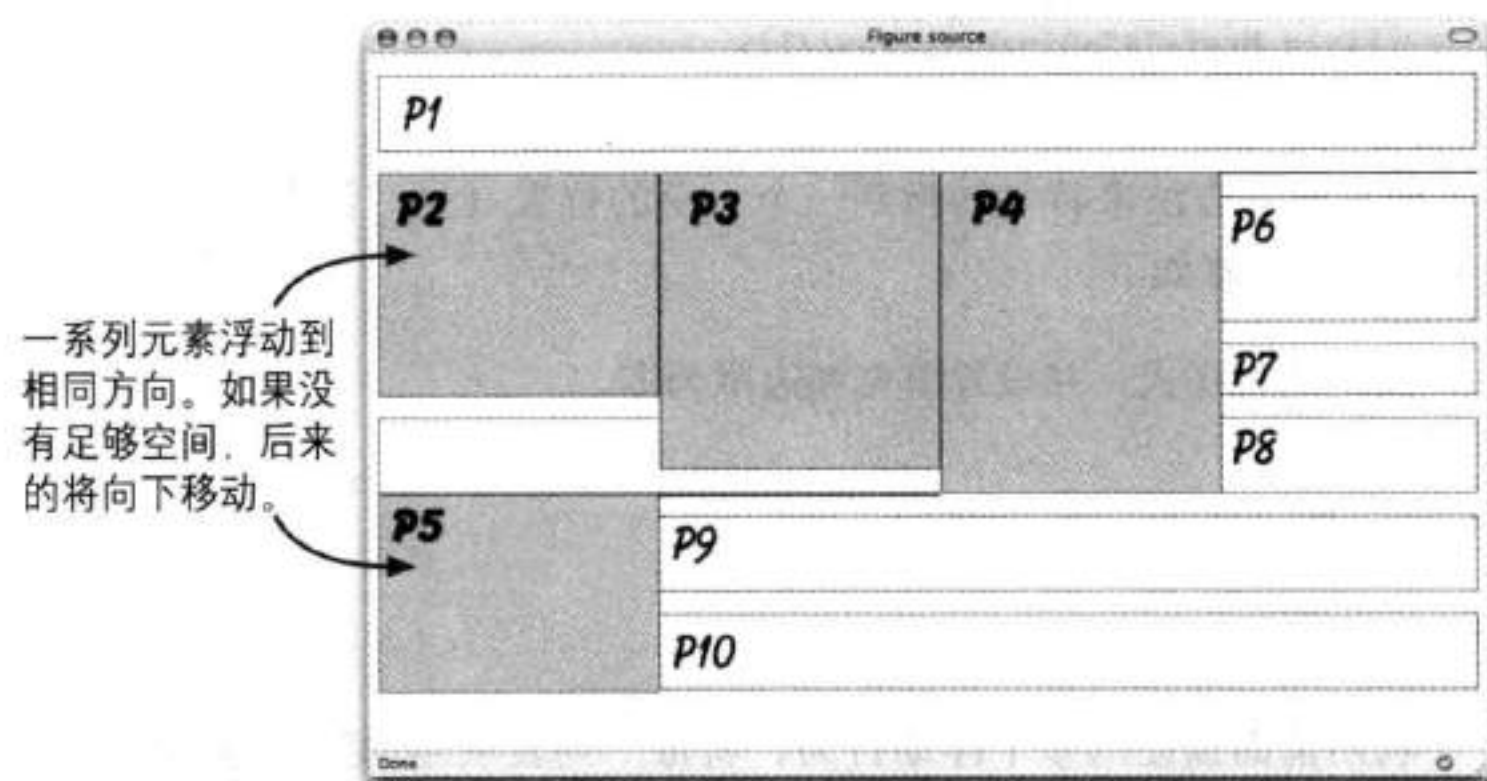


图15-8：多浮动元素排列而不重叠

### 源码

```
<p>P1</p>
<p class="float">P2</p>
<p class="float">P3</p>
<p class="float">P4</p>
<p class="float">P5</p>
<p>P6</p>
<p>P7</p>
<p>P8</p>
<p>P9</p>
<p>P10</p>
```

### 样式表

```
p.float {
```



```

float: left;
width: 200px;
margin: 0px;
background: #CCC;
}
p
{border: 1px solid red;
}

```

这是底层的行为，我们可以用它来做一个导航菜单。作为一个无序列表，它用来标记导航是有意义的，如下所示。为了简化标记，我忽略了a元素中的URL值。

```


 Serif
 Sans-serif
 Script
 Display
 Dingbats


```

有各种各样的方法来将它转换到一个水平的位置（见“注意”），浮动例子的主要步骤如下。

1. 关闭列表样式，并设置填充和边距为零。

```

ul {
 list-style-type: none;
 margin: 0;
 padding: 0;
}

```

2. 利用前面描述的多个浮动行为，将每个列表项浮动到左边，使它们排列起来。

```

ul li {
 float: left;
}

```

3. 将列表项中的锚元素显示为块元素，这样就可以设置尺寸、填充、空白边和其他视觉效果了。你可以为其他链接状态设置样式（如a:hover），但我想让这个例子简短些。

```

ul li a {
 display: block;
 /* more styles */
}

```

4. 清除文档菜单之后的元素，使得它从菜单下方开始。

至少，你会想给锚元素添加一些填充或边框，从而给链接一点空间，而且你可以添加目前见过的所有颜色、边框、圆角、背景图片——来给导

**注意：**另一个排列列表项的办法是使它们显示在内联元素中，而不是显示在块元素中：(li {display: inline;})。这样你就可以使锚元素像块一样显示，并且对其应用样式。但是由于浏览器根据容器的font-size设置了列表项的间距，所以该方法难以精确地控制导航项之间的间距。

航你想要的外观。下面的样式将前面的列表示例转换为像标签一样的菜单，如图15-9所示。



图15-9：只使用CSS，不需要使用图像，就可以把无序列表变成像标签一样的菜单

## 包含浮动

就像已经学过的多个浮动一样，浮动的包含也是浮动的另一个问题。默认情况下，浮动元素可以浮出包含它的元素。这允许文本在浮动图像周围流动，但有时这种行为可能会导致一些不必要的问题。

例如，在图15-10的例子中。显然，如果边界能延伸，足以包含所有的内容，那么看起来效果会更好，但浮动的图像正好悬出底部。

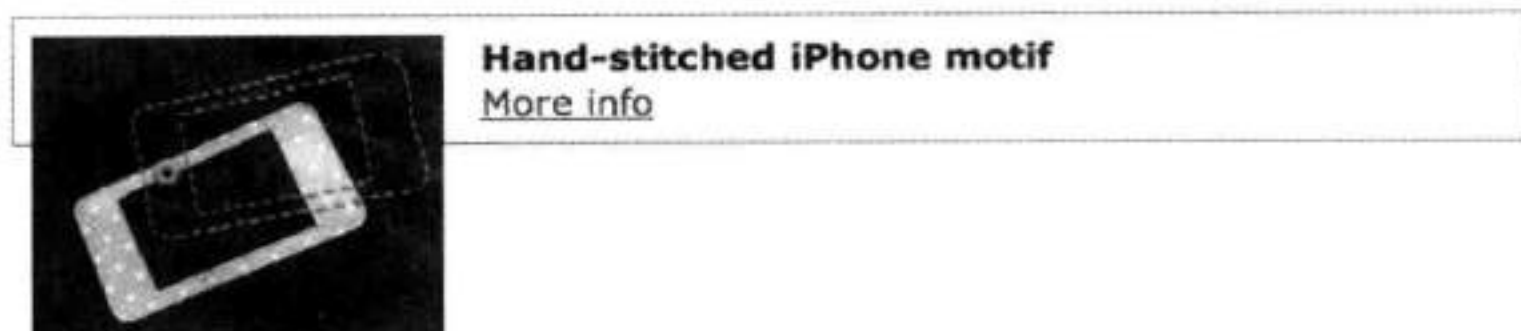


图15-10：包含元素不能拉伸来容纳浮动的图像

如果浮动容器中所有的元素——就像创建一个多列布局一样——则流中没有元素可以打开包含元素。这种现象在图15-11中得到了例证。`#container div`容器包含两个段落。普通流（左）显示`#container`容器有一个背景颜色边框环绕在内容周围。然而，当这两段浮动时，`#container`的元素盒子就缩在了一起，高度为0，使浮动倒挂在下面（你仍然可以看到顶部的空边框）。这显然不是我们所追求的效果。

在普通流中，容器div围绕着段落

*Etiam convallis, nulla ut ullamcorper mollis, ipsum purus imperdiet tellus, ut ultrices massa tortor vitae nulla. Fusce non arcu quam. Nullam lacinia facilisis lacus, et varius ligula imperdiet ut. Morbi molestie auctor magna, quis venenatis felis adipiscing sed. Aliquam ipsum nibh, dapibus sit amet tristique at, tincidunt in leo. Quisque accumsan lobortis lacus, id gravida tortor luctus et. Donec quis diam et odio volutpat blandit nec nec enim. Nam vitae vestibulum risus. Cras in adipiscing odio. Nam vel dolor id purus pretium suscipit quis in quam. Proin varius tincidunt facilisis. Maecenas eget felis ut nisi ullamcorper pretium non at nulla. Etiam suscipit aliquet velit ac facilisis. Etiam eget ante eu velit ullamcorper ornare. Suspendisse vestibulum leo sed lectus posuere eget convallis nisi placerat. Vestibulum porttitor egetas ornare.*

*Cras id ipsum dui. Donec semper congue lectus quis vulputate. Ut felis leo, bibendum at blandit non, luctus ac lorem. Nunc vitae ligula ut neque convallis sagittis. Quisque consequat orci sed arcu tincidunt et volutpat tellus tempor. Nulla vulputate ante nec felis elementum auctor. Duis magna neque, posuere eu hendrerit sit amet, dapibus quis quam. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nunc dapibus dui dignissim dolor rutrum vel consequat nibh sagittis. Morbi non dolor diam, nec iaculis neque. Aenean at eros sit amet velit iaculis porttitor. Nam lobortis sodales augue, sit amet tincidunt erat sagittis eu. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Donec ut ultricies velit. Quisque tempor fermentum ante, quis tempus est fringilla eu.*

当两段都浮动时，容器没有拉伸来包含它们

*Etiam convallis, nulla ut ullamcorper mollis, ipsum purus imperdiet tellus, ut ultrices massa tortor vitae nulla. Fusce non arcu quam. Nullam lacinia facilisis lacus, et varius ligula imperdiet ut. Morbi molestie auctor magna, quis venenatis felis adipiscing sed. Aliquam ipsum nibh, dapibus sit amet tristique at, tincidunt in leo. Quisque accumsan lobortis lacus, id gravida tortor luctus et. Donec quis diam et odio volutpat blandit nec nec enim. Nam vitae vestibulum risus. Cras in adipiscing odio. Nam vel dolor id purus pretium suscipit quis in quam. Proin varius tincidunt facilisis. Maecenas eget felis ut nisi ullamcorper pretium non at nulla. Etiam suscipit aliquet velit ac facilisis. Etiam egetas ante eu velit ullamcorper ornare. Suspendisse vestibulum leo sed lectus posuere eget convallis nisi placerat. Vestibulum porttitor egetas ornare.*

*Cras id ipsum dui. Donec semper congue lectus quis vulputate. Ut felis leo, bibendum at blandit non, luctus ac lorem. Nunc vitae ligula ut neque convallis sagittis. Quisque consequat orci sed arcu tincidunt et volutpat tellus tempor. Nulla vulputate ante nec felis elementum auctor. Duis magna neque, posuere eu hendrerit sit amet, dapibus quis quam. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nunc dapibus dui dignissim dolor rutrum vel consequat nibh sagittis. Morbi non dolor diam, nec iaculis neque. Aenean at eros sit amet velit iaculis porttitor. Nam lobortis sodales augue, sit amet tincidunt erat sagittis eu. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Donec ut ultricies velit. Quisque tempor fermentum ante, quis tempus est fringilla eu.*

图15-11：当所有的内容浮动时，容器盒子整个消失了



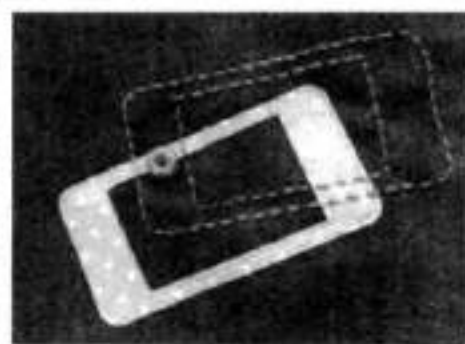
幸运的是，有几个直接的方式可以解决这个问题。一种选择是将包含元素也设为浮动的，并将包含元素的宽度设为100%。

```
#container {
 float: left;
 width: 100%;
 background-color: #GGG;
 padding: 1em;
}
```

其他常见的解决方案是利用overflow属性的优势。将包含元素的溢出设为auto或者hidden，也将使其拉伸以包含浮动元素。我也增加了一个明确的宽度值，以解决旧版本的IE中的bug，但要注意，如果容器元素有边框，100%的宽度会使边框悬在浏览器窗口之外。

```
#container {
 overflow: auto;
 width: 100%;
 background-color: #GGG;
 padding: 1em;
}
```

图15-12：悬出的浮动现在被边框包含了



#### Hand-stitched iPhone motif

[More info](#)

图15-12显示了在之前的例子中运用包含技术的结果。这个技巧谁都会做。

现在可以完成练习15-2中Jenware页面的导航部分了。

**Etiam convallis**, nulla ut ullamcorper mollis, ipsum purus imperdiet tellus, ut ultrices massa tortor vitae nulla. Fusce non arcu quam. Nullam lacinia facilisis lacus, et varius ligula imperdiet ut. Morbi molestie auctor magna, quis venenatis felis adipiscing sed. Aliquam ipsum nibh, dapibus sit amet tristique at, tincidunt in leo. Quisque accumsan lobortis lacus, id gravida tortor luctus et. Donec quis diam et odio volutpat blandit nec nec enim. Nam vitae vestibulum risus. Cras in adipiscing odio. Nam vel dolor id purus pretium suscipit quis in quam. Proin varius tincidunt facilisis. Maecenas eget felis ut nisi ullamcorper pretium non at nulla. Etiam suscipit aliquet velit ac facilisis. Etiam egestas ante eu velit ullamcorper ornare. Suspendisse vestibulum leo sed lectus posuere eget convallis nisi placerat. Vestibulum porttitor egestas ornare.

**Cras id ipsum dui**. Donec semper congue lectus quis vulputate. Ut felis leo, bibendum at blandit non, luctus ac lorem. Nunc vitae ligula ut neque convallis sagittis. Quisque consequat orci sed arcu tincidunt et volutpat tellus tempor. Nulla vulputate ante nec felis elementum auctor. Duis magna neque, posuere eu hendrerit sit amet, dapibus quis quam. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nunc dapibus dui dignissim dolor rutrum vel consequat nibh sagittis. Morbi non dolor diam, nec iaculis neque. Aenean at eros sit amet velit iaculis porttitor. Nam lobortis sodales augue, sit amet tincidunt erat sagittis eu. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Donec ut ultricies velit. Quisque tempor fermentum ante, quis tempus est fringilla eu.

## 练习15-2 制作导航栏

如果还没准备好，可以打开你复制的 *jenware.html*（或者 *jenware\_ch15.html*）。

1. 首先，尽可能使 `ul` 元素中性。样式类型已经关闭，但是我们需要清除所有可能的填充和空白。

```
#nav ul {
 list-style-type: none;
 padding: 0;
 margin: 0;
}
```

2. 接下来，将列表项浮动到左边，并且清除后面的 `products div`。

```
#nav ul li {
 ...
 float: left;
}

#products {
 ...
 clear: both;
}
```

保存文档，在浏览器中查看。你应该看到链接紧密排列成一行，但是紫色的导航栏却缩了起来——浮动包含失败了！可以使用溢出技巧来解决这个问题。现在对 `#products div` 做同样的事，这样它就可以确保包含浮动图像了。

```
#nav {
 ...
 overflow: hidden;
 width: 100%;
}

#products {
 ...
 overflow: hidden;
}
```

3. 现在来处理链接的外观。首先，将 `a` 元素显示为块级元素，而不是内联元素。不必为每个链接指定尺寸，我们可以使用填充（.5em）来为它们在边框内部提供一点空间，并且使用空白边（.25em）

来给链接间添加一些空白。我已经添加了默认的淡紫色的边框，但是我为 `:focus` 和 `:hover` 增亮成白色。

```
#nav ul li a {
 display: block;
 padding: .5em;
 border: 1px solid #ba89a8;
 border-radius: .5em;
 margin: .25em;
}
```

```
#nav ul a:focus {
 color: #fc6;
 border-color: #fff;
}
```

```
#nav ul a:hover {
 color: #fc6;
 border-color: #fff;
}
```

4. 最后，把 `nav` 部分的列表放在中间。我们可以对 `ul` 元素指定宽度，并且把它的侧边空白区设置为 `auto`。坦白的说，我试了好几个测量值，来找到一个恰到好处的值来适应整个菜单（19.5em）。如果它太宽了，菜单就不可能真的放在中间。

```
#nav ul {
 list-style: none;
 padding: 0;
 margin: 0 auto;
 width: 19.5em;
}
```

图15-13显示了你的导航在浏览器中看起来的样子。



图15-13：链接列表成为了一个水平菜单栏



## 使用浮动创建列

到目前已经浮动了一小部分页面，但正如前面提到的，你也可以浮动页面的整个部分来创建列。事实上，专家也是这么做的！有两种解决方案，可以根据自己的偏好来选择。

对于一个两列的浮动，你可以做到以下几点：

- 浮动一个div，并在文本元素周围添加大量的空白。
- 浮动两个div到左边或右边。
- 浮动一个div到左侧，浮动第二个div到右边（或反之亦然）。

三列的浮动工作方式基本相同，只是需要更多的计算。

无论你认为哪个方法好，有几件事情你需要记住。首先，每一个浮动需要有一个指定的宽度。其次，你需要仔细认真地计算每列的宽度，并分配给填充、边框和空白。如果所有列的总宽度超过了浏览器或其他包含元素的宽度，你将得到所谓的“浮动下降”：最后浮动的列会用尽空间而被挤下来，排在它旁边的列下面。

使用浮列的限制是，它是依赖于源文件中元素的顺序。浮动元素必须出现在它所环绕的内容之前，而且你的源代码并不总是可以方便排序的。

现在，采用上面列出的“一个浮动加上空白”的技术，在练习15-3中完成一个两列的布局。

注意：使用第16章的负值空白，会有很多办法来摆脱源文档中的顺序。

### 练习15-3 使用浮动创建多列布局

我们已经完成的Jenware站点的布局对于小屏幕设备是不错的，但是在大的浏览器上就不好看了。在本练习中，我们会使用浮动技术来完成样式，给页面提供流动的两列布局。我建议复制当前的Jenware文件，并且重命名为 *jenware-float.html*。这会为后面的练习保存一份副本，否则你还得把本练习所做的工作再改回去。

我们要给#products div提供一个宽度，将它浮动到左边，并且创建第二列，使得testimonials盒子在右侧围绕着它。我想让这个布局总是可以调整尺寸来填满屏幕的宽度，所以我将为所有的水平值使用百分比值（意味着需要修改之前的代码）。

1. 首先设置#products div的宽度为55%，并且将它浮动到左边。现在四周的填充和空白边都设置为1em，但是为了流动布局将左边和右边的填充和空白边改变为2%。也就是说，Products盒子现在总共占据了屏幕63%的宽度（2%+2%+55%+2%+2%），边框也占了一些像素。图15-14显示了这些改变的结果。此外，把#products顶部的空白边设置为0。

```
#products {
 background-color: #FFF;
 line-height: 1.5em;
 padding: 1em 2%;
 border: double #FFBC53;
 margin: 0 2% 1em;
 clear: both;
 float: left;
 width: 55%;
}
```

这里有一些有意思的东西。Testimonials文本果然移动到了Products盒子的右边，但是Testimonials盒子（还有感叹号图像）隐藏在了Products盒子里。只有内容包覆盖在上面，元素盒子只是往上移，也并没有调整尺寸。

2. 该轮到Testimonials盒子了。我们需要做的是调整空白边，特别是将Testimonials盒子左边的空白边变宽，使其清除Products盒子。Products盒子占据了页面63%的宽度，所以，给Testimonials盒子左边设置64%的空白，并且在这两个盒子之间添加一点空白。我还设置了2%的右空白（还记得吧，值的顺序是Top, right, bottom, left）。重新载入页面，Testimonials盒子应该在右列的中间了。

```
#testimonials {
 ...
 margin: 1em 10%; /* delete */
 margin: 1em 2% 1em 64%;
}
```

3. 还有一点没完成。清除copyright段，这样它就显示在页面的底部了。最后，我想“New products” h2在这个布局中靠左对齐，应该看起来会更好，所以把它也调整一下。

```
p#copyright {
 ...
 clear: left;
}
#products h2 {
 ...
 text-align: center left;
}
```

结果显示在图15-15中。看看怎么样！你的第一个两列布局，创建了一个浮动，还有一个宽的空白区。这就是很多以CSS为基础的布局模板背后的基本概念。你将会

在第16章看到。



图15-14：浮动products div的结果



图15-15：Jenware主页的两列布局，通过对后面的内容添加一个浮动和一个宽空白来创建，这个布局在平板设备或者桌面浏览器中也运行良好



**警告：** 使用流动列和边框时要小心。为了适应边框宽度（如果使用边框），以及浏览器有时的四舍五入，所使用百分比值加起来要小于100%。如果太多的列宽四舍五入，这些列加起来对于浏览器可能就太宽了，然后就可能让浮动下降。

混用100%和Ems

在练习15-2中，使用百分比值和ems指定空白。在现代的网页设计中，这么做很普遍，尤其是在创建流动布局时。一些开发者为所有的水平测量值都使用百分比值，所以这些值都与视窗大小有关，但是对所有的纵向测量值都用ems，因为它们与文本行的多少和排列有关。这个技术只是一种偏好，而不是非要这么做，但需要了解一下。

这涵盖了浮动的基础。接下来介绍另外一个在页面移动元素的方法——定位。

定位基础

CSS提供了几种方法来定位页面上的元素。他们可相对于通常在流中出现的地方定位，也可以完全从流中取出，并放置在页面特定的位置上。你也可以将元素相对于浏览器窗口（技术上称为CSS的建议视窗）定位，当页面的其他部分滚动时，它会保持不变。

定位的类型

position

- 属性值： static|relative|absolute|fixed|inherit
- 默认值： static
- 适用对象： 所有元素
- 是否可继承： 否

position属性指示元素将定位，并指定将使用的定位方法。这里将简单地介绍每个关键字值，然后在本章最后部分详细介绍各个方法。

static

这是普通的定位方案，元素将与在普通文档流中一样定位。

relative

相对定位将盒子相对于它在流中的原始位置移动。相对定位与众不同之处在于它将保留元素在普通流中占据的空间。

absolute

绝对定位元素从文档流中完整地移除，并且相对于容器元素（更多的在后面讨论）定位。不同于相对定位，它所占据的空间将被关闭。事实上，它对周围元素的布局没有任何影响。

fixed

固定定位的特性是元素将待在窗口的一个位置，即使在文档滚动的

时候。固定元素从文档流中移除并相对于浏览器窗口（或其他的视口），而不是文档中的另一个元素定位。

每个定位方法都有其目的，但绝对定位的目的最多样。运用绝对定位，你可以在视窗的任何地方放置一个对象，或另一个元素。绝对定位甚至可以用于创建多栏布局，但它比较常用于小任务，比如，在标题的上方放置一个搜索框。你也可以使用绝对定位，打破图像或它的包含箱，创建悬挂缩进或重叠效果。这是一个方便的工具，应当仔细和谨慎地使用。

## 指定position属性

一旦确定了定位方法，真正要做的就是指定四个偏移属性。

top, right, bottom, left

属性值： 长度计量值 | 百分比值 | auto | inherit

默认值： auto

适用对象： 可定位元素（元素的position属性值为relative、absolute或fixed）

是否可继承： 否

这些值提供了各个元素与相应边距离的偏移属性。例如，top属性值定义了从浏览器或其他容器元素的上边缘到元素上边缘的偏移距离。top属性值导致元素盒子以这个值向下移动。类似地，left属性值将向右移动定位元素（相对于窗口块的中心）指定值的距离。

更多的关于偏移属性的解释和例子将在各个定位方法的讨论中提供。首先使用相当直接的relative方法，来简单开始定位。

**注意：** 负值是可接受的，将元素向相反方向移动。例如，top属性的负值将使元素向上移动。

## 相对定位

如前面所提到的，相对定位是相对于元素在流中的原始位置移动的。它所占据的空间将被保留，并继承影响周围内容的布局。用下面的例子就可很容易地理解。

这里将定位一个内联的em元素（背景色将使边界可见）。首先，我将使用position属性设置方法为relative，然后top偏移属性将元素从原始位置下移30像素，left属性使元素向右移动60像素。记住，偏移属性值从指定边按相反方向移动元素，所以，如果你想把元素移到右边，那就像我这样做，使用left偏移属性。效果如图15-16所示。



```

em {
 position: relative;
 top: 30px;
 left: 60px;
 background-color: fuchsia;
}

```

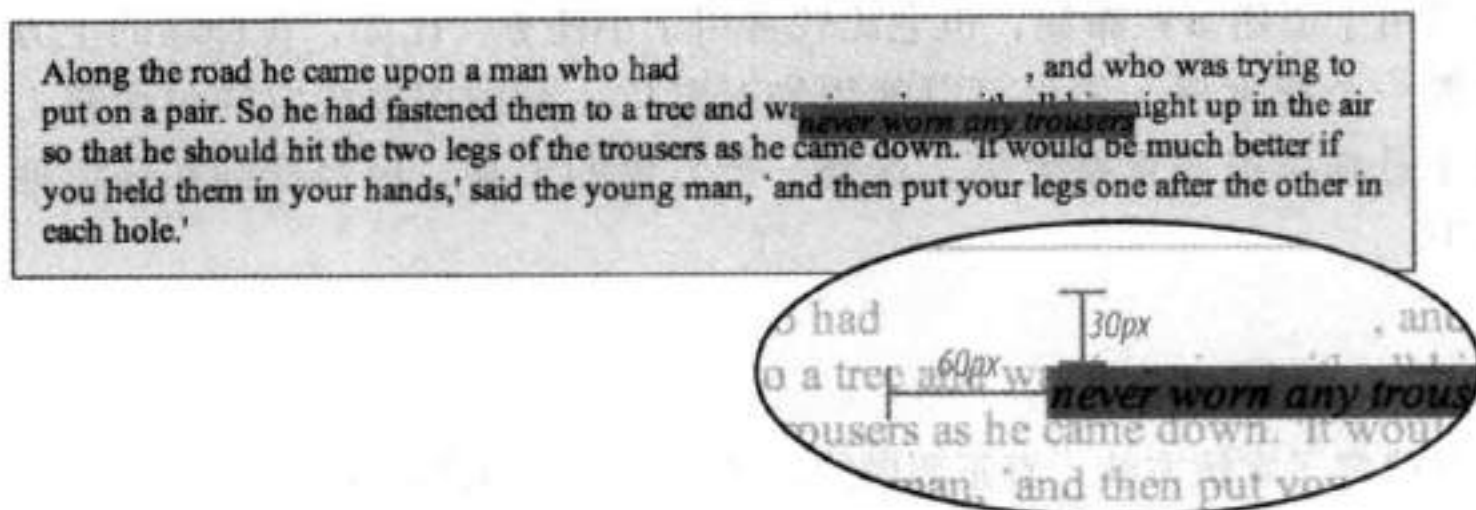


图15-16：当元素使用相对方法定位时，它所占据的空间将被保留

我将指出这里发生的一些事：

### 文档流中的原始空间被保留

你可以看到有一个空白空间，如果元素没有被定位，强调文本应该在那个位置。周围的内容就像元素还在那里一样布局，因此可以认为这个元素仍在“影响”周围内容。

### 发生交迭

因为这是个定位元素，潜在地，它可能与其他元素交迭，如图15-16所示。

相对定位的对象左后方的空白可能显得有一点笨拙，所以这种方法不像浮动和绝对定位那样常用。然而，相对定位通常用于为绝对定位元素创建“定位环境”，下面将对此进行解释。

## 绝对定位

绝对定位工作方式有一点不同，事实上它比相对定位更灵活地完成网页布局。既然你已经见过相对定位是如何工作的，那就再使用如图15-16中相同的例子，不过这次将把position属性值设为absolute。

```

em {
 position: absolute;
 top: 30px;
 left: 60px;
 background-color: fuchsia;
}

```

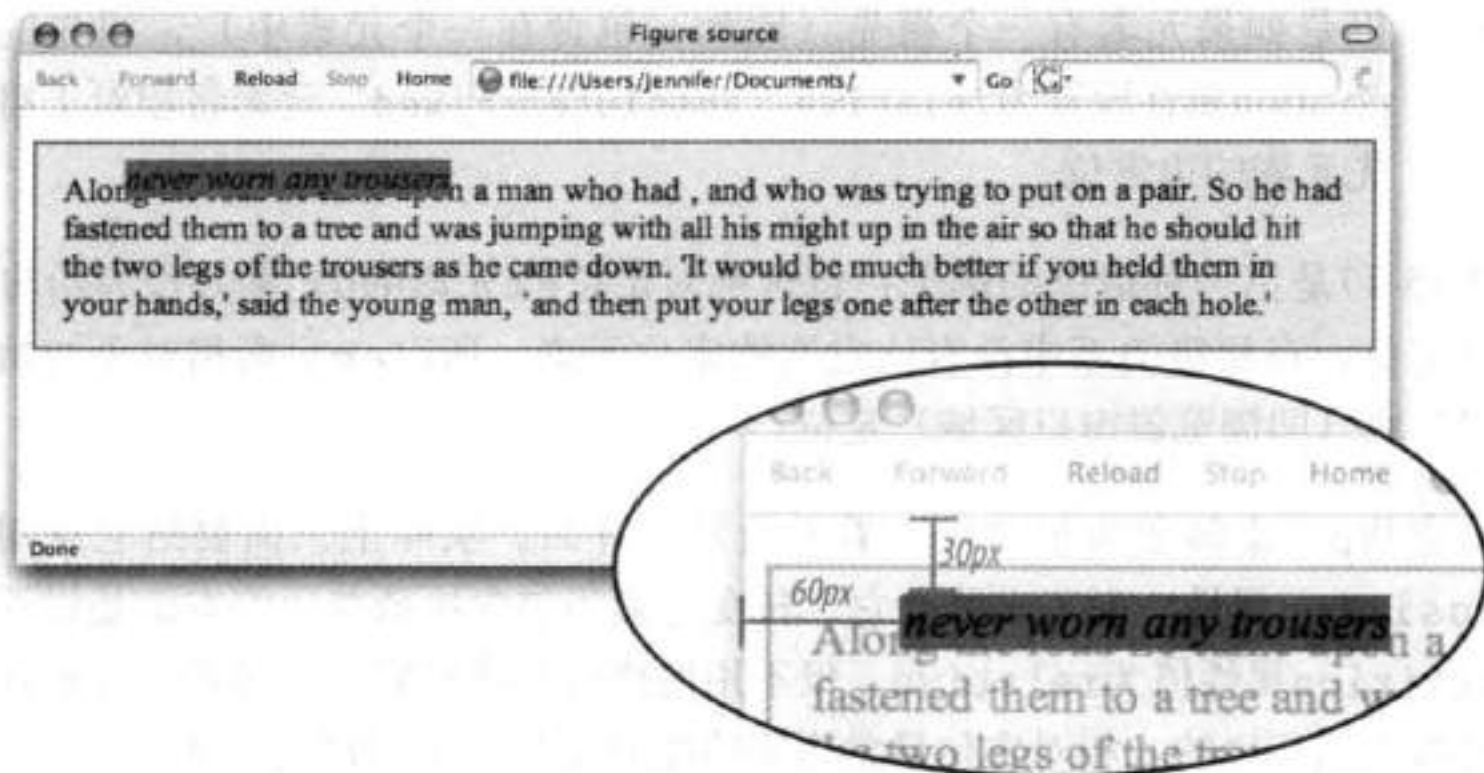


图15-17：当元素被绝对定位时，它将从流中移除，空间会关闭

如图15-17所示，作为所有绝对定位元素的一个例子，曾经被em元素占据的空间现在关闭了。在新位置上，元素与周围内容交迭。最后，绝对定位元素对周围元素的布局没什么影响。

然而，最大的不同之处是浮动元素的位置。这次，偏移值把em元素定位在浏览器窗口左上角下方30像素，右方60像素。

但是，在你开始认为绝对定位元素总是相对浏览器窗口定位之前，我想你还需要了解更多的信息。

在绝对定位中，实际发生的是元素相对于最近的包含块定位。在图15-17中，恰好最近的包含块是根（html）元素（也称为初始包含块），所以偏移值相对于整个浏览器窗口区域定位。

掌握包含块概念是学习绝对定位的第一步。

## 包含块

CSS2.1介绍声明，“元素盒子的位置和尺寸有时相对于某个称为元素的包含块的矩形计算”，知道你想定位的元素的包含块很重要。我们有时候把它叫做“定位环境”。

CSS2.1展示了很多判断元素包含块的复杂规则，但基本可以简化为下面几条：

- 如果定位元素没有包含在另一个定位元素中，那么它将相对于初始包含块（由html元素创建的）确定位置。

**注意：**一些浏览器基于body元素作为初始包含块。最后的结果与填充浏览器窗口相同。

### 或者，试试另一种方式？

绝对定位元素的包含块是最先用来定位的祖先元素（也就是说包含块会有position的值，而不是static）。

如果没有出现包含块（也就是说，如果定位元素并不包含于另一个定位元素中），那么将会使用初始包含块（html元素创建的块）。



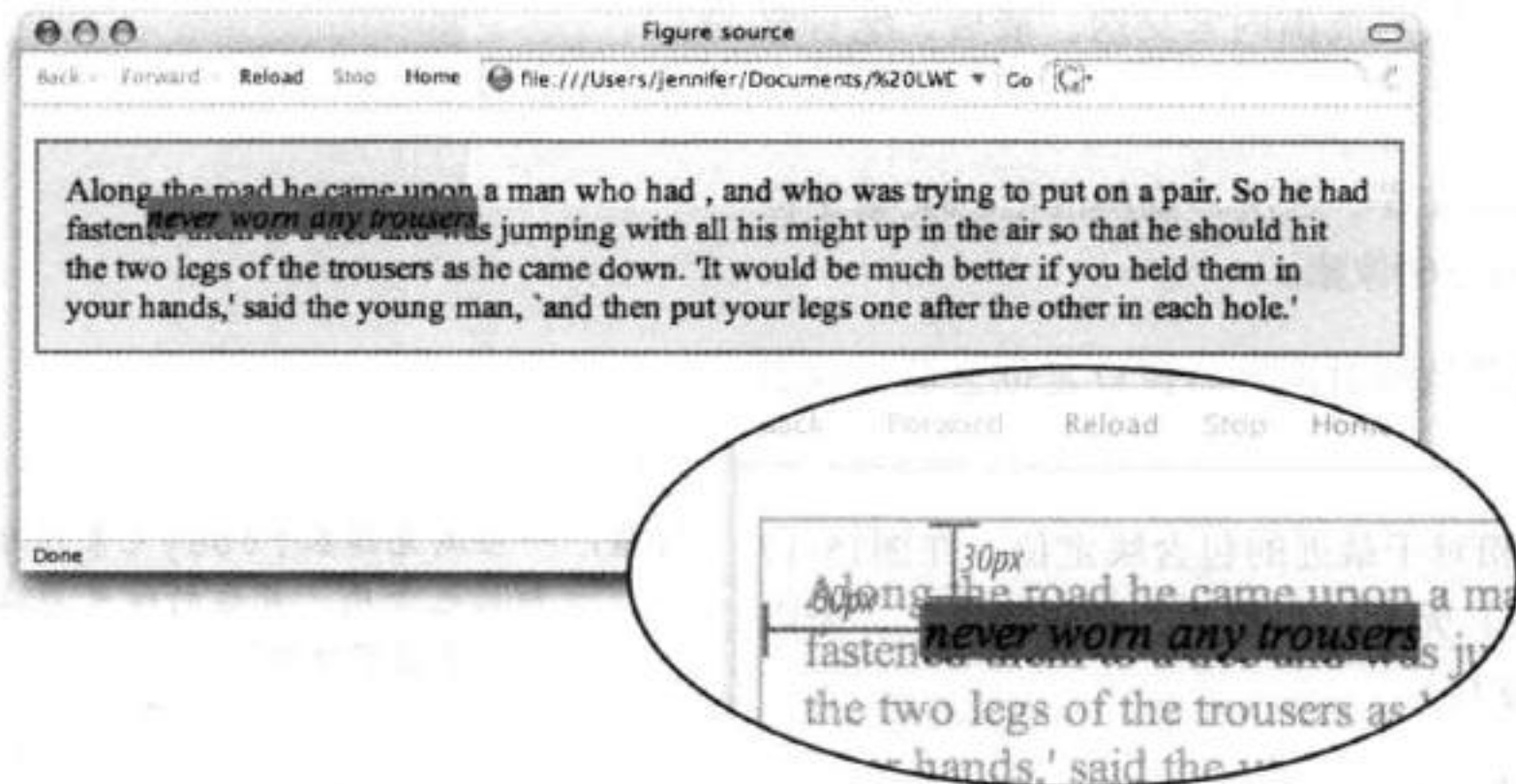
- 但是如果元素有一个祖先（比如，包含在一个元素中），祖先的 `position` 属性设置为 `relative`、`absolute` 或 `fixed`，元素将相对于祖先元素的边定位。

图15-17是第一种情况的例子：包含绝对定位 `em` 元素的 `p` 元素并没有给自己定位，在层级关系中没有比它高的定位元素，所以 `em` 元素相对于初始包含块（即浏览器窗口区域）定位。

故意将 `p` 元素转变为包含块，看看会发生什么。实际上只需要给它应用 `position` 属性，而不必移动它。转变元素为包含块最普通的方法是设置 `position` 属性值为 `relative`，但不用偏移值来移动它。（另外，这也就是我之前所说的：相对定位最常用来给绝对定位元素创建上下文。）

图15-18：相对定位的 `p` 元素就像 `em` 元素的包含块

我们将给 `em` 元素使用相同的样式规则，但将给 `p` 元素添加 `position` 属性，从而将它设置为定位 `em` 元素的包含块。效果如图15-18所示。



```
p {
 position: relative;
 padding: 15px;
 background-color: #DBFDBA;
 border: 2px solid #6C4788;
}
```

可以看到，`em` 元素现在位于段落盒子（而不是浏览器窗口）左上角的下方30像素和右方60像素。还要注意，它相对于段落的填充边缘（仅包括边框内的），而不是内容区域边缘。这是当块元素用作包含块时的普遍行为（见“注意”）。

**注意：** 当内联元素用作窗口块（或可能成为）时，定位元素将相对于内容区域边缘确定位置，而不是相对于填充边缘。

我将更详细地揭示绝对定位对象。这次要给定位 `em` 元素添加 `width` 和 `margin` 属性（如图15-19所示）。

```
em {
 width: 200px;
 margin: 25px;
 position: absolute;
 top: 30px;
 left: 60px;
 background-color: fuchsia;
}
```

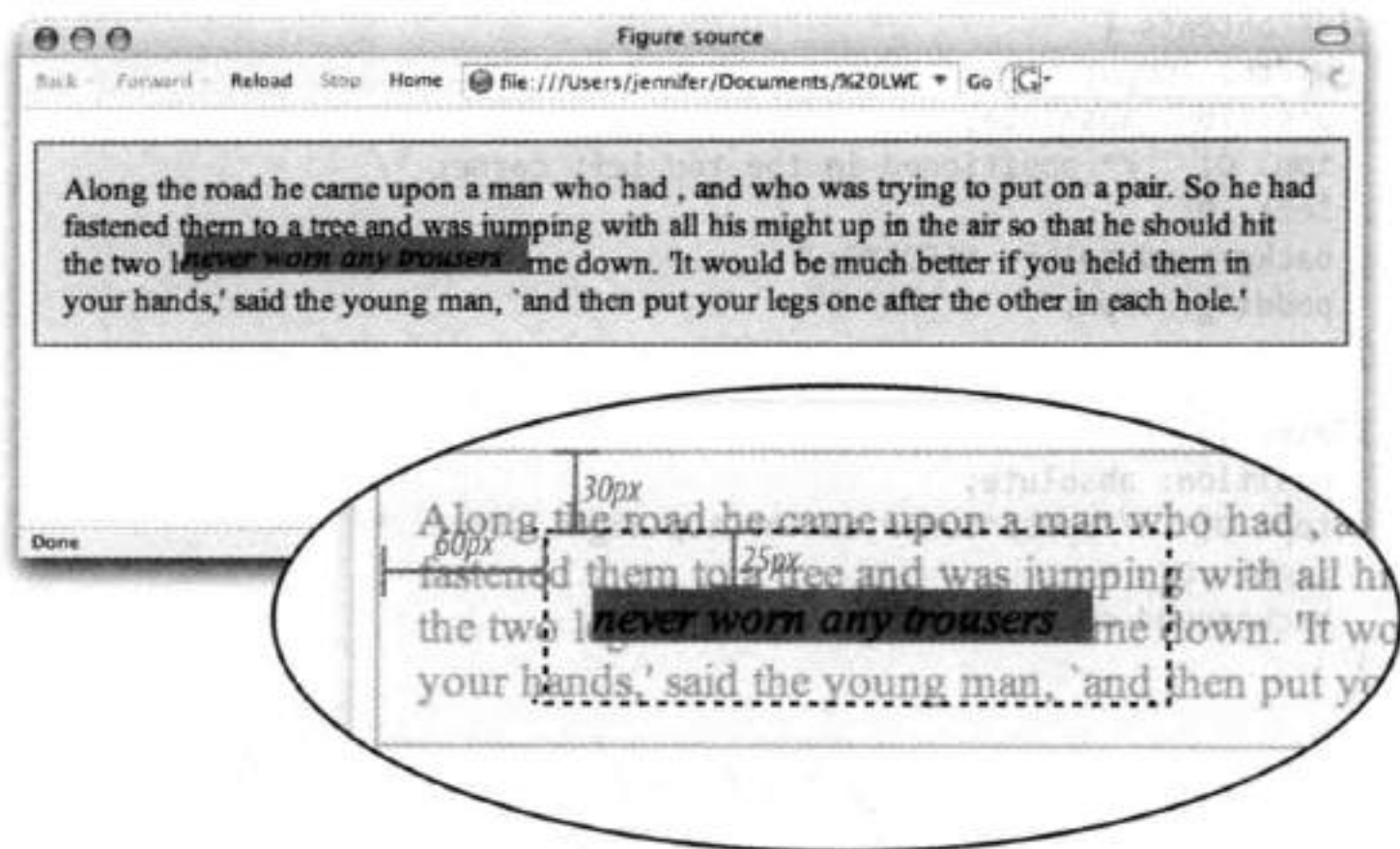


图15-19：给定位元素添加宽度和空白边

这里可以看到：

- 偏移值应用于元素盒子的外边缘（从空白边边缘到空白边边缘）。
- 绝对定位元素总是表现为块级元素。例如，所有方向的空白边都保持着，即使这是一个内联元素。同时也允许给元素设置宽度。

记住这点很重要：一旦你已经定位了一个元素，它将变为它包含的所有元素的新包含块。考虑这个例子，名为“content”的div元素定位于网页的左上角。当div中的定位列表项设置偏移值，将其置于右上角时，它会出现于content div元素的右上角，而不是整个网页（如图15-20所示）的右上角。这是因为一旦div定位了，它将表现为li元素的包含块。

### 标记文本

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

<div id="contents">
<h2>Contents</h2>

 The Nix in Mischief
 <li id="special">The Ogre Courting
 Murdoch's Wrath
 The Little Darner
 The Magic Jar

</div>
```

### 样式表



```

div#contents {
 width: 200px;
 position: absolute;
 top: 0; /* positioned in the top-left corner */
 left: 0;
 background-color: #AFD479;
 padding: 10px;
}

li#special {
 position: absolute;
 top: 0; /* positioned in the top-right corner */
 right: 0;
 background-color: fuchsia;
}

```

定位后的“contents”div成为定位li元素的包含块，并且创建了一个新的定位环境



图15-20：定位元素变成它们包含的元素的包含块。本例中，列表项相对于所在的div元素定位，而不是整个网页

## 指定position属性

既然已经对包含块概念有了更好的理解，那就花些时间更好地熟悉偏移属性。现在只看到元素向下或向右移动了一些像素，其实还有很多其他的属性。

## 像素计量值

前面提到过的，正偏移值推动定位元素盒子远离指定的边缘，并朝向包含块的中心移动。如果没有给边缘提供值，它就设置为auto，浏览器添加足够的空间使布局正常运行。在本例中，已经给四个偏移属性使用了像素长度，来定位元素到容器元素的特定位置（如图15-21所示）。

```
div#a {
 position: relative; /* 创建包含块 */
 height: 120px;
 width: 300px;
 border: 1px solid;
 background-color: #CCC;
}

div#b {
 position: absolute;
 top: 20px;
 right: 30px;
 bottom: 40px;
 left: 50px;
 border: 1px solid;
 background-color: teal;
}
```

**div#a** (*width: 300px; height: 120px;*)

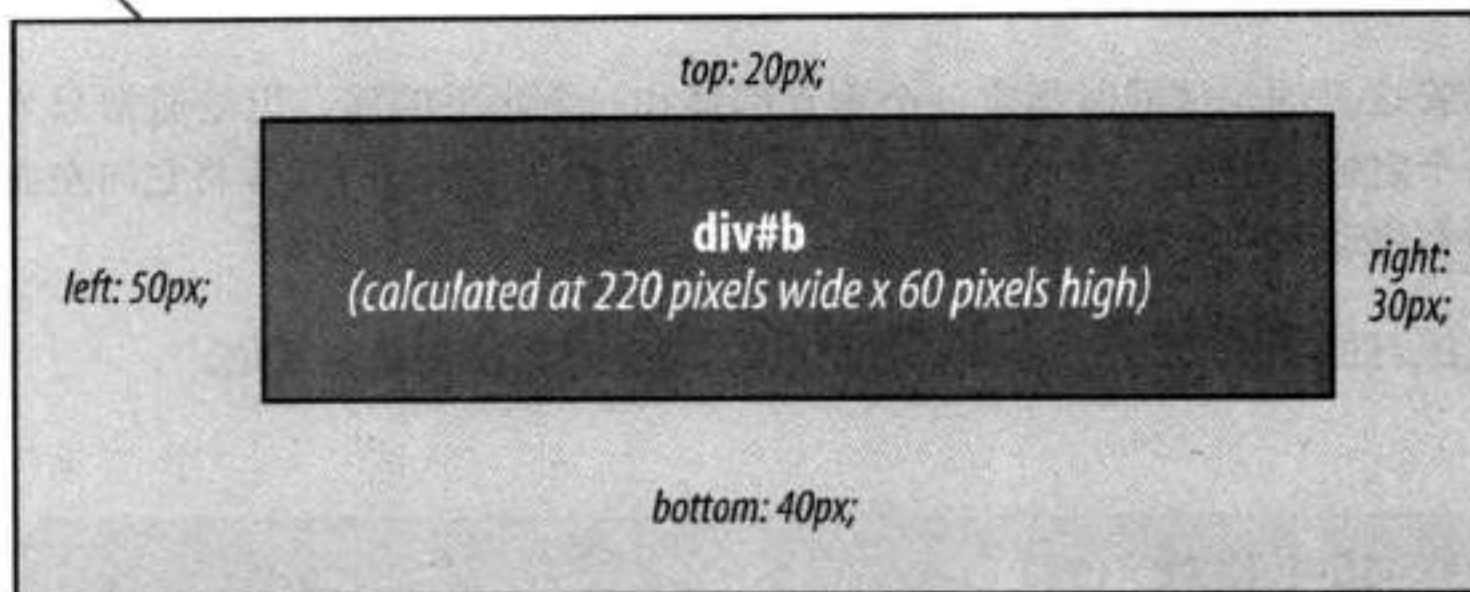


图15-21：给定位元素的四个方向设置偏移值

注意，通过给四个方向设置偏移值，我已经间接地设置定位

div#b

元素的尺寸（偏移值应用后，会填充220×60像素包含块中的空间）。如果我也给

div#b

指定width和其他盒子属性，那如果定位盒子的属性值与偏移量与包含块中的可用空间不匹配，就可能引发冲突。

CSS规范提供了令人畏惧地处理冲突的规则集，但结果是只要仔细些，不要过分指定属性和偏移。通常，width（加上可选的padding、border和margin）和一两个偏移属性是完成你期待的布局所需的全部要素。剩下的计算就由浏览器来负责了。

## 百分比值

你也能用百分比值来指定定位属性。在图15-22的第一个例子中，图像定位于包含块左边50%位置的下面。在右边的第二个例子中，img元素被定位，这样，它将一直显示在包含块的右下角。

```
img#A {
```



**警告：** 在定位元素位于初始包含块（html元素）下方时要小心。虽然你可能希望它位于整个网页的下方，但事实上浏览器将元素置于浏览器窗口的下面的角落。结果往往是不可预料的。如果你希望元素位于网页下方的角落，那就把它放在文档源代码末尾的包含块元素中，然后从那里开始布局。

```
position: absolute;
top: 50%;
left: 0%; /* the % symbol could be omitted for a 0 value */
}
img#B {
position: absolute;
bottom: 0%; /* the % symbol could be omitted for a 0 value */
right: 0%; /* the % symbol could be omitted for a 0 value */
}
```

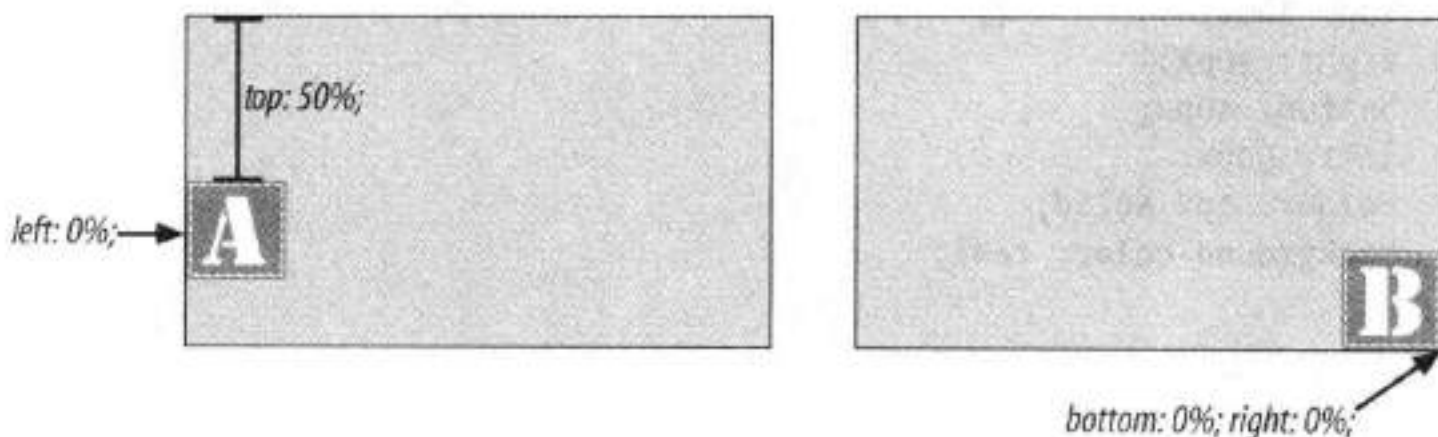


图15-22：使用百分比值将元素置于包含块的下面的角落

尽管这里的示例同时指定一个垂直偏移和一个水平偏移，但是通常只为一个定位元素提供一个偏移，例如，使用left或者right属性将它向左或向右移动到一个空白边。

在练习15-4中，将使Jenware主页变得有趣，这次使用绝对定位。

#### 练习15-4 绝对定位

在练习中，使用绝对定位来添加奖励图像到网站，并创建两栏布局。先在文本编辑器中打开练习15-3之前的jenware.html (jenware\_ch15.html)。你可以使用浮动图像和水平菜单，从单列布局开始。

1. 假想Jenware.com赢得了“本周最可怕网站”大奖，现在可以有选择地在主页显示一些横幅。因为它是一个新内容，所以我们需要添加到标记文本中。又因为它是不重要的信息，所以将在版权段落之后，在一个新的div元素中添加图像。

```
<div id="award">

</div>
```

虽然它在文档源代码的最后，但并不意味着它需要显示在网页的下文。通过添加用来定位的样式表，我们可以使用绝对定位将#award div置于醒目的浏览器窗口的左上角，如下所示：

```
#award {
position: absolute;
top: 35px;
```

```
left: 25px;
}
```

保存文档并查看（如图15-23所示）。把浏览器窗口缩小到很窄，你可以看到那个定位的获奖图像与标题内容交迭，图像随着网页的其他内容滚动。试用其他的偏移属性和值，感觉浏览器窗口（或准确术语中的“初始包含块”）中的定位。



图15-23：绝对定位的获奖图像

2. 在练习15-3中，使用浮动创建了两列布局。现在用绝对定位完成同样的事情。在这个例子中，我们会给Testimonials设定固定宽度，并且使Products盒子填满剩余的空间。这也是我想让你了解的另一种布局方法。

对于现在的文档，如果对Testimonials div定位，它就会相对于浏览器窗口定位，但这不是我们想要的。我们希望它总是显示在#nav div下面，所以将会在#nav之后创建一个新的包含块，以包含products和testimonials div，并且会把它当做新的定位环境。

现在需要对标记做一些改变。把#products和#testimonials包含在一个新的div中，并将它的id设置为“content”。文档的结构看起来应该是下面的样子：

```
<div id="content">
 <div id="products"> ... </div>
 <div id="testimonials">... </div>
</div>
<p class="copyright">...</p>
```

3. 通过“不移动相对定位”的小技巧，现在能简单地将“content” div转变为一个包含块。

```
#content {
 position: relative;
}
```

4. 完成后，可以把#testimonials盒子定位到#content div包含块的右上角。像下面一样，给#testimonials规则添加同样的右上角属性。此外，给content设置14ems宽度。将顶部空白调整为0，将左右两边空白从10%改变为1em。

```
#testimonials {
 ...
 Margin: 1em 10% 0 1em;
 position: absolute;
 top: 0;
 right: 0;
 width: 14em;}

```

5. 保存文档，在浏览器中查看。你可以看到Testimonials盒子在右角，刚好在Products盒子的右上方。下一步给Products盒子添加一个右空白，从而给Testimonials添加一些空间。但是应该添加多少空间呢？让我们像Web极客一样计算一下。
6. Testimonials盒子大约有3.5ems的左空白（55px）、14em宽的内容，1em的右填充以及1em的右空白，总共19.5em。如果把#products的右空白设置为20.5em，就会使Testimonials盒子在两列之间有了一点空间。使用TRBL快捷属性，如下所示：

```
#products {
 ...
 margin: 1em 20.5em 1em 1em;
 ...
}
```

保存文档，并在浏览器中查看（图15-24）。调整窗口尺寸，并且与上一个流动列的例子比较，看看现在的盒子怎么样。





图15-24：由绝对定位testimonials盒子创建的两列格式

真实性校对

在使用绝对定位轻易创建多列布局后感到兴奋之前，我要指出，这个练习描绘的是理想情况——旁边的列保证比主要内容短，而且也不用担心重要的页脚。如果侧栏由于更多的testimonials div元素而变长，它将会与同一网页中的全宽度页脚交迭，这就不是理想情况了。请思考这个问题，我们还有很多内容没有讲述，这些将在第16章中看到。

堆积顺序

在结束绝对定位之前，我还要介绍最后一个相关的概念。如我们所见，绝对定位元素与其他元素交迭，所以它遵循一个规则，多个定位元素可能会一个接一个的排列。

默认情况下，元素按照在文档中出现的顺序堆积，但是你能使用z-index属性来改变堆积顺序。将z轴描述为垂直于网页的线，就像从你的鼻尖穿过这张纸到另外一面一样。

## z-index

属性值: 数字|auto|inherit

默认值: auto

适用对象: 可定位元素

是否可继承: 否

z-index属性的值是一个数字（正的或负的）。数字越大，元素显示在这个堆的层就越高。为了更清楚地说明，让我们看一个例子（如图15-25所示）。

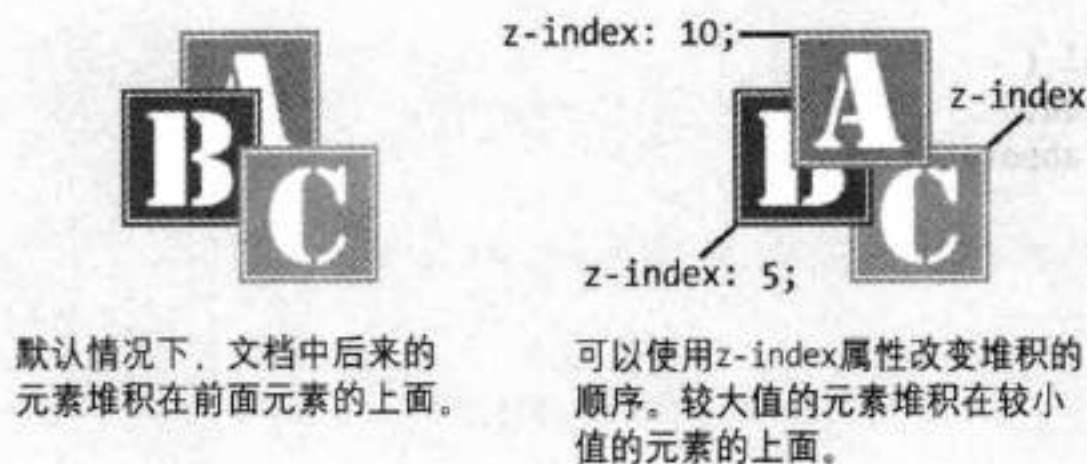


图15-25：使用z-index属性改变堆积顺序

有三个段落元素，每个包含一个字母图像（分别为A、B和C），通过定位方法，它们交迭在网页上。默认情况下，段落“C”将在最高层，因为在源代码中它出现在最后。然而，通过给“A”和“B”段落设置更大的z-index值，我们可以强制它们以我们喜欢的顺序堆积。

注意，z-index值不需要连续，它们并不精确相关。需要关注的是：数字越大的元素在堆中的位置越高。

## 标记文本

```
<p id="A"></p>
<p id="B"></p>
<p id="C"></p>
```

## 样式表

```
#A {
 z-index: 10;
 position: absolute;
 top: 200px;
 left: 200px;
}

#B {
 z-index: 5;
```



```
position: absolute;
top: 225px;
left: 175px;
}

#C {
 z-index: 1;
 position: absolute;
 top: 250px;
 left: 225px;
}
```

事实上，`z-index`属性并不在多数网页布局中需要，但是如果你需要，就应该知道有这个属性。如果你想保证定位元素一直在顶层，只需要设置一个很高的`z-index`属性值即可，例如：

```
img#essential {
 z-index: 100;
 position: absolute;
 top: 0px;
 left: 0px;
}
```

## 固定定位

我们已经学过了相对定位和绝对定位，现在开始学习剩下的方法：固定定位。

在大部分情况下，固定定位与绝对定位工作方式相同。重要的区别在于固定元素的偏移值总是相对于浏览器窗口（或其他视口），这意味着，即使网页的其他部分滚动，定位元素还保持固定。与之对比的，你可能记得，当你在练习15-4中滚动Jenware网页时，获奖图像随着文档滚动——即使相对于初始包含块定位（即浏览器窗口）也是如此，与固定定位的位置不同，看明白了，是固定的位置。

固定元素常常用于顶部、底部或侧面的、位置不变的菜单，所以固定元素总是可见，即便内容在滚动。请阅读侧栏“在移动设备上，小心`position:fixed`”，以避免潜在问题。

让我们使用固定定位来完成Jenware网页的获奖图像，并进行区分。

警告：IE 6不支持固定定位。

## 练习15-5 固定定位

这个很简单。打开Jenware页面，修改样式表，将#award div改为fixed定位，而不是absolute定位。

```
#award {
 position: fixed;
 top: 35px;
 left: 25px;
}
```

保存文档并在浏览器中打开。当滚动网页时，你可以看到award部分固定在浏览器窗口的位置，并不滚动（如图15-26所示）。



图15-26：当文档滚动时，获奖图像停在浏览器窗口左上角的相同位置

现在已经介绍了所有的基于CSS布局的常用工具：浮动和三种定位方式（相对、绝对和固定）。在开始把它们用于第16章中各种设计方法和模板时，你应该完全理解它们的工作方式。

## 自我测验

在继续前进之前，花些时间来看看本章的原理掌握得如何。

1. 关于浮动元素下列哪个说法不正确？
  - a. 所有浮动元素表现为块元素
  - b. 浮动元素倚着容器元素的填充的边缘定位
  - c. 内联元素在浮动元素周围流动，但元素盒子不变
  - d. 必须给浮动块元素提供width属性

## 在移动设备上，小心

### position:fixed

position:fixed属性会在移动设备浏览器上引起诡异的问题。一些浏览器把这个属性当做static，让它随着其余内容滚动。一些支持这个属性的浏览器会将固定元素滚动到屏幕外，但是当滚动停止时，又会把固定元素贴到相应的位置上（至少有一个浏览器算错了相应的位置），结果带来了尴尬的用户体验。而其他浏览器看起来更难受。

虽然有解决的办法，但其实也有缺陷。一个方案是禁止用户缩放页面，但这会移除一个有用的功能。另一个方案是使用JavaScript来创建正确的固定定位效果，但是这有一定的复杂度，而且可能带来不兼容。最好的方案是仔细考虑你是否需要使用固定元素，如果需要就使用JavaScript。

要想了解这个问题，并且了解JavaScript解决方案，我推荐Brad Frost的文章“Fixed Positioning in Mobile Browsers”，位于“[bradfrost-web.com/blog/mobile/fixed-position/](http://bradfrost-web.com/blog/mobile/fixed-position/)”。因为设备支持的问题变化很快，所以一定要搜索最新的推荐方案。



2. 下列哪个样式规则不准确？为什么？
- a. `img { float: left; margin: 20px; }`

b. `img { float: right; width: 120px; height: 80px; }`

c. `img { float: right; right: 30px; }`

d. `img { float: left; margin-bottom: 2em; }`
3. 如何保证“footer” div从浮动侧栏下方开始摆放？
4. 写出与下列描述最匹配的定位方法名称（一个或多个）（static, relative, absolute或fixed）。
- a. 相对于包含块定位元素。

b. 从普通流中移除元素。

c. 相对于视口定位元素。

d. 定位元素可能与其他内容交迭。

e. 在普通流中定位元素。

f. 元素在普通流中占据的空间将被保留。

g. 元素在普通流中占据的空间将被关闭。

h. 你可以使用z-index改变堆积顺序。

i. 相对于普通流中的原始位置定位。

## CSS回顾：浮动和定位属性

这是本章中属性的总结，按照字母顺序来排列。

属性	描述
float	将元素移动到左边或右边，允许后面的文本环绕流动
clear	防止元素紧挨着浮动元素布局
position	指定应用到元素的定位方法
top, bottom, right, left	指定各个边的偏移值
z-index	指定在交迭定位元素堆中的显示顺序

# 第16章 使用CSS进行网页布局

既然你已经明白使用CSS浮动和定位在网页周围移动元素的规则，那就可以把这些工具用在一些标准网页布局中。本章关注各种基于CSS网页设计的方法，并提供一些简单的模板，它们能让你用自己的方式构建两栏和三栏网页。

开始之前，必须要讲的是，表面上使用CSS创建多列布局可以有无穷无尽的变化形式。本章将成为一个“初学者工具箱”。这里出现的模板已被简化了，可能不会在各个情况下都起作用（虽然我已经试着指出各个的相应不足）。你也可能知道更好的方法。事实上，因为关于CSS网页布局有太多的内容，所以我在网上提供了更多的例子和教程的信息。

## 网页布局策略

在开始分析CSS布局前，先来谈谈用于构造一个网页的各种方法。正如你所知道的，网页需要在各种尺寸的浏览器上显示，从小小的手机屏幕到影院显示器。此外，用户还可以调整自己的文字，这也会影响一个页面的布局。随着时间的推移，几个标准的页面布局方法已经出现，它们通过不同的方式解决了问题：

- 固定布局保持在指定像素宽的网页区域内，而不管浏览器或文本尺寸如何。
- 流动（或液态）布局随着浏览器窗口缩放。
- 弹性布局随着文本尺寸而调整。
- 混合布局结合了固定区域和可扩展区域。

让我们看看每种策略如何工作，并且了解它们各自的优劣势。

### 本章内容

固定、液态和弹性网页布局  
使用浮动的两栏和三栏布局  
使用浮动的源独立布局  
使用绝对定位的三栏布局  
从底部到顶部的“人造”列



## 最佳行长

行长是一行文本中单词或字母个数的计量。拇指规则是指最佳行长为10~20个单词或60~75个字母。

当行长太长时，文本将变得难以阅读。不仅很难集中精神读完这一行，而且还需要费力地去找下一行的开头。

行长是探讨哪个布局技术更优问题的核心。在液态布局中，当浏览器尺寸很宽时，行长可能太长了。在固定宽度设计中，在宽度窄且固定的行中，如果文本的字体尺寸很大，那么行长就短得难用。然而，本章稍后介绍的弹性布局，即使当文本字体尺寸变大时，也能提供可预计的行长。这使得它成为考虑平衡设计和易用性时，很受欢迎的选择。

## 固定布局

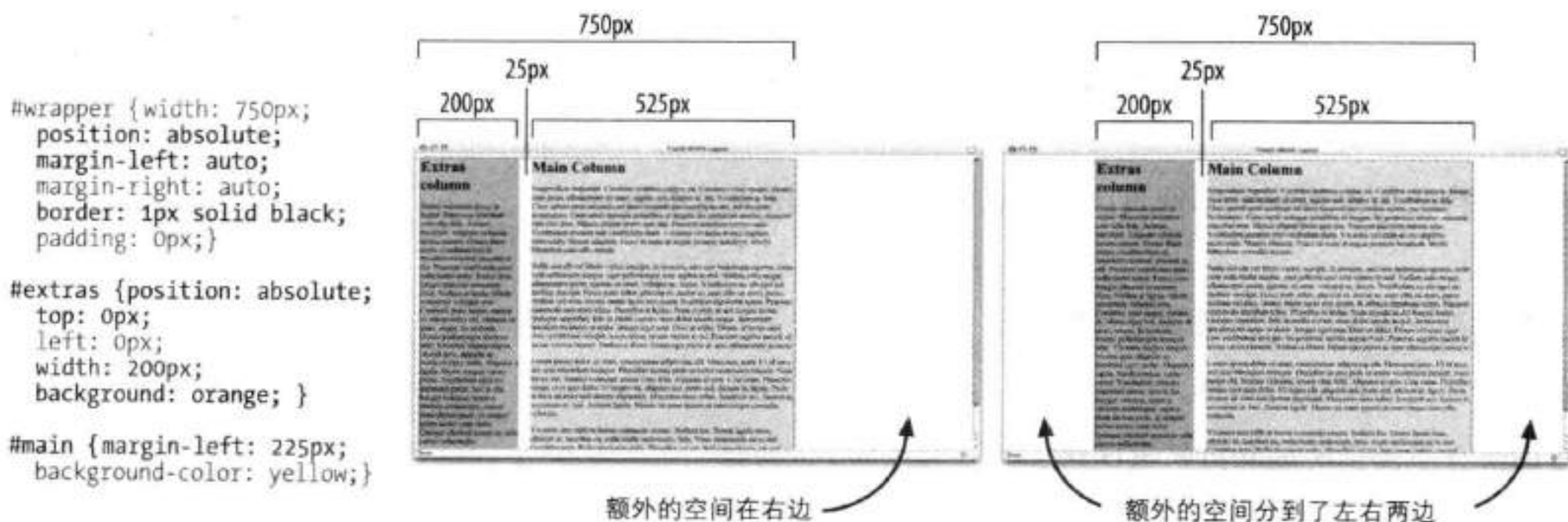
固定布局，如名称所暗示的，宽度保持固定在由设计师指定的具体像素上。同样，设计师可以控制页面元素的关系，如对齐、行长等（见侧栏“最佳行长”）。由于人们习惯首先在桌面显示器上查看网页，而且设计师总是想让网页在各个屏幕上看起来都一样，所以这个布局方法逐渐流行起来。但是如你所知，时代在变，我们不再做这些假设，也不再争取让每个像素都完美。

当设置网页为某个宽度时，你需要决定两件事。首先，你需要选择一个网页宽度，通常基于普通浏览器分辨率。在写本书时，大多数固定宽度网页都设计为960像素，或者适应1024×768分辨率的显示器。一些设计师总是保持窄布局；但是另一些设计师却用更宽的布局，毕竟显示器分辨率在提高。当然无论如何，这都只是个设计决议。

你还需要决定固定宽度布局将处于浏览器窗口的什么位置。默认情况下，它将在浏览器的左边，多余的空白在它右边。你也可以让网页居中，将多余的空白分割到左右空白边，这样可以使网页看起来好像较好地填满了浏览器窗口。图16-1展示了两个固定宽度布局，但它们在浏览器窗口中的位置不同。

使用固定布局时需要特别注意的是，如果用户的浏览器窗口与页面不一样宽，那么页面右边缘的内容就会被隐藏。虽然可以使用水平滚动栏，但是这样并非一目了然。此外，如果用户为了便于阅读，将文字设置得很大，那么虽然页面的结构并没有改变，但是每一行就可能只有几个字——这样的布局让人看起来很不舒服，完全无法阅读。

图16-1：固定布局的例子（左对齐和居中）





让我们看看固定宽度策略的利弊。

### 优势

布局可预见，且可以更好地控制行长。

比较容易设计和制作。

在写本书时，大多数网页都是这么做的，但是如果用户首先使用非桌面设备来访问网页，就需要改变了。

### 劣势

如果浏览器窗口比网页窄，右边的内容将被隐藏。

在大屏幕上，左边可能会有很大的空白。

文本字体尺寸非常大时，行长将会短得难用。

用户无法控制。

### 如何创建固定宽度布局

通过用像素单位指定宽度属性值，来创建固定宽度布局。典型地，把整个网页内容放到一个div元素中（常命名为“content”、“container”、“wrapper”或“page”），然后设置div元素的具体像素的宽度值。这个div元素也可以设置为浏览器窗口居中。各栏元素的宽度，甚至空白边和填充，也都指定为像素值。在本章稍后将看到这个技术的例子。

### CSS网格框架

从早期图像设计开始，设计师就是用网格来对齐和组织内容的，而网格系统也成为了网页设计师的有用工具。网格是不可见的，它可以把页面分成相同大小的单元，这些单元可以用来决定列、标题、图像应该放在哪里（图16-2）。使用网格单元，不仅可以确保你的内容比例合适，也可以使设计更快。

很多CSS网格框架（可以当做“套件”）已经可以用来精简设计和开发流程。最广为人知的就是960网格系统（960.gs），它可以把960像素宽的页面分成12列或16列。Blueprint（[www.blueprintcss.org](http://www.blueprintcss.org)）和BlueTrip（[bluetrip.org](http://bluetrip.org)）也是相似的基于固定宽度的网格。对于一个液态两列或三列网格，也有Yahoo! YUI12（[developer.yahoo.com/yui/grids/](http://developer.yahoo.com/yui/grids/)）。

随着移动设备的兴起，也有合适的网格系统，包括1140 CSS网格（[cssgrid.net](http://cssgrid.net)）、Skeleton（[getskeleton.com](http://getskeleton.com)）以及Twitter的Bootstrap（[twitter.github.com/bootstrap](http://twitter.github.com/bootstrap)）。

当然，这只是CSS框架的冰山一角，还有很多其他的CSS框架。一定要去搜索查找最新和最好的框架。如果你想做得快一些，使用需要固定的HTML和CSS的框架，它们绝对可以节省你的时间。缺陷是，它生成的代码比手工编写的要臃肿很多，可能会带来不必要的下载流量浪费。因此，一些设计师使用框架来加速设计，但是自己来创建站点最终代码。

如果你对此感兴趣，想了解更多，我推荐Khoi Vinh的书“Ordering Disorder, Grid Principles for Web Design”（[grids.subtraction.com/](http://grids.subtraction.com/)）。



图16-2：使用16列的网格系统设计网页的一个例子



液态网页设计

在液态网页布局（也称为流动布局）中，网页中的网页区域和栏允许变宽或变窄，从而填充浏览器窗口中的可用空间。换句话说，它们遵循普通流的默认行为。不要试图控制内容宽度或换行——文本允许按需要重新流动。图16-3展示了W3C站点（w3.org），这是液态布局的好例子。

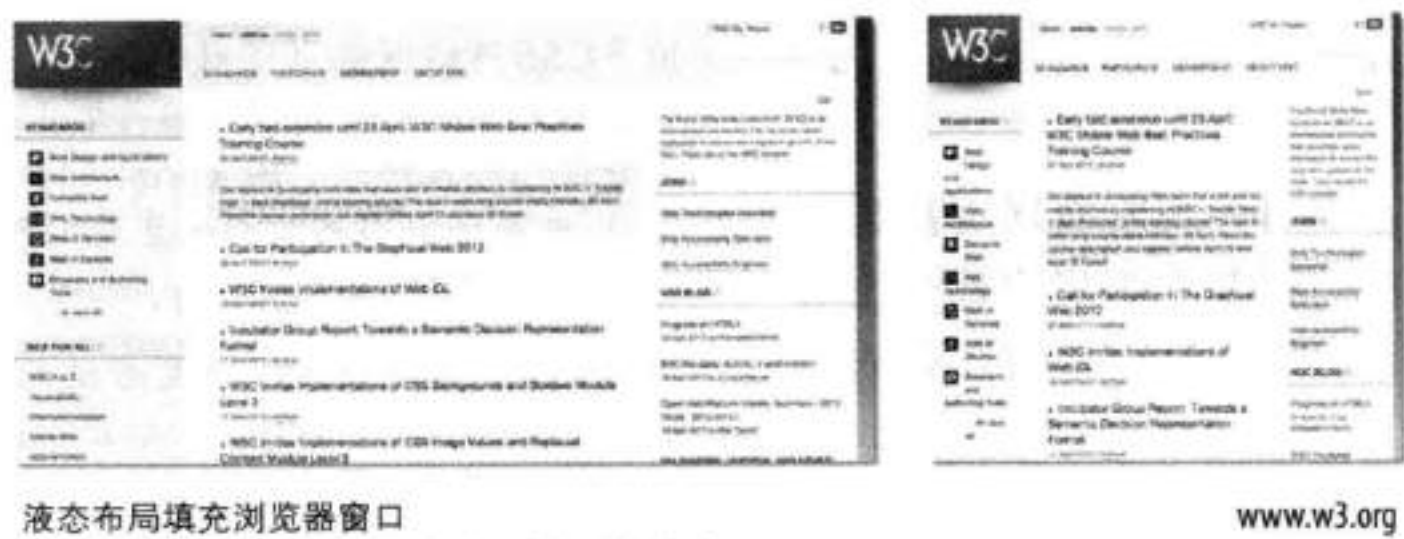


图16-3：液态布局的例子

**注意：** Ethan Marcotte（“自适应网页设计”术语的创造者）在他的文章“Fluid Grids”（[www.alistapart.com/articles/fluidgrids/](http://www.alistapart.com/articles/fluidgrids/)）中讲述了如何使用液态网格来设计W3C网站。这说明使用液态布局并不意味着完全放弃控制。

液态布局是网页设计技术的一个基石。现在网页设计师面对多种多样的浏览器窗和屏幕尺寸，尤其是比传统的台式机显示器小的屏幕，很多设计师都注意设计的灵活性，以填补任何浏览器的宽度。由于对每种屏幕尺寸试图建立一个固定宽度设计是劳而无功的，所以我想液态布局将会复兴。

当然，液态布局有优点和缺点。

优势	劣势
液态网页保持媒体的本性	在大的显示器中，行长太长且不适于阅读
文本填满窗口，从而避免了可能出现的丑陋空白	很难预见。在极端的浏览器尺寸上，元素可能散开，也可能局促拥挤
在桌面浏览器，用户可以控制窗口和内容的宽度	很难获得空白
没有水平滚动条	在计算测量方面，需要使用更多的数学方法

如何创建液态布局

可以通过指定百分比值宽度来创建液态布局。也可以忽略width属性，这样宽度将设置为默认值auto，元素将填满窗口或其他包含元素的可用宽度。

以百分比值指定宽度来创建液态布局。

在这个两栏布局（图16-4）中，每个div元素的宽度已经指定为可用网页宽度的某个百分比值。无论窗口尺寸是多少，主栏总是窗口宽度的70%，右栏填充25%（剩下的5%用于两栏之间的空白边）。当你在练习15-3中使用浮动来创建一列（栏）时，已经了解了这种方法。

流体布局的一个潜在的缺点是使行太长，但通过指定页面的max-width，可以防止布局变得可笑（见本章的侧栏“用maxwidth说‘够了’”）。你也可以使用min-width来防止页面变得太瘦。这两个属性提供了固定布局的一些优势，同时还在最大宽度和最小宽度之间提供了灵活性。

**注意：**IE 6不支持min-width和max-width属性，但是如果真的需要支持，可以使用IE特有的CSS包。可以在Cameron Moll的站点上了解详细信息：[www.cameronmoll.com/archives/000892.html](http://www.cameronmoll.com/archives/000892.html)。

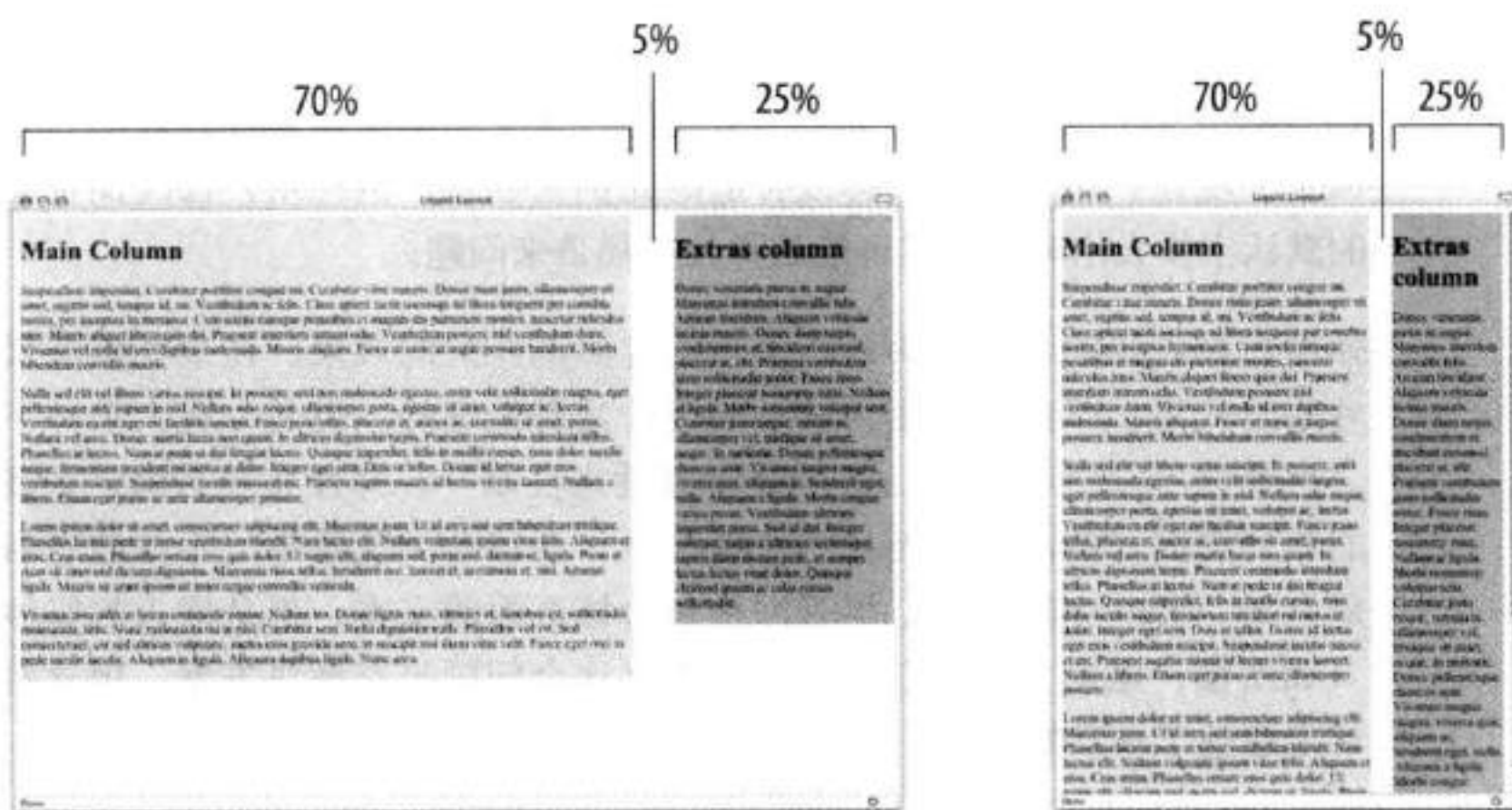


图16-4：使用百分比值的液态布局

```
div#main {
 width: 70%;
 margin-right: 5%;
 float: left;
 background: yellow;
}

div#extras {
 width: 25%;
 float: left;
 background: orange;
}
```

## 弹性布局

第三个布局方法将缩放文本与可预计的行长结合到一起。弹性布局随着文本尺寸扩张或收缩。如果用户放大文本，包含它的盒子将同比例扩大。同样，如果用户喜欢文本很小，容器盒子将收缩来适应。结果是不管文本尺寸多少，行长（每行单词或字母数的术语）保持不变。这是相对于液态布局（行长可能太长）和固定布局（大尺寸文本可能导致每行字母少得难用）的优势。

图16-5显示了由Patrick Griffiths在CSS Zen Garden中设计的“Elastic Lawn”（[www.csszengarden.com/?cssfile=/063/063.css](http://www.csszengarden.com/?cssfile=/063/063.css)），它作为经典例子，在讲解弹性布局时常被引用。注意，当每个例子中的文本尺寸变大时，网页内容区域也变大。注意，在大布局空间中换行依然保持不变。

**注意：**Elastic Lawn的创建者Patrick Griffiths写过关于弹性布局的文章，参见A List Apart（[alistapart.com/articles/elastic](http://alistapart.com/articles/elastic)）的“Elastic Designs”。虽然已经是几年前的文章，但仍不失为一种好方法。



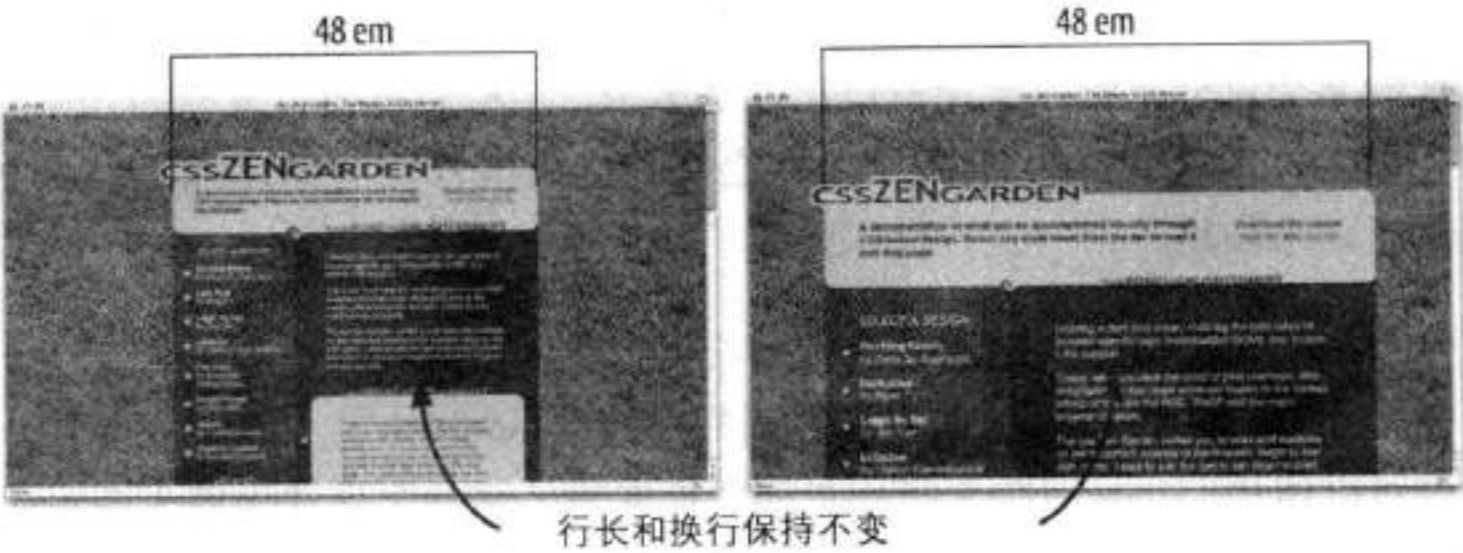


图16-5：由Patrick griffiths在CSS Zen Garden中设计的“Elastic Lawn”是弹性网页布局的一个经典例子

目前大多数浏览器都支持全页缩放功能，这个功能抢了弹性设计的一些风头。现在，所有的网页都以合适的比例来显示，但在用户修改浏览器的默认字体大小时，仍然可能给弹性布局带来问题。

弹性布局设计的支持者喜欢页面与内容协调匹配的效果。在未知屏幕尺寸的时代里，以内容元素为核心的设计是有道理的。但是，在大尺寸下（尽管你可以使用一个max-width属性来控制），弹性布局与固定宽度的布局有同样的问题，而且一般情况下，在移动浏览器中，不如液态布局有用。另一个缺点是，尽管网格包含该页面的文本，但是嵌入的媒体（如图像、电影）并不在网格范围中（这个问题也有解决方案，但这些都是超越了本章的范围）。

现在可以看看弹性布局的正反面：

优势	劣势
允许文本尺寸适当变动时，提供一致的布局体验	图像和影像不能随着文本和布局的其他部分缩放。（但也有方法来实现这一功能）
比液态和固定布局更紧密地控制行长	最大文本尺寸时布局的宽度可能超过浏览器窗口的宽度
	在解决不同设备和不同浏览器的大小时不是很有用
	比固定宽度布局创建起来更复杂

如何创建弹性布局

弹性布局的关键是em，基于文本尺寸的计量单位。例如，对于元素中16像素的文本，em就是16像素。一般总是以em指定字体尺寸。在弹性布局中，容器元素的尺寸也用em指定。这就是宽度对应于文本尺寸的方

法。例如，如果主文本尺寸设置为16像素（当前大多数浏览器的默认尺寸），网页设置为40em，结果网页宽度将为640像素（40em × 16像素/cm）。如果用户放大文本到20像素，网页将变为800像素。

用em单位指定宽度来创建弹性布局。

## 混合布局

将像素、百分比和em混合在一起使用的布局称为混合布局（hybrid layout）。在很多时候，同时使用固定和可缩放的内容区域是很有用的。例如，你也许有一个包含很多广告横幅的侧栏，而且这个侧栏必须是固定尺寸。你可以把侧栏指定为某个像素，并且把它旁边的栏设置为可缩放，使其填满剩余的空间。你应该还记得在练习15-4中创建的网页。

图16-6展示了一个混合布局。左边第二列（栏）被设置为固定像素宽度，主内容区域设置为auto，并且填满窗口的剩余空间。温馨提示：当混用长度单位时（px、%和em），计算页面和元素宽度就会更加复杂。如果有必要，就要这样混用。

图16-6：组合了固定像素宽度和auto的混合布局



```
div#main {
 width: auto;
 position: absolute;
 top: 0;
 left: 225px;
 background: yellow; }

div#extras {
 width: 200px;
 position: absolute;
 top: 0;
 left: 0;
 background: orange; }
```

## 我该用哪个？

如你所见，每个布局方法都有它自己的优点和缺点。布局潮流潮起潮落，我们看到从桌面适合的固定布局，转为适应各种设备的液态布局。你可以发现在自适应网页中，液态布局比较适合小屏幕，但是当在大显示器上浏览网页时，固定布局使你可以随心所欲地控制。或者当你的站点是一个需要固定布局的、复杂的内联网（intranet）应用，并且只需在桌面浏览器使用时，固定布局会更好。所以并没有“绝对正确”的方法，你必须了解所有的选择，并且根据你的内容、目的和主要用途，为你的站点或应用做最好的决定。



## 网页布局技术

本节是你一直在期待的：如何使用CSS创建两栏和三栏布局。本节中的代码例子为你理解布局如何工作提供了一个好的开头，但这并不是通用的解决方案。你的内容部分可能要求更复杂的解决方案。

本节提供下列模板和技术：

- 使用浮动元素的两栏和三栏布局
- 使用浮动和负空白边的源独立布局
- 使用定位的多栏布局

### 使用样例

本节中的示例网页并不漂亮。事实上，我已经将它们内容剥去，直至最简化状态，使结构和策略尽可能的清晰。这里是关于模板的一些注意事项和使用方法。

#### 简化标记文本

在样例中，只包含了最小的标记和样式，对于说明如何创建每个布局已经足够了。所有与布局无关的样式规则都省去了，并把重点放在如何移动元素上。

#### 颜色编码

我已经为每栏都添加了outline（见侧栏“CSS Outline”），你可以看到布局中浮动或定位元素的边缘。主要是含有创建布局的标记和样式的颜色编码，这可以让连接之处更加清晰。

#### 页头和页脚

在这些例子中很多都包含了页头和页脚，但页头和页脚都可以轻易地从一个最小化的两栏或三栏布局中除去，或只除去一个。

#### 个性化

很明显，可以对文本、背景、空白边、填充和边框做很多事，使这些网页更吸引人。一旦你用这些模板搭建了一个框架，你就可以自由地改变计量值，添加你自己的样式。

## 使用浮动元素的多栏布局

浮动是网页中创建栏的主要工具。作为一个工具，它也有缺点，但它是我们当前可以获取的最好工具。可以阅读侧栏“CSS布局的未来”，以了解当前更精巧的方案。

注意：用于本节所有模板的HTML和CSS可以在[learningwebdesign.com](http://learningwebdesign.com)中获取。

### CSS Outline

在本节的例子中，我已经利用了outline属性的优势，来显示浮动和固定栏的边缘。outline看起来像边框，而且语法也一样，但是有一个很重要的不同。outline与border边框不同，它并不是用元素盒子的宽度来计算。它只是在顶部，对其他没有任何影响。这使得outline成为一个检测布局的好工具，因为你可以把它打开或者关闭，而不影响你的宽度值。

outline快捷属性绑定了宽度值（outline-width）、样式（outline-style）和颜色（outline-color）属性，就像边框border一样：

```
div#links { outline: 2px
dashed red; }
```

对布局来说，浮动比绝对定位的好处是它可以阻止内容重叠，并且可以轻易地在网页下面保留页脚内容。基于浮动的布局的缺点是，它们依赖元素在源代码中出现的顺序，虽然有一个工作区使用负空白边，我们将在后面看到这样的例子。

下面的例子展示了使用浮动元素实现两栏和三栏布局的通用策略，并可以作为实现你自己的布局的良好开端。

### CSS布局的未来

层叠样式表用来满足设计师和开发者的需求，有一些新的布局技术，可以从使用浮动和定位元素创建多栏布局中解脱出来。

#### 栏

已经在浏览器实现的最直接的布局改进是将一个元素分到不同的栏里。感谢上帝！你可以指定许多栏（`column-count`），或者指定栏的宽度，使其可以以固定宽度重复，直到用尽所有空间（`column-width`）。关于这个技术有很多内容，你可以在W3C CSS3多栏布局模块中看到（[www.w3.org/TR/css3-multicol](http://www.w3.org/TR/css3-multicol)）。当前的Safari和Chrome支持使用`-webkit-`前缀的CSS栏，Firefox支持使用`-moz-`前缀的栏，Opera和IE 10则无需前缀。

#### Flexbox

CSS的伸缩性盒模型（简称Flexbox）提供了一个很简单的方式来安排元素盒子。例如，你可以将子元素在父元素中排成一行，选择额外的空间，把元素水平或纵向放在中间，甚至改变元素的显示顺序，而不需要借助浮动或者空白偏移技术，也不必进行令人厌烦的计算。使用Flexbox，你可以简单地说，“使其成为一个盒子，并且在它内部，将它的子元素水平或垂直放在中间”。这个技术说起来内容很多，我推荐Stephen Hay的介绍文章（[www.thehaystack.com/2012/01/04/learn-you-](http://www.thehaystack.com/2012/01/04/learn-you-)

[a-flexbox/](http://a-flexbox/)），同Recommendation本身一样。Flexbox在最新的浏览器上都得到了支持，而且无须使用供应商前缀。

#### 网格布局系统

微软实现了CSS属性的一个集，它使你可以为一个元素建立一个行和列的网格，并且将其他元素定位到这个网格上。在写本书的时候，这些技术还都是雏形，但是非常值得留意。可以在W3C获取更多信息（[dev.w3.org/csswg/css3-grid-layout](http://dev.w3.org/csswg/css3-grid-layout)），微软开发者社区也提供了很多资料（[msdn.microsoft.com/library/ie/hh673536.aspx#\\_CSSGrid](http://msdn.microsoft.com/library/ie/hh673536.aspx#_CSSGrid)）。

#### Regions和Exclusions

创造了Photoshop和其他设计产品的Adobe为CSS做出了新的贡献，它提供了布局模块，这些模块复用了页面布局产品中的一些功能。CSS区域（Region）允许内容从一个元素流向另一个元素，与InDesign中文本流动的方式类似。CSS围绕（Eclusion）可以使文本围绕在不规则形状周围，就像你在杂志中看到的。你可以在W3C中阅读更多的新锐功能（[dev.w3.org/csswg/css3-regions](http://dev.w3.org/csswg/css3-regions)和[dev.w3.org/csswg/css3-exclusions](http://dev.w3.org/csswg/css3-exclusions)），也可以在Adobe和HTML站点获取更多信息（[html.adobe.com](http://html.adobe.com)）。



**注意：** 本节所有的布局样例都在栏之间使用空白边。如果你想给浮动元素添加填充和边框，记住调整宽度值。当然，你也可以将box-sizing属性设为box-model（记得使用供应商前缀），这样你就不需要去计算填充和边框了。

记住空白边的TRouBle值顺序：Top、Right、Bottom、Left。

两栏流动布局

在练习15-3中，通过浮动一个元素，并对第二个元素使用空白边来留出空间，创建一个两列的布局。在下面的例子中，我们将所有的元素浮动到一边。你可以根据文档中的源代码顺序，以及你希望栏显示的位置，将栏浮动到左侧或右侧。我们将从一个非常简单的两栏式布局开始。

策略

对两栏都设置宽度，并把它们浮动到左侧。清除页脚，将其保持在页面的底部。基础结构和得到的布局如图16-7所示。

标记文本

```
<div id="header">Masthead and headline</div>

<div id="main">Main article</div>

<div id="extras">List of links and news</div>

<div id="footer">Copyright information</div>
```

样式

```
#main {
 float: left;
 width: 60%;
 margin: 0 5%;
}
#extras {
 float: left;
 width: 25%;
 margin: 0 5% 0 0;
}
#footer {
 clear: left;
}
```

注意

这是一个很简单的例子，但因为这是第一个例子，我会指出几点：

- 请记住，我省略了页眉、页脚和文本样式，使例子尽可能简单。请记住，实际工作比样式下的工作多（没有你无法弄清楚的东西，只是背景颜色、填充等类似的东西）。
- 源文件已被分成4个div，分别针对header、main content、extras和footer。这些标记以它们源代码中的顺序显示。
- #main和#extras已经浮动到左侧。因为它们是浮动的，所以需要分别为它们指定宽度。你可以将你的栏设为你喜欢的宽度。

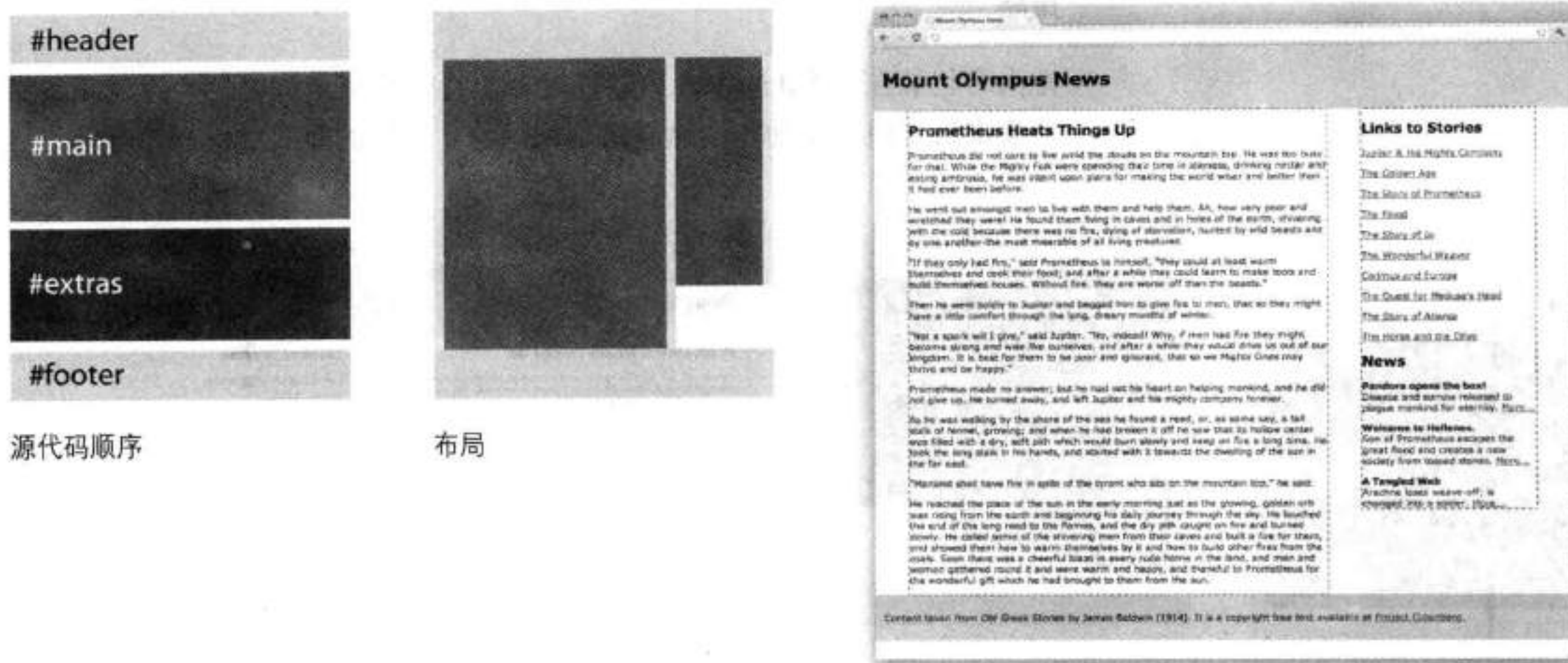


图16-7：浮动的两列

- #main的左、右两侧上都有一个5%的空白。#extras元素只需在右边有一个空白。顶端的空白已被设置为零，以便他们垂直对齐。
- #footer被清除，所以它从浮动内容的下方开始。

**注意：**你也可以将一栏浮动到左边，将另一栏浮动到右边，效果一样。

### 使用max-width说明“够了”

液态布局非常不错，因为它可以适应屏幕或者浏览器窗口的大小。我们用了很长时间了解如何使网站适应小屏幕，但是不要忘了还有另一个极端——宽度达到或超过2000像素高分辨率显示器。用户不能使浏览器窗口最大化以填满整个屏幕，虽然可以让浏览器与屏幕一样宽，但是会让栏中的文本难以阅读。

你可以使用max-width属性来结束这个噩梦。对可能变得难念（就像“两栏，液态布局”例子中的#main列）的column元素应用这个属性，或者把整个网页放入一个包装元素中，并且使其与页面宽度相同。

类似地，如果你不想让页面看起来挤在一起，可以使用min-width属性。注意，这两个属性IE 6都不支持。

## 两列固定宽度布局

这一次，使用固定宽度，而不是液态布局。

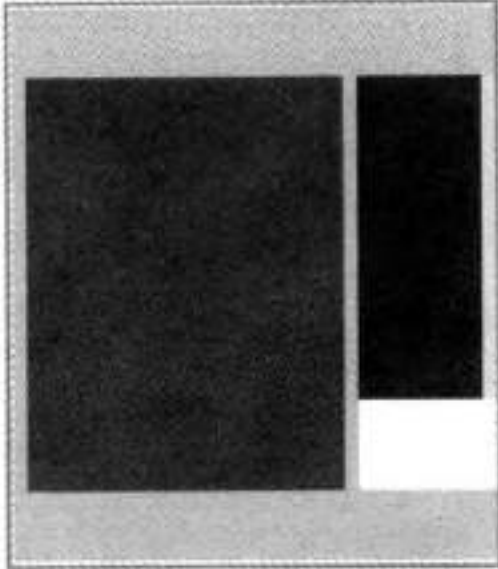


策略

把内容包装在一个div中，可以设置一个特定的像素宽度。为浮动元素指定像素值，但是浮动与清除方法是相同的。得到的布局如图16-8所示。



源代码顺序



布局

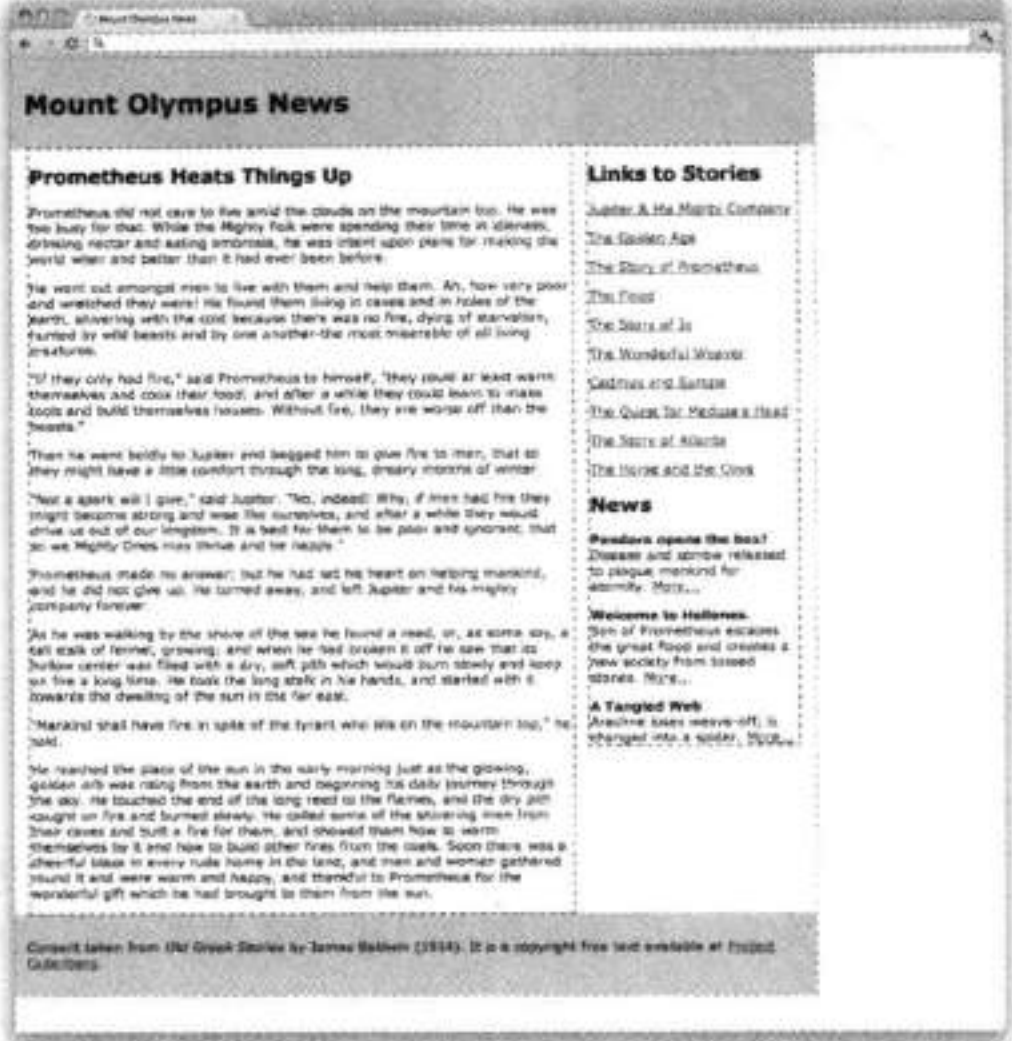


图16-8：使用浮动的固定宽度的两列布局

样式

```
#wrapper {
 width: 960px;
}
#main {
 float: left;
 width: 650px;
 margin: 0 20px;
}
#extras {
 float: left;
 width: 250px;
 margin: 0 20px 0 0;
}
#footer {
 clear: left;
}
```

标记文本

```
<div id="wrapper">
 <div id="header">Masthead and headline</div>
 <div id="main">Main article</div>
 <div id="extras">List of links and news</div>
 <div id="footer">Copyright information</div>
</div>
```

注意

- 所有的内容都包含在#wrapper div中，并且该div已被设置为非常流行的960像素宽。
- 宽度和空白已经改用像素测量单位，注意总像素数不要超过960。如果加起来超过#wrapper容器的宽度，浮动就会下降，看起来很可怕。请注意，如果加上填充或边框，需要减去它俩的宽度值，以保持总宽度相同。

## 两列固定宽度居中的布局

现在，很容易就可以把固定宽度的布局放在中间。

### 策略

#wrapper的左、右页边距设置为auto，这将使整个页面居中。该标记与前面例子中的完全一样。只需要在样式中增加空白声明。得到的布局如图16-9所示。

### 样式

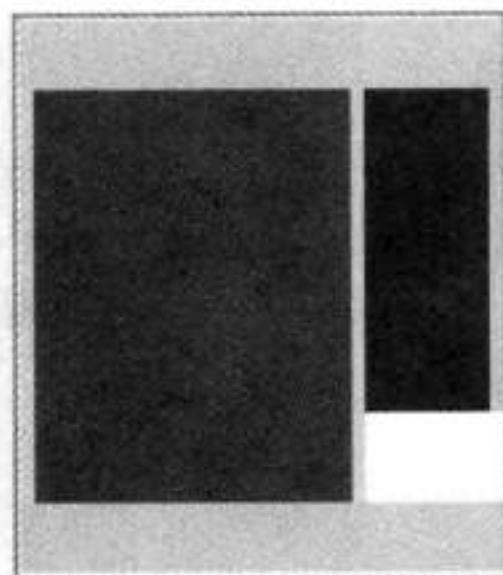
```
#wrapper {
 width: 960px;
 margin: 0 auto;
}
```

### 注意

- 在页面的左右两边设置auto空白，使#wrapper显示在浏览器窗口中间。



源代码顺序



布局



图16-9：固定宽度的布局在浏览器窗口中居中



全宽header和footer

如果你想让#header和#footer与浏览器一样宽，但是仍想让它们之间的内容保持固定宽度并居中（如图16-10所示），可以修改标记文本，使得只有#main和#extras在#wrapper中。其他所有的一切都与“两列、固定宽度、居中”的例子中相同。

```
<div id="header">Masthead and headline</div>
<div id="wrapper">
 <div id="main">Main article</div>
 <div id="extras">List of links and news</div>
</div>
<div id="footer">Copyright information</div>
```

图16-10：header和footer填满了浏览器的宽度，但是它们之间的内容仍然是固定宽度

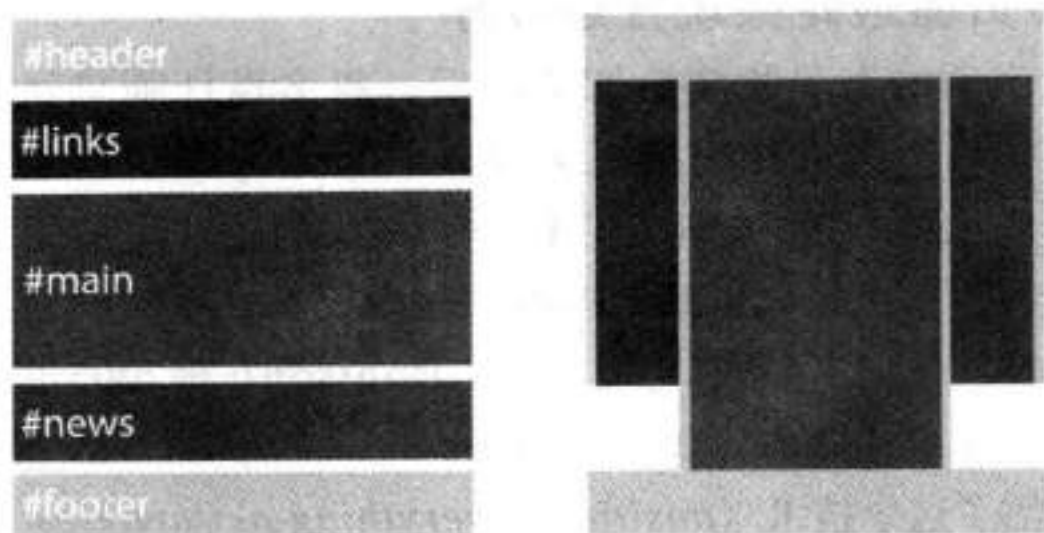


流动的三列布局

我担心你已经有些心不在焉了。现在要完成三列布局，使用相同的原则，但需要一些额外的技巧。在这个例子中，使所有元素浮动到左侧。使用简单的浮动，你会发现我们非常依赖三个浮动元素在源文档中出现的顺序。

策略

给所有三列的元素设置宽度，并把它们浮动到左侧。清除页脚，将其保持在页面的底部。基础结构和布局结果如图16-11所示。



源代码顺序

布局

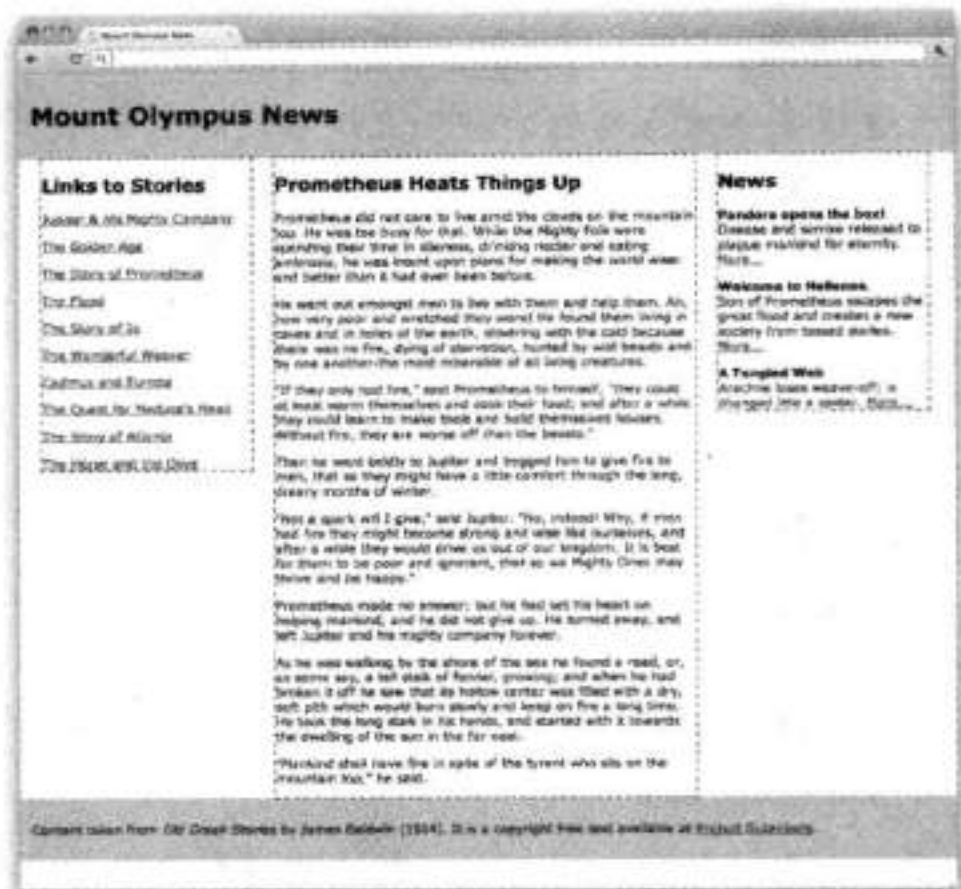


图16-11: 使用三个浮动的流动宽度的三列布局

## 标记文本

```
<div id="header">Masthead and headline</div>
<div id="links">List of links</div>
<div id="main">Main article</div>
<div id="news">News items</div>
<div id="footer">Copyright information</div>
```

## 注意

- 标记文本显示现在有一个包含5个div的文档 #header、#links、#main、#news和#footer。
- 仅使用简单的浮动，如果希望主要内容列出现在链接和新闻列之间，在源文档中#main div需要出现在#links和#news divs之间。（将在“使用负值边距实现任意顺序”的例子中打破这个顺序）。
- 三列都设置了宽度，并且浮动到左侧。必须考虑width和margin的总数不大于100%。

## 样式

```
#links {
 float: left;
 width: 22.5%;
 margin: 0 0 0 2.5%;
}
#main {
 float: left;
 width: 45%;
 margin: 0 2.5%;
}
#news {
 float: left;
 width: 22.5%;
 margin: 0 2.5% 0 0;
}
#footer {
 clear: left;
}
```



## 练习16-1 试试看

我们已经看过很多使用浮动的两栏和三栏布局的例子，包括液态布局或者固定宽度布局。我想让你在这里试试这些技术，从使用三栏液态布局开始。本练习所用的文件 `mountolympus-ex1.html` 在 `learningwebdesign.com` 中专门针对本章的 `materials` 文件夹中。最终的样式在附录A中。样式概要已经包含，但是如果你想关掉这些样式，看看没有样式的布局是什么样子，你可以把它们注释掉（用 `/*` 和 `*/` 包含要注释的部分即可）。

首先，重新安排侧栏，使得 `#links` 在右侧，而 `#news` 在左侧。你没必要修改标记文本，只需要修改一些样式值即可。（提示：调整浮动方向。）记得调整左右两侧的边距，并且清除 `#footer`。

其次，把这个液态设计转化为居中、固定宽度的设计。这里需要添加一些标记（如果不知道怎么做，可以看看两列固定宽度的例子）。最终的网页如图16-12所示。

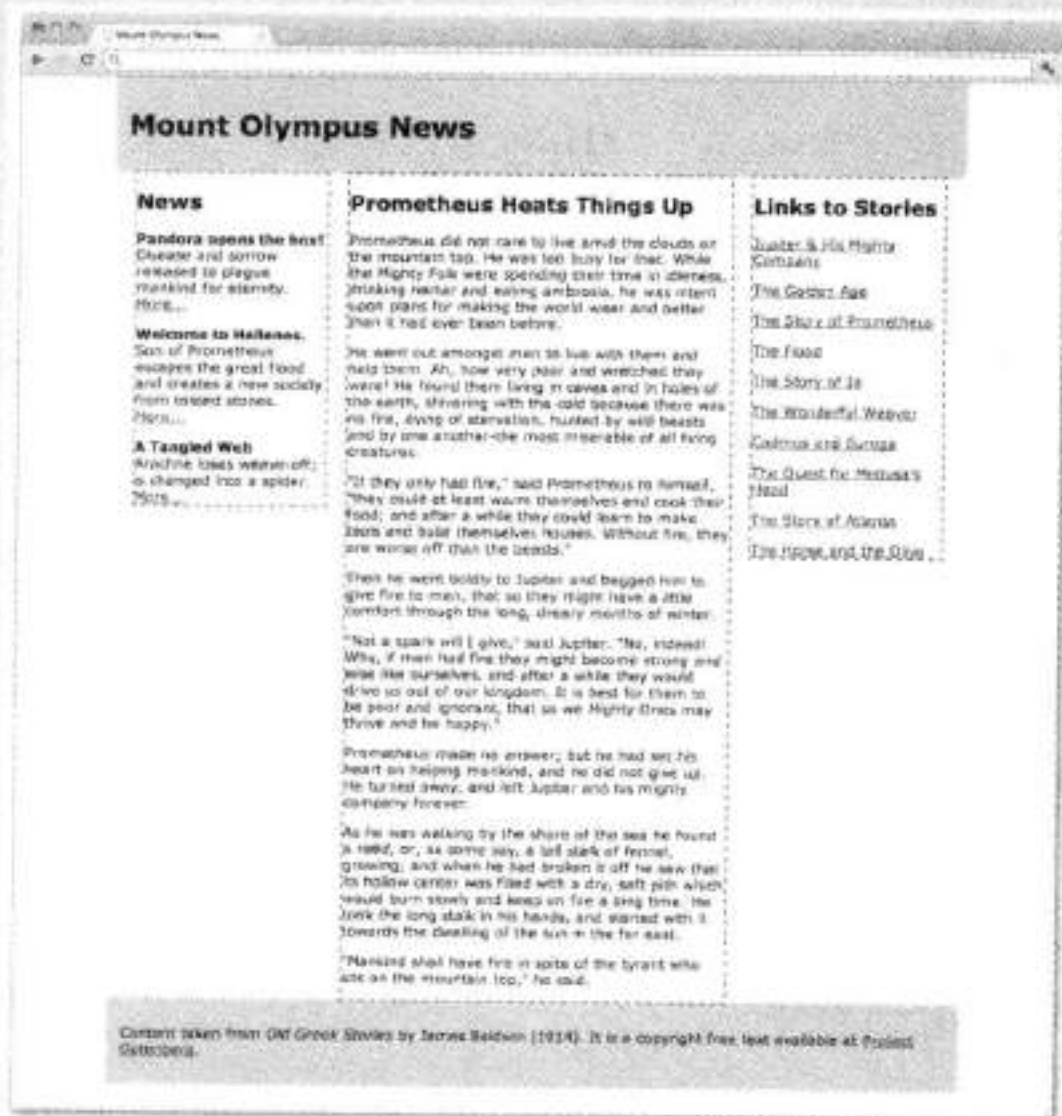


图16-12：左右两栏置换后的固定宽度布局

## 使用负值边距实现任意顺序

基于浮动的布局收到热切关注后，很多设计师想知道，“有没有办法完成三列浮动，但是与源文档中的顺序无关”？答案是“有”！办法的诀窍是使用神奇的负边距值，以及一些数学知识（有一点数学，不会伤害任何人对吧？）。Alex Robison在他2005年经典的文章“[The Search for One True Layout](http://www.positioniseverything.net/articles/onetruelayout/)”第一次介绍了这个技术（`positioniseverything.net/articles/onetruelayout/`）。

### 策略

对所有三个列元素应用宽度和浮动，并使用负边距值来把左栏“拖到”整个页面的左边位置。基础结构以及由此产生的布局如图16-13所示。请注意，虽然 `#main` 在源代码中首先出现，但它是在第二列的位置显示。此外，`#links` `div`（在源代码中的最后）在左边的第一列。这个例子是固定的，但你可以使用百分比值对流体布局做同样的事情。

### 标记文本

```
<div id="wrapper">
 <div id="header">Masthead and headline</div>
 <div id="main">Main article</div>
 <div id="news">News items</div>
 <div id="links">List of links</div>
 <div id="footer">Copyright information</div>
</div>
```

### 样式

```
#wrapper {
 width: 960px;
 margin: 0 auto;
}
#main {
 float: left;
 width: 520px;
 margin-top: 0;
 margin-left: 220px;
 margin-right: 20px;
}
#news {
```

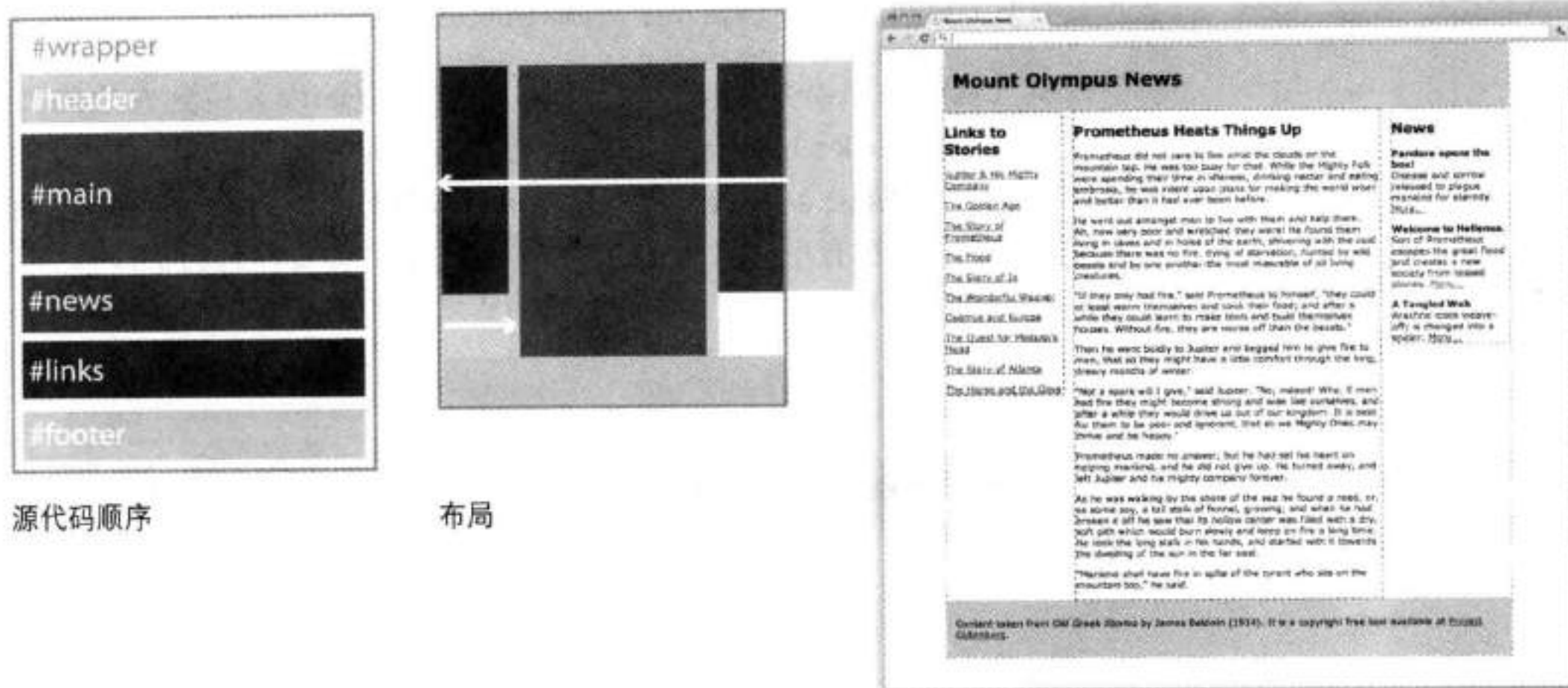


图16-13：一个使用三个浮动效果的固定宽度，三列布局。它看起来像前面的示例，但其实不一样，列出现的顺序与源文档中不一样

```
float: left;
width: 200px;
margin: 0;
}
#links {
float: left;
width: 200px;
margin-top: 0;
margin-left: -960px;
}
#footer {
clear: left;
}
```

## 注意

这需要更多一点的解释，所以看看这是如何一步一步完成的。

在标记文本中，可以看到#main排在第一位的，因为它是最重要的内容，而#links排在最后。整个页面包裹在#wrapper中，以便它可以设置为一个特定的宽度（960像素）。然而，在布局中由左到右的顺序是：#links（200像素宽）、#main（520像素宽），然后是#news（200像素宽）。并且在这些列之间有20像素的空间。

要做的第一步骤是把#main内容移动到中间位置，方法是给其左边添加边距，使其左边腾出空间来容纳左列（200像素）和20像素的空间。因此，编写margin-left:220px。然后，给#main添加一个20像素的右边距，在

**注意：** 如果你需要支持IE 6，给#main添加display:inline; 声明，同时解决IE 6边距增倍的问题。这样做不会影响其他浏览器。



其右侧留出空间。图16-14显示了对#main应用样式后的外观。

下一步（这一步很酷），使用负边距，把你希望的内容拉到左边的列（这里就是#links）。这里的诀窍是找出要向左侧移动的距离。在图16-14中，如果#wrapper足够宽，你可以看到#links移动的距离。我发现这样看布局很有用，因为它清楚地表明，我们需要移动#links的距离，包括源代码中所有在它之前的元素的宽度。

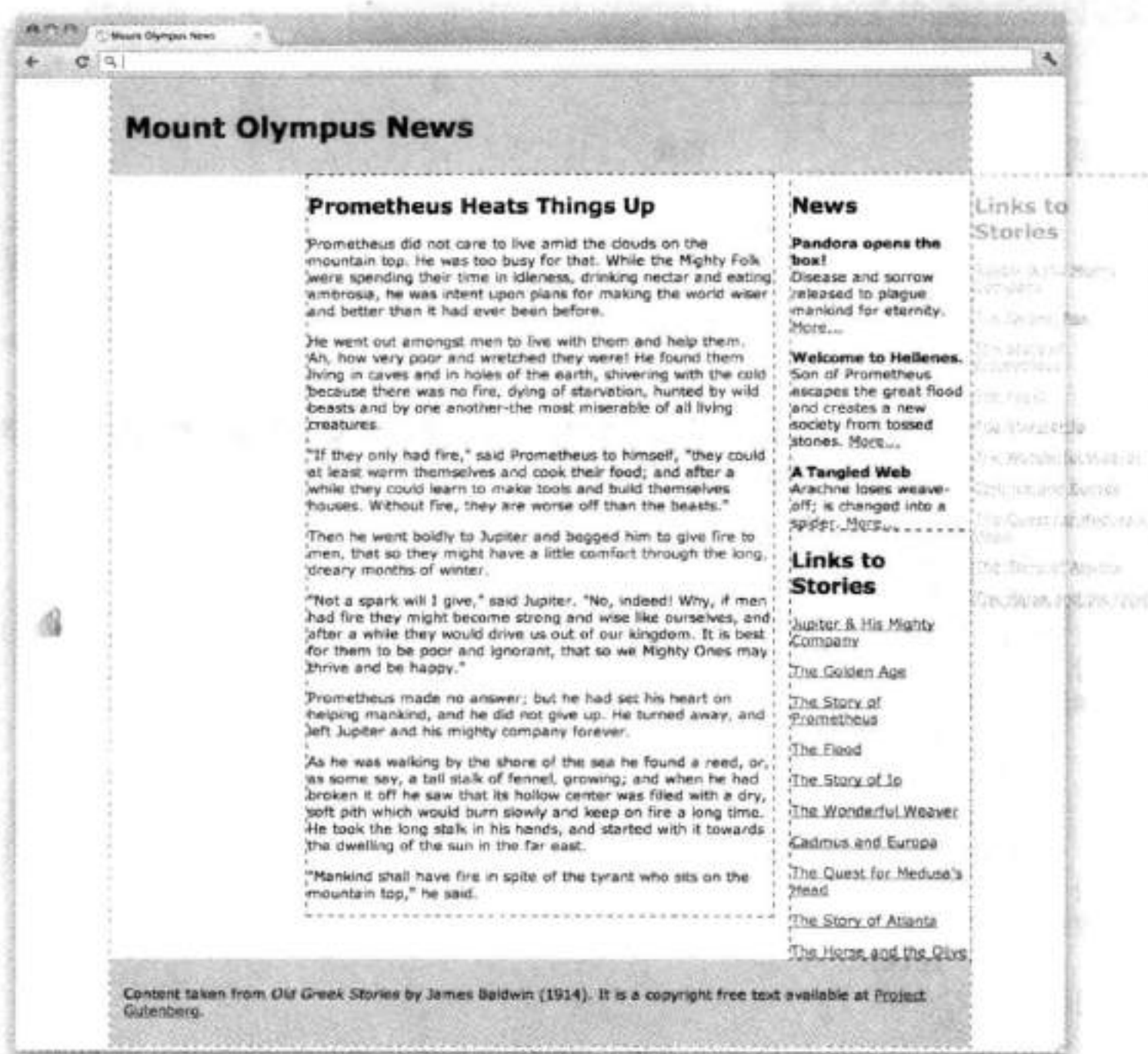


图16-14：使用边距后的布局，应用于中间的（#main）列元素。右侧的阴影框显示链接可能的位置

**警告：** 当你自己工作时，记得在元素盒子总宽度中包含填充和边框，除非你使用border-box盒子尺寸模型。

在此示例中，#main的宽度是520像素加220像素，加上左右两侧边距各20像素，共760像素。#news的总宽度为200像素（没有边距）。这意味着#links div需要被拉回960像素，才能放在左侧列中（margin-left: -960px;）。当使用负边距时，#link就会拉回，最终的布局如图16-13所示。

负边距技术能够以任意顺序定位任意多个列。在练习16-2中，你会得到一个重新安排列（栏）的机会，以使#news显示在左边的列中。

## 练习16-2 使用负边距

现在你已经了解策略，你应该可以编写样式，把#news放在左列，#links放在右列。这个练习与练习16-1使用的HTML源文档相同。注意，为了增加难度，列的宽度已经改变。与前面一样，列之间要有20像素的空间。

如果你喜欢在文本编辑器中处理，*mountolympus-ex2.html*在专门针对本章的*materials*文件夹中。否则，你也可以用铅笔来试着写写下面的样式规则。最终的样式在附录A中。

记住，关键使用负边距是把#news移动到左边，包括源文档中在它前面所有元素的宽度。

```
#main {
 float: left;
 width: 400px;
 /* write your margin declarations below */
}

#news {
 float: left;
 width: 300px;
 /* write your margin declarations below */
}

#links {
 float: left;
 width: 220px;
 /* write your margin declarations below */
}
```

生成的布局如图16-15所示。

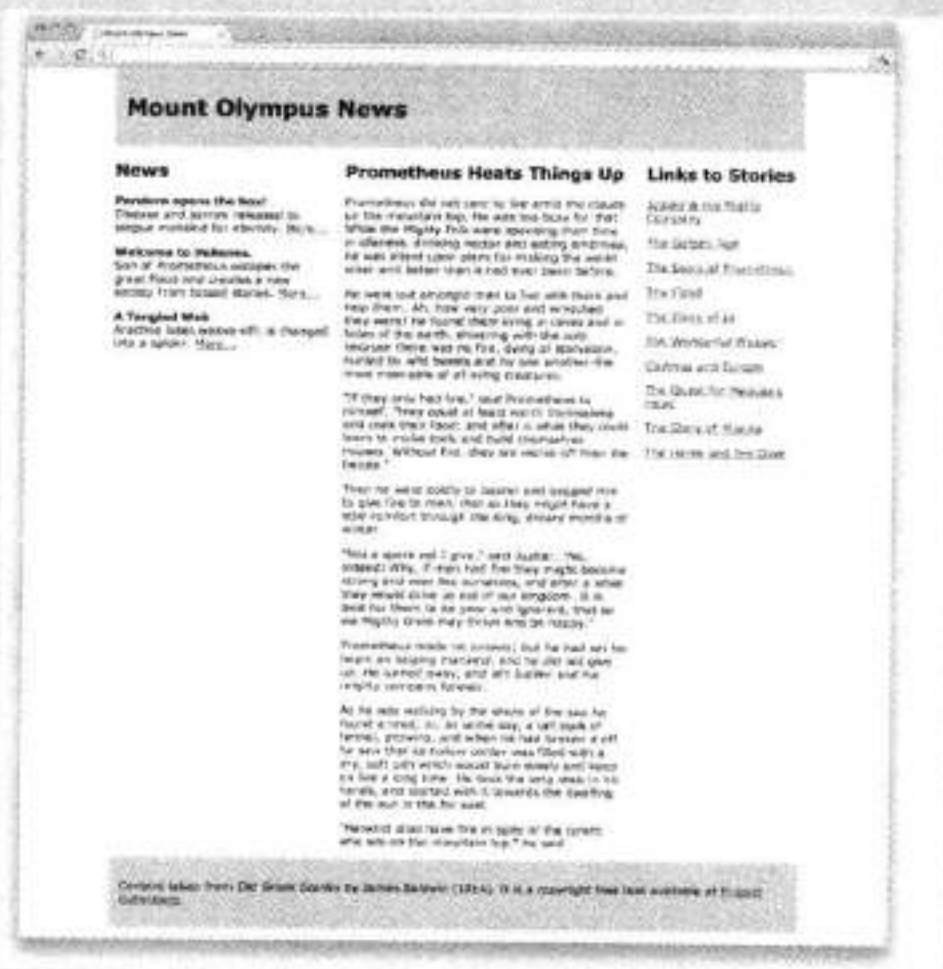


图16-15: 练习16-2的最终三列布局



# 定位布局

我们已经学完了浮动列。创建列的其他方法是在布局中使用绝对定位。在练习15-4中，就已经使用定位和固定宽度列来创建了一个混合的两栏布局。现在将在液态和固定宽度网页上使用定位来创建三列。

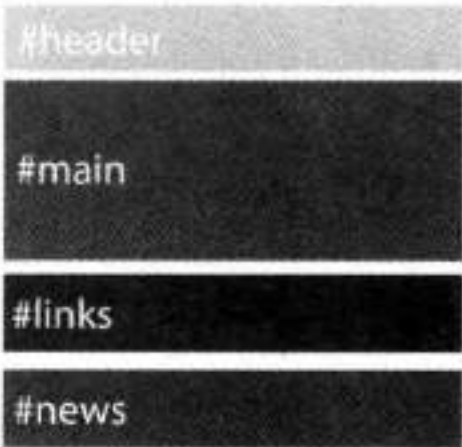
需要注意的是，在这两个例子中，我省略了#footer元素。这有两个原因。首先，当你在一个布局中定位所有元素时（就像在这些例子中那样），它们不再是布局的“参与者”，这意味着页面中没有什么东西可以容纳底部的一个页脚。这时，页脚就会上升到顶部。使用JavaScript可以解决这个问题，但它们超出了本章的范围。

但我们只对两个边栏定位，让中间列留在流中，来容纳页脚。这当然是可能的，但是如果其中一个侧边栏比中间列长，它们就会覆盖页脚内容。在跳过页脚和重叠之间，容易让人凌乱，这就是我选择在这里省略页脚的原因（也是浮动是更为流行的布局技术的原因）。

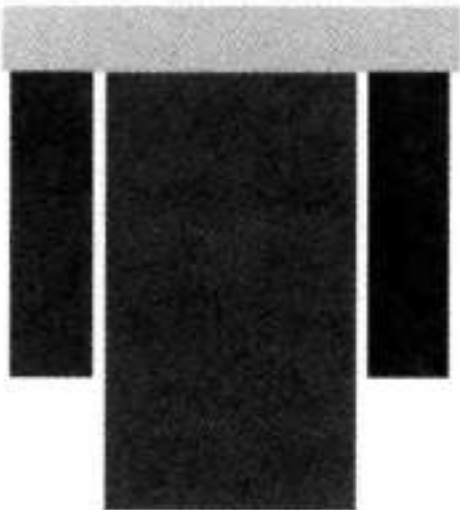
## 三列定位的流动布局

此布局使用百分比值来创建三个可伸缩的列。这个结果如图16-16所示。

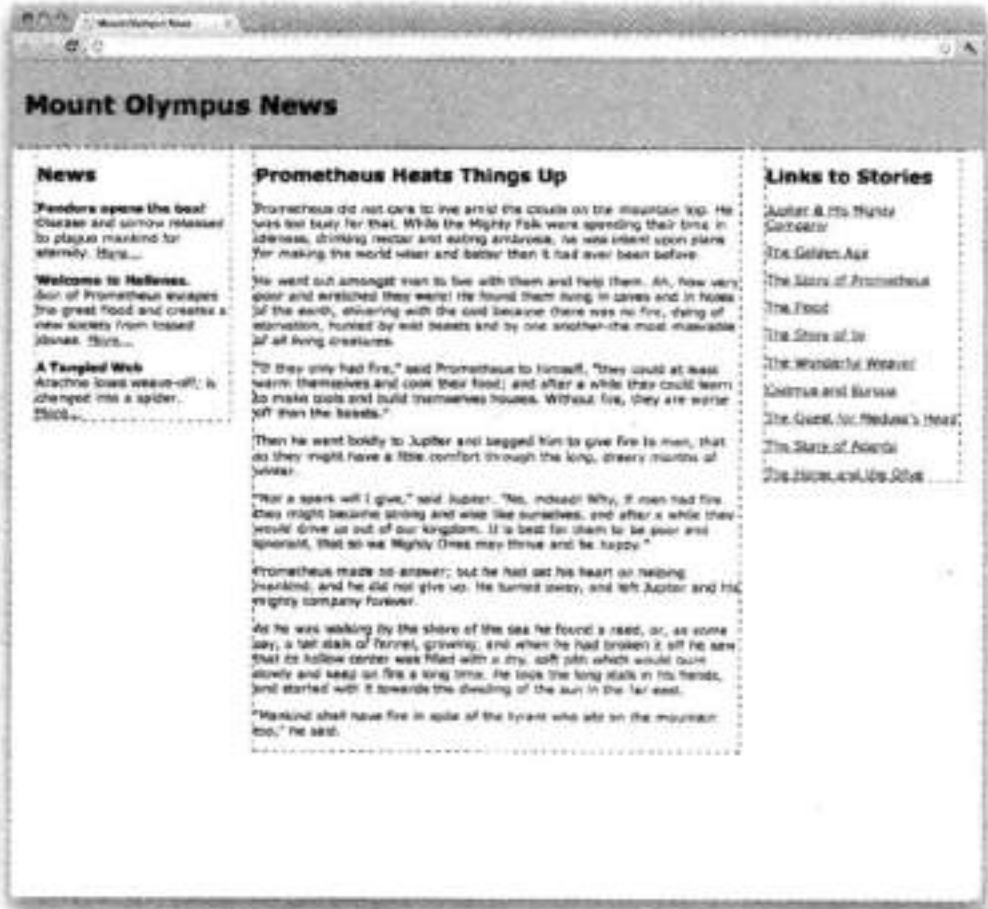
图16-16：三个定位的流动列



源代码顺序



布局



## 策略

把三个div (#main、#news、#links)包含到一个div (#content)中，使其成为包含三个定位列的块。然后给这三个定列元素设置宽度，并在#content 元素中将它们定位。

## 标记文本

```
<div id="header">Masthead and headline</div>

<div id="content">
 <div id="main">Main article</div>

 <div id="news">News items</div>

 <div id="links">List of links</div>
</div>
```

## 注意

我想，你会发现这种布局的样式是相当简单的。

- 因为我们希望列总是从#header下面开始，所以我创建了#content 包含块，并定位列。如果相对于浏览器窗口（初始包含块）来定位，而且header的高度改变，它们可能就在错误的地方了，就像h1文本尺寸变化的结果。通过应用声明position: relative, 使#content div成为一个包含块。
- #main div给定宽度为50%，并使用绝对定位来将其放置在#content div的顶部，并且设置25%的左边缘。这将容纳左列宽度20%，加上左右两边各2.5%的边距。
- #new div定位在#content div的顶部,距左边缘2.5% (top: 0; left: 2.5%; )。
- #links div的位置在#content div的顶部 (top: 0; right: 2.5%; )，距右边缘2.5%。无须计算到左侧边缘的距离，只是把它放在那里就可以了！注意，也可以定位#news和#links列紧邻其各自的边缘，并且使用填充给双方一点点空间。通常总是有多种方法来实现相同的布局。
- 正确布局的唯一的诀窍就是确保width和margin不超过100%。别忘了填充和边框。

## 样式

```
#content {
 position: relative;
 margin: 0;
}
#main {
 width: 50%;
 position: absolute;
 top: 0;
 left: 25%;
 margin: 0;
}
#news {
 width: 20%;
 position: absolute;
 top: 0;
 left: 2.5%;
 margin: 0;
}
#links {
 width: 20%;
 position: absolute;
 top: 0;
 right: 2.5%;
 margin: 0;
}
```



三列、定位、固定

如果你想让你的定位布局中有像素级的控制也不难，可以在这个例子中了解（图16-17）。它不同于以前的流体例子，整个页面都包含在一个#wrapper中，所以它可以是固定并且居中的，而且使用像素值来测量。为了节省空间，我就在这里直接告诉你最终的样式。定位策略没有变。

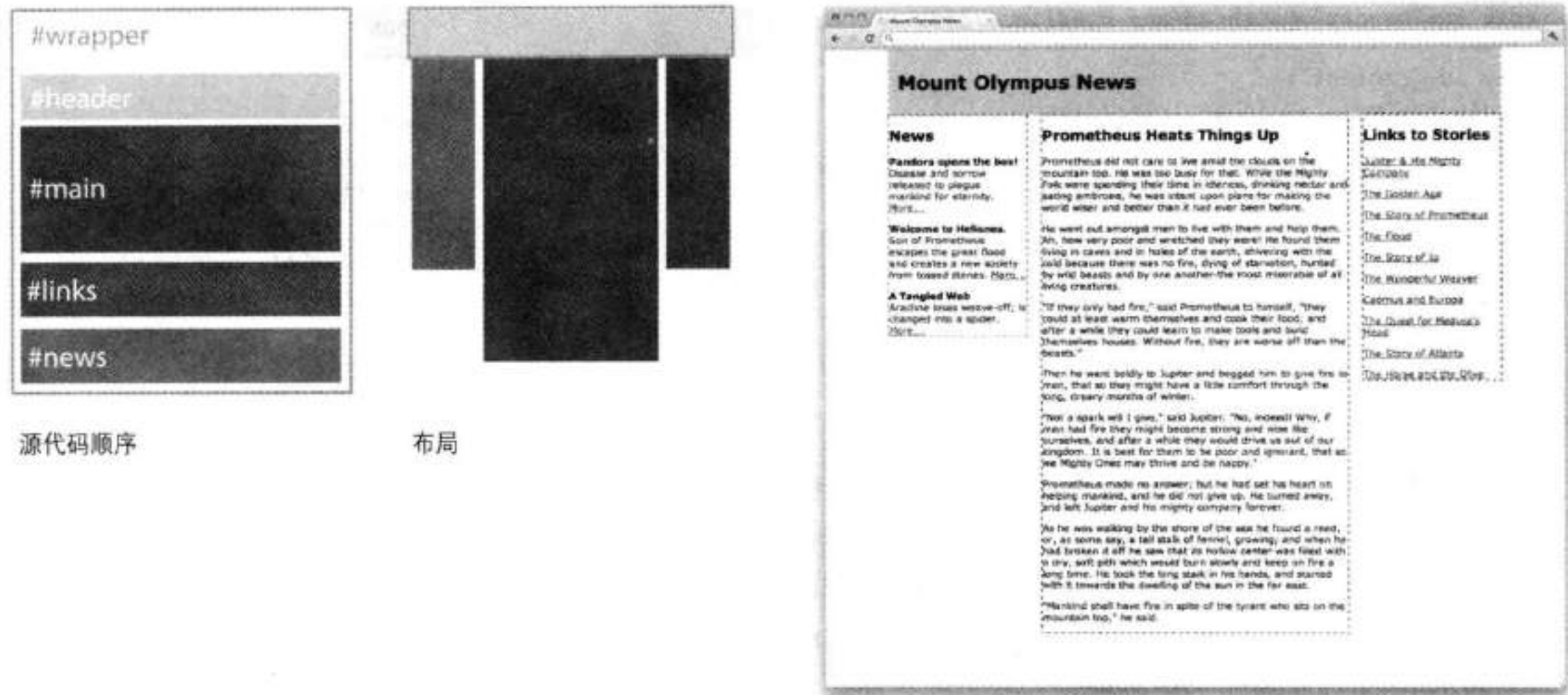


图16-17：居中、固定宽度页面中的三个定位列

样式

```
#wrapper {
 width: 960px;
 margin: 0 auto;
}
#content {
 margin: 0;
 position: relative;
}
#main {
 width: 520px;
 position: absolute;
 top: 0;
 left: 220px;
 margin: 0;
}
#news {
 width: 200px;
 position: absolute;
 top: 0;
 left: 0;
 margin: 0;
}
#links {
 width: 200px;
 position: absolute;
 top: 0;
 right: 0;
 margin: 0;
}
```

## 自上而下的栏目背景

给栏目添加颜色可以有效地进一步强调信息的区别，给网页添加一些色彩。但是如果你看看之前例子的截图中的虚线边框，你会发现列元素column刚好停止在页面的底部。这意味着如果想要从上到下都有背景，就需要仔细斟酌。

遗憾的是，无法设置元素的高度为网页高度的100%，虽然有一些JavaScript工作区和新出现的伸缩式盒模型可以产生全高的栏目元素，但它们超出了本章的范围。

但不用烦恼。有一个称为“伪栏”技巧，可以在本章的任何固定宽度模板中起作用。在这种技术中，你可以在适当的位置使用列颜色创建一个图像，并且将其作为垂直拼贴背景图，应用给页面或者包含元素（如例子中的#wrapper）。伪栏方法在2004年首先被Dan Cederholm在a List apart网站和他的书《Web Standards Solutions》中介绍。

下面是它的工作原理。栏目在图16-18中的阴影是与栏目相同宽度的，带有颜色的水平图像产生的效果。但图像设置为在背景上垂直拼贴，结果是多栏布局位置下的垂直条纹。当然，这种方法只在当栏目或网页的宽度设置为具体像素值时才起作用。待会会介绍流动列背景。

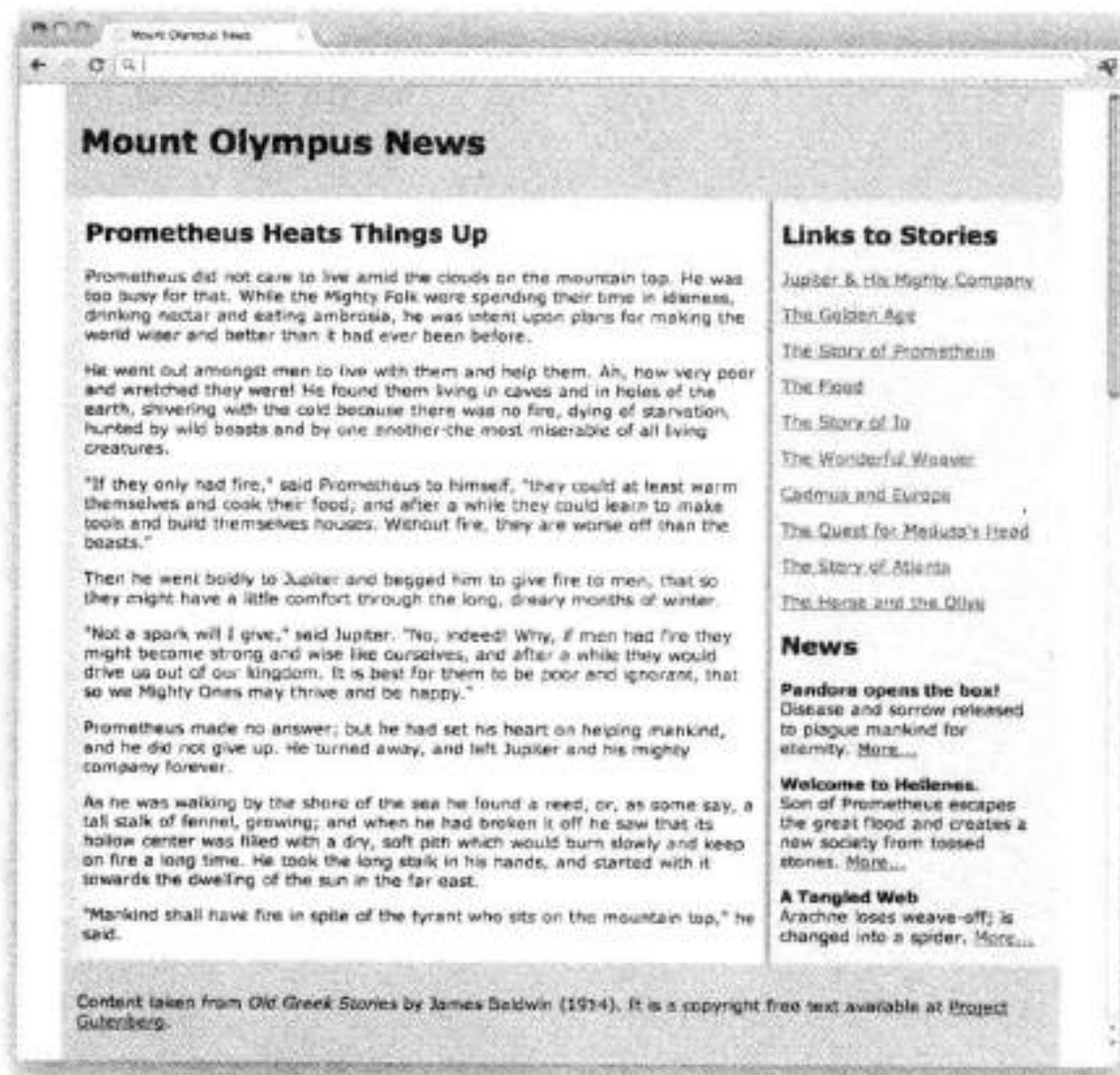
图16-18：小贴图用来创建有颜色的列

你可能认识图16-18中的布局，这是图16-9中创建的两列固定的居中布局。这里，有一个two\_column.png的图像垂直拼贴在#wrapper元素中。

```
#wrapper {
 width: 960px;
 margin: 0 auto;
 background-image: url(two_column.gif);
 background-repeat: repeat-y;
}
```

**注意：**在列浮动后，如果你的布局缺少footer元素来使容器元素打开，对#wrapper应用overflow:hidden，使其围绕浮动拉伸。

two\_columns.png





提示

如果你只是需要简单的规则，而不需要填充背景，Aaron Gustafson (Adaptive Web Design的作者) 有一个好办法。可以给相邻列的外边框设置为相同的宽度，然后使用与该宽度相等的负边距来使边框重叠。这种方法会让最后的边框稍微高一些，它与本页后面的规则看起来相似。

人造列流体布局

现在，你已经了解基本的技术，你可能想知道，如何使其适用于不同宽度的列。秘诀就是宽背景图片和background-position属性。

我们可能不知道流体布局中列的确切宽度，但我们知道列在哪个点分开。用图16-7的两列、流体例子讲解。列在距左边缘67.5%的地方分开 (5%左边距+60%#main宽度+2.5%，也就是列之间空白边距的一半)。

在Photoshop (或者你喜欢的图像编辑器) 中，可以创建一个水平的、比任何监视器都宽的图像——3000像素应该可以了。由于图形只需要几个像素高，并有可能由几个单调的颜色组成，文件大小应保持相当小。当你创建列颜色时，确保它们符合列的比例。在这个例子中，左边列的背景应占图像67.5%的宽度 (67.5% × 3000=2025像素)。

对body元素应用宽图像作为背景图案，并且使用background-position将图像中颜色变化的位置 (67.5%) 与页面上列分开的位置 (也是67.5%) 对齐。以这种方式，图像中的分隔将始终处在列之间的空间上。你应该知道了吧，伪列会随着列扩张和收缩。



图16-19：背景图像定位在两列之间，所以当浏览器窗口变大或变小时，它总是在正确的位置上。图像文件足够宽，以便有足够的图像来填充两列，即使是在最宽的浏览器上。

三个伪列

了解如何处理两列，那么试试看三列？三列也毫无难度，这得感谢Doug Bowman介绍的“liquid bleach”技术。之所以称为“Liquid Bleach”，是

因为Doug将他的博客模板“Bleach”改为液态布局后的称呼。

简单来说，过程与前面基本相同：在容器

标记文本需要两个容器。在这个例子中，将它们命名为#wrapper和#inner：

```
<div id="wrapper">
 <div id="inner">
 <div id="main"></div>
 <div id="news"></div>
 <div id="links"></div>
 </div> <!-- end inner div -->
</div> <!-- end wrapper div -->
```

左列图像在#wrapper中，定位在左列和中间列交汇处（如图16-20所示，26.25%）。右列图像在#inner中，定位在中间列和右列之间（73.75%）。当浏览器窗口调整大小时，背景图像保持在同样的定位点，并且背景色会填充之间的空白。

**注意：** 在第21章中会讨论透明的GIF和PNG。

**注意：** 在#wrapper中，放置多个背景图像也可以获得同样的效果，但是这需要一些额外的标记文本。简单地将一个图像垂直拼贴到左边，将另一个图像垂直拼贴到右边即可。图像必须足够宽，以便可以延伸到列分隔之处，并且延伸到浏览器的边缘。缺陷是，它无法在IE 6~IE 8中工作，在老版本的Firefox中也不行。

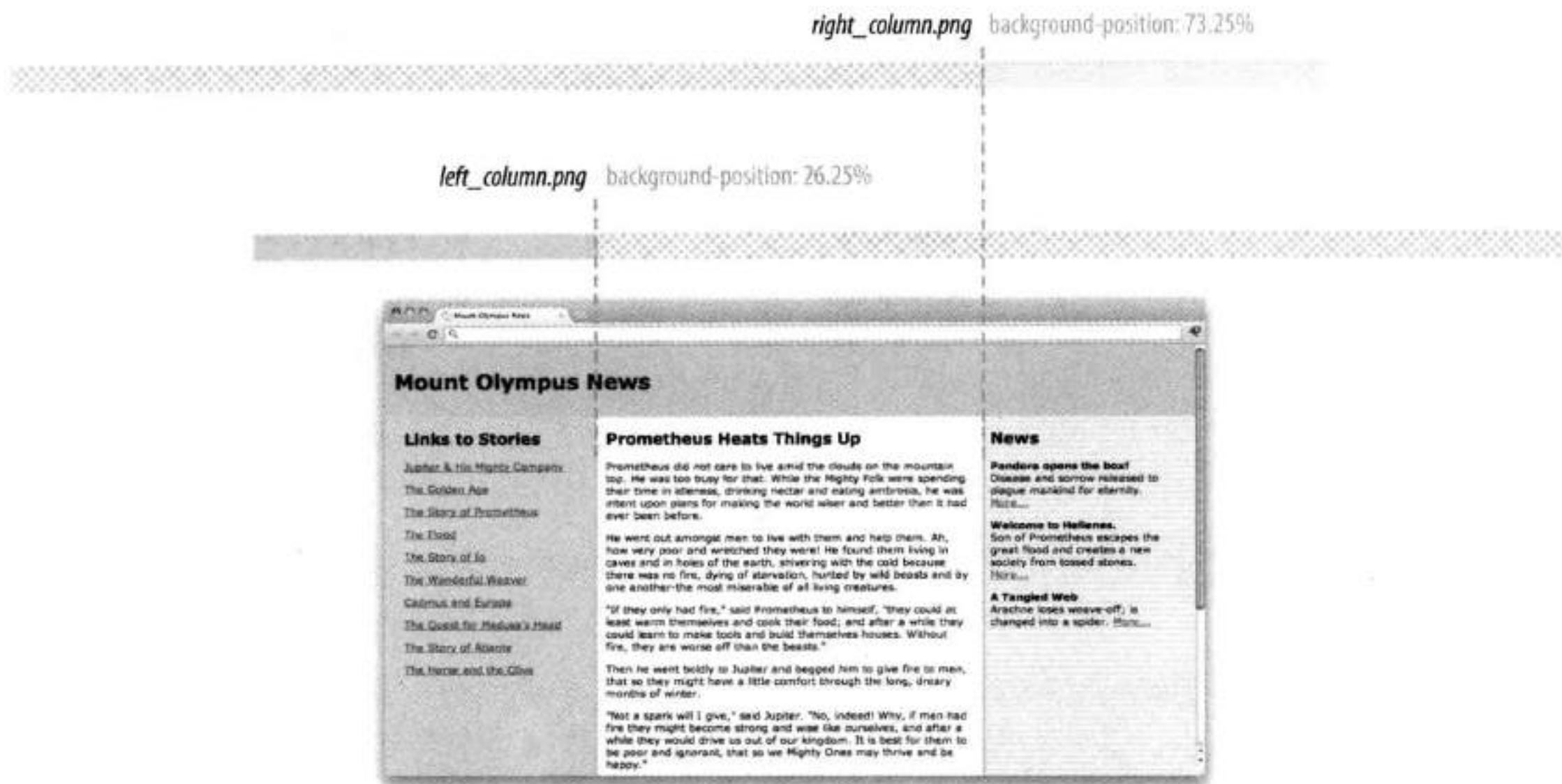


图16-20：流动的、三列布局中的伪列



## 自我测验

如果你在练习中成功地创建了三栏布局，那么你就掌握本章的要点了。下面的问题将保证你更好地了解详细内容。

1. 将各个布局与确定网页区域最终尺寸的因素配对。

固定宽度布局	a. 浏览器窗口
液态布局	b. 字体尺寸
弹性布局	c. 设计师
2. 将各个布局与创建时采用的计量单位配对。

固定宽度布局	a. em
液态布局	b. 像素
弹性布局	c. 百分比值和/或auto
3. 将各个布局与它的潜在优势配对。

固定宽度布局	a. 可预计的行长
液态布局	b. 没有笨拙的“剩余”空间
弹性布局	c. 完美像素的布局网格
4. 将各个布局与它的潜在劣势配对。

固定宽度布局	a. 不适用长的行长
液态布局	b. 网页中的图像不能缩放
弹性布局	c. 在窄浏览器窗口中，页面的右边被切断

# 第17章 过渡、变换和动画

已经介绍过CSS圆角、颜色渐变和阴影，这些在从前都需要创建图形。在本章中将介绍一些CSS3属性可以生成动画交互效果，这在以前只能用Flash或JavaScript来完成。

现在将开始学习CSS过渡，这是一个很好的方法，它可以使样式顺利地从一个过渡到另一个。然后将讨论CSS变换，关于重新定位、缩放、旋转和倾斜的元素，看看如何让它们通过变换来成为动画。本章最后会简要介绍3D变换和CSS动画，这是需要知道的比较重要的知识，但是由于内容非常多，在这里无法详细讲解，所以只会进行简单介绍。

本章的问题是，动画和基于时间的效果无法在纸面上显示，所以我不能在这里展示。但是你可以在[www.learningwebdesign.com/4e/chapter17/figures.html](http://www.learningwebdesign.com/4e/chapter17/figures.html)中找到本章的大多数图像。

## 很容易做到（CSS过渡）

想象一下，一个导航菜单中的链接，当鼠标悬停在它上面时，该链接从蓝色变为红色。背景是蓝色的，当鼠标滑过时，哗！成了红色！它瞬间从一个状态进入另一个状态。现在想象一下把你的鼠标放在链接上，背景逐渐从蓝色变到红色，中间经过一些紫色的阴影。这是相当平滑的。当你移除鼠标时，它会退回蓝色。

这就是CSS过渡的作用。他们一方面很平滑，而另一方面又可以通过填充时间间隔中的帧，从一种状态突然变到另一种状态。动画师称为渐变。当巧妙地使用时，他们可以增加精巧性，并且让你的界面更加漂亮，使用户获得良好的体验。

CSS过渡最初是由Safari浏览器的Webkit团队开发的，但它们现在是W3C的

### 本章内容

- 创建平滑过渡
- 移动、旋转和缩放元素
- 联合过渡和变换
- 关于3D变换的一些概念
- 关于关键帧动画的一些概念



**注意：** 你可以自己阅读CSS过渡模块：[www.w3.org/TR/css3-transitions/](http://www.w3.org/TR/css3-transitions/)。

**警告：** CSS过渡模块正在转变。这里只是写本书时的概览，你应该到W3C站点去查看最新进展。

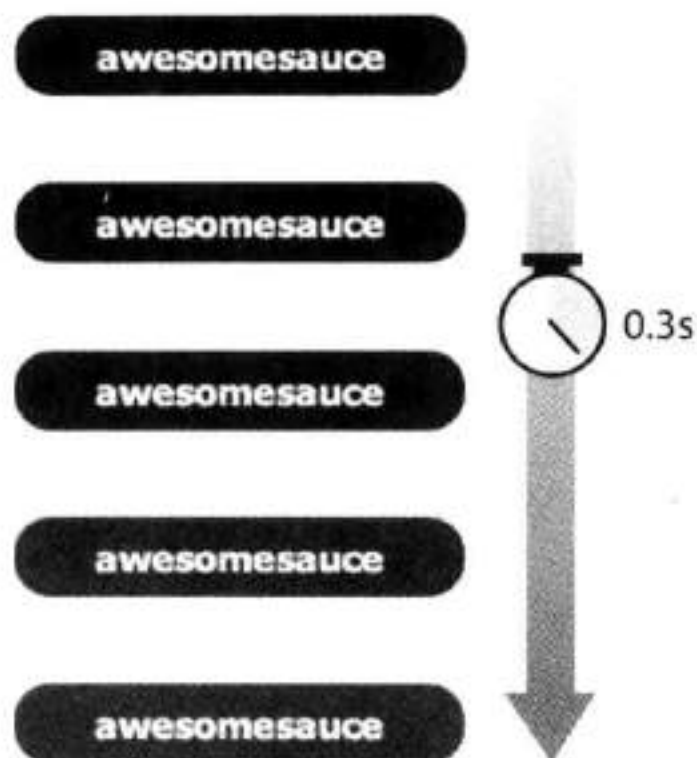


图17-1：当过渡应用时，这个链接的背景色从蓝色渐变到红色

工作草案。写本书时，过渡属性还是很尖端的技术，而且规格可能会发生变化，因此，所有的浏览器都需要使用它们的供应商前缀才能支持。它们在iOS、Android和手机的Opera平台上得到了很好的支持。

唯一不支持过渡的浏览器版本是IE 9及其更早的版本、Firefox 3.6及其更早版本和Opera 10.1及其更早版本。但是，如果你使用过渡来作为增强效果，那么这些浏览器的用户无法看到效果也没关系。对于那些用户来说，直接从蓝色到红色也不是一个大问题。

## 过渡基础

过渡有很多的乐趣，现在进行一个简短的介绍。当运用一个过渡时，需要做一些决定，所有这些都需要使用CSS属性：

- 哪些CSS属性需要更改 (transition-property)
- 它需要持续多久 (transition-duration)
- 以何种方式加速过渡 (transition-timing-function)
- 开始前是否应该有一个暂停 (transition-delay)

你还需要一些东西来触发过渡。状态改变 (如: hover、: focus 或: active) 是很好的触发器，这也是在本章中将使用的例子。你可以使用JavaScript来改变元素 (如添加一个class属性)，并使用它作为一个过渡触发器。

让我们一起完成一个简单的例子。下面是之前想象的从蓝色到红色变化的链接 (图17-1)。没有什么特别的标记。我添加了一个类名，这样我就可以具体指定哪个链接进行过渡。

过渡属性是以普通状态运用到a元素。你会看到a.smooth中的一些声明，如padding和background-color。这使它们可以在文档中被其他状态重复使用。通过为: hover状态声明background-color，我已经改变链接的颜色为红色 (还有: focus，有时候，有些人通过键盘来选中链接)。

### 标记文本

```
awesomesauce
```

### 样式

```
a.smooth {
 display: block;
 text-decoration: none;
 text-align: center;
 padding: 1em 2em;
```

```
width: 10em;
border-radius: 1.5em;
color: #fff;
background-color: mediumblue;
transition-property: background-color;
transition-duration: 0.3s;
}
a.smooth:hover, a.smooth:focus {
 background-color: red;
}
```

指定属性

transition-property

- 属性值: 属性名称|all|none
- 默认值: all
- 适用对象: 所有元素, :before和:after伪元素
- 是否可继承: 否

Transition-property标识要平缓过渡的CSS属性。在这个例子中, 就是background-color。你还可以更改前景颜色、边框、尺寸字体和文本相关的属性或者更多。表17-1中列出了完整列表 (撰写本书时)。更多的属性有可能被添加到该列表中, 浏览器也会实现它们, 所以请检查规约的更新。

需要持续多长时间?

transition-duration

- 属性值: 时间
- 默认值: 0秒
- 适用对象: 所有元素, :before和:after伪元素
- 是否可继承: 否

Transition-duration设置了动画完成需要的时间, 以秒 (s) 或毫秒 (ms) 为单位。我选择了0.3秒, 这就足以使人注意到发生了一些事情, 而且也不会让用户感到单调无味或速度减缓。没有所谓正确的时间, 但是以我的经验, 0.2秒似乎是一个受欢迎的UI元素过渡时间。你可以去实验一下, 看看你的应用程序的该持续多久。

表17-1: CSS动画属性

Backgrounds
background-color
background-position
Borders and outlines
border-bottom-color
border-bottom-width
border-left-color
border-left-width
border-right-color
border-right-width
border-top-color
border-top-width
border-spacing
outline-color
outline-offset
outline-width
Color and opacity
color
opacity
visibility
Font and text
font-size
font-weight
letter-spacing
line-height
text-indent
text-shadow
word-spacing
vertical-align
Element box measurements
height
width
max-height
max-width
min-height
min-width
margin-bottom
margin-left



margin-right
margin-top
padding-bottom
padding-left
padding-right
padding-top
crop
Position
top
right
bottom
left
z-index
clip
Transforms
(not in the spec as of this writing, but supported)
transform
transform-origin

注意：在写本书时，只有Chrome浏览器实现了步骤函数，所以如果你想看到全部效果，可以采用Chrome。

定时功能

transition-timing-function

属性值：	ease linear ease-in ease-out ease-in-out step-start step-end steps cubic-bezier (#,#,#,#)
默认值：	ease
适用对象：	所有元素，:before和:after伪元素
是否可继承：	否

该属性和持续时间形成了过渡的基础，但你可以进一步完善它。随着时间的不同，有许多过渡方式可以推出。例如，它可以快速启动，然后缓慢停下，开始缓慢，然后加快，或一直保持相同速度等。我认为这些都是过渡的“样式”，但在规约中，这称为定时功能。

我可以设置transition-timing-function为ease-in-out，以使链接由蓝色柔和地变成红色。说实话，在很短的持续时间内，几乎察觉不到任何差异。

```
a.smooth {
 ...
 transition-property: background-color;
 transition-duration: 0.3s;
 transition-timing-function: ease-in-out;
}
```

transition-timing-funcion属性采用了下面某个关键值：

- ease  
缓缓启动，然后加速，最后缓慢停下来。这个对于大多数短暂过渡是一个默认值，而且效果很好。
- linear  
从过渡的开始到结束，速度保持一致。
- ease-in  
开始缓慢，然后加快。
- ease-out  
快速启动，然后速度减慢。
- ease-in-out  
缓缓启动，加速，最后再次减慢。这与ease类似，但中间的加速不太明显。
- cubic-bezier(#,#,#,#)  
这是一个函数，用于定义一条Bezier曲线描述的过渡加速。它是比

较高深的数学，我无法在这里全部解释清楚。如果所有的关键字你都觉得不合适，那么可以在规约中阅读详细信息，了解如何实现这个函数 ([www.w3.org/TR/css3-transitions/#transition-timing-function-property](http://www.w3.org/TR/css3-transitions/#transition-timing-function-property))。

#### steps (#,start|end)

通过分步函数把过渡分成一系列的步骤。第一个值是步骤的数量，start和end关键字定义状态变化是否发生在每一步的开始或结尾。有关详细信息，请参阅规约。

#### step-start

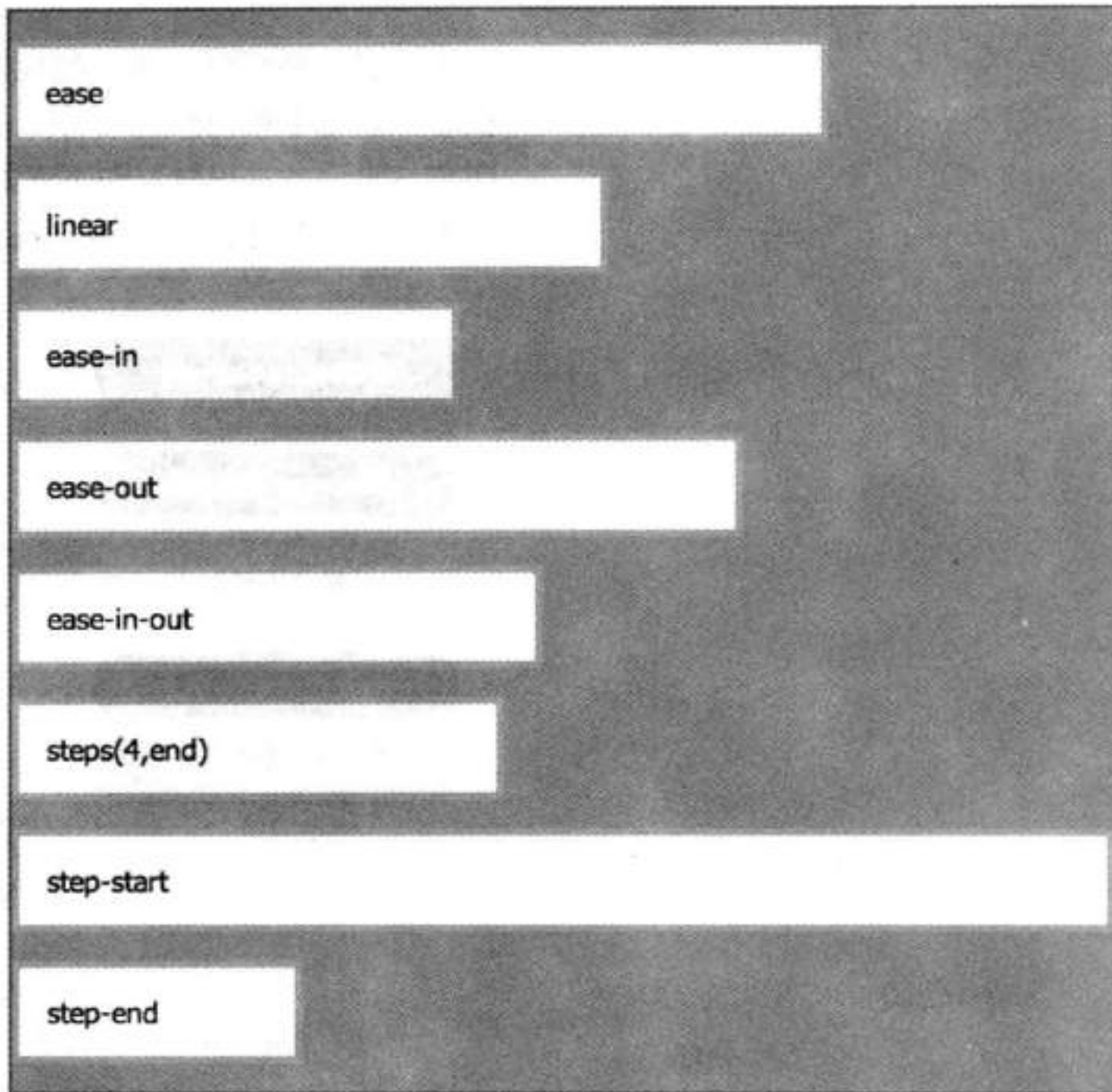
在持续时间刚开始时，改变一个步骤的状态（与steps (1, start) 效果相同）。其结果是一个突然的状态改变，与没有应用过渡的效果相同。

#### step-end

在持续时间结束时，改变一个步骤中的状态（与steps (1, end) 效果相同）。

图17-2：在这个transition-timing-function 演示中，所有元素同时达到最大宽度，但是采取了不同的方式

我不能论证这一页的所有选项，但我已经在网上放了一些演示（如图17-2所示）。当你将鼠标悬停在蓝色的盒子时，每一个标记元素（白色与橙色边框）的宽度会有超过4秒的过渡。它们最后同时达到最大宽度，但是采取的方式不同。





## 设置延时

transition-delay

属性值: 时间

默认值: 0秒

适用对象: 所有元素, :before和:after伪元素

是否可继承: 否

Transition-delay属性, 你可能已经猜到, 它是指启动的动画有指定时间的延迟。在下面的示例中, 背景颜色过渡在指针移到链接上0.2秒之后启动。

```
a.smooth {
 ...
 transition-property: background-color;
 transition-duration: 0.3s;
 transition-timing-function: ease-in-out;
 transition-delay: 0.2s;
}
```

如果需要, 你可以在用户把他的手或指针按在一个按钮上 (:active) 2秒后, 让这个按钮消失 (opacity:0), 即设置transition-delay: 2s; , 如图17-3所示。(积极) 和下面的示例中所示一样。这可以告诉用户: 你在犹豫不决! 当然, transition-delay既有好处, 也有不好的地方。

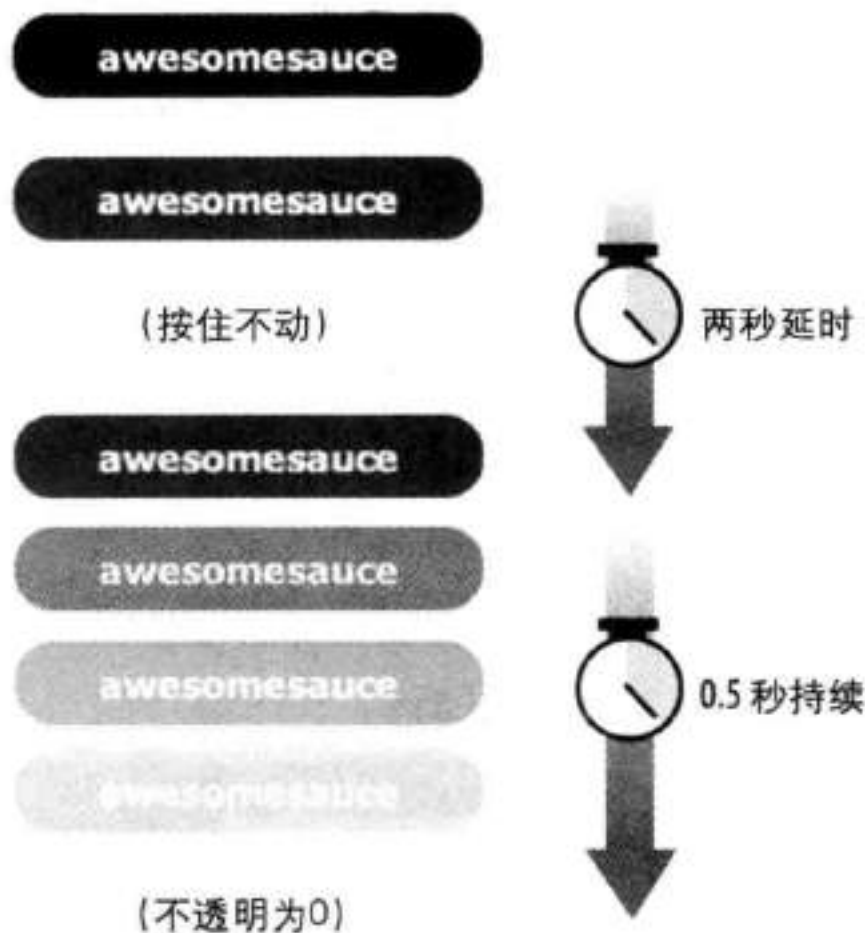


图17-3: transition-delay属性在两秒后, 开始动画效果 (在这里, 动画效果就是使用opacity属性, 使按钮消失)

```

a.smooth {
 ...
 transition-property: opacity;
 transition-duration: .05s;
 transition-timing-function: ease-out;
 transition-delay: 2s;
}
a.smooth:hover, a.smooth:focus {
 background-color: red;
}
a.smooth:active {
 opacity: 0;
}

```

需要指出的是，我一直在整个例子中使用不带前缀的属性，使它们更容易掌握，但要记住，如果你在页面中使用过渡，你必须为所有的浏览器包含供应商前缀。为了前向兼容未来会提供支持的浏览器，也要始终包含不带前缀的语句。下面是实际工作中，完成蓝色到红色过渡的代码：

```

a.smooth {
 ...
 -webkit-transition-property: background-color;
 -webkit-transition-duration: 0.3s;
 -webkit-transition-timing-function: ease-in-out;
 -webkit-transition-delay: 0.2s;

 -moz-transition-property: background-color;
 -moz-transition-duration: 0.3s;
 -moz-transition-timing-function: ease-in-out;
 -moz-transition-delay: 0.2s;

 -o-transition-property: background-color;
 -o-transition-duration: 0.3s;
 -o-transition-timing-function: ease-in-out;
 -o-transition-delay: 0.2s;

 -ms-transition-property: background-color;
 -ms-transition-duration: 0.3s;
 -ms-transition-timing-function: ease-in-out;
 -ms-transition-delay: 0.2s;

 transition-property: background-color;
 transition-duration: 0.3s;
 transition-timing-function: ease-in-out;
 transition-delay: 0.2s;
}

```

这里增加了额外的工作，但是这种方法对可预见的未来是有用的，除非所有的旧浏览器都淘汰，规约也定稿，而且实现也都一致。幸运的是，有快捷方式可以减少一些代码。

## 快捷过渡属性

幸运的是，CSS3规约的作者也意识到了这一点，所以提供了过渡属性的



快捷方式，将所有这些属性结合到一个声明中。你已经看到过类似的边框属性的快捷方式，下面是句法：

```
transition: property duration timing-function delay;
```

每个transition-\*属性的值已经列出，并且被字符空格分离。如果你只提供一个时间值，它会假设是持续时间。如果提供了两个时间值，确保持续期间列在第一位。

使用蓝色到红色链接作为例子，到目前已经运用的4个过渡属性可以合并到这一行：

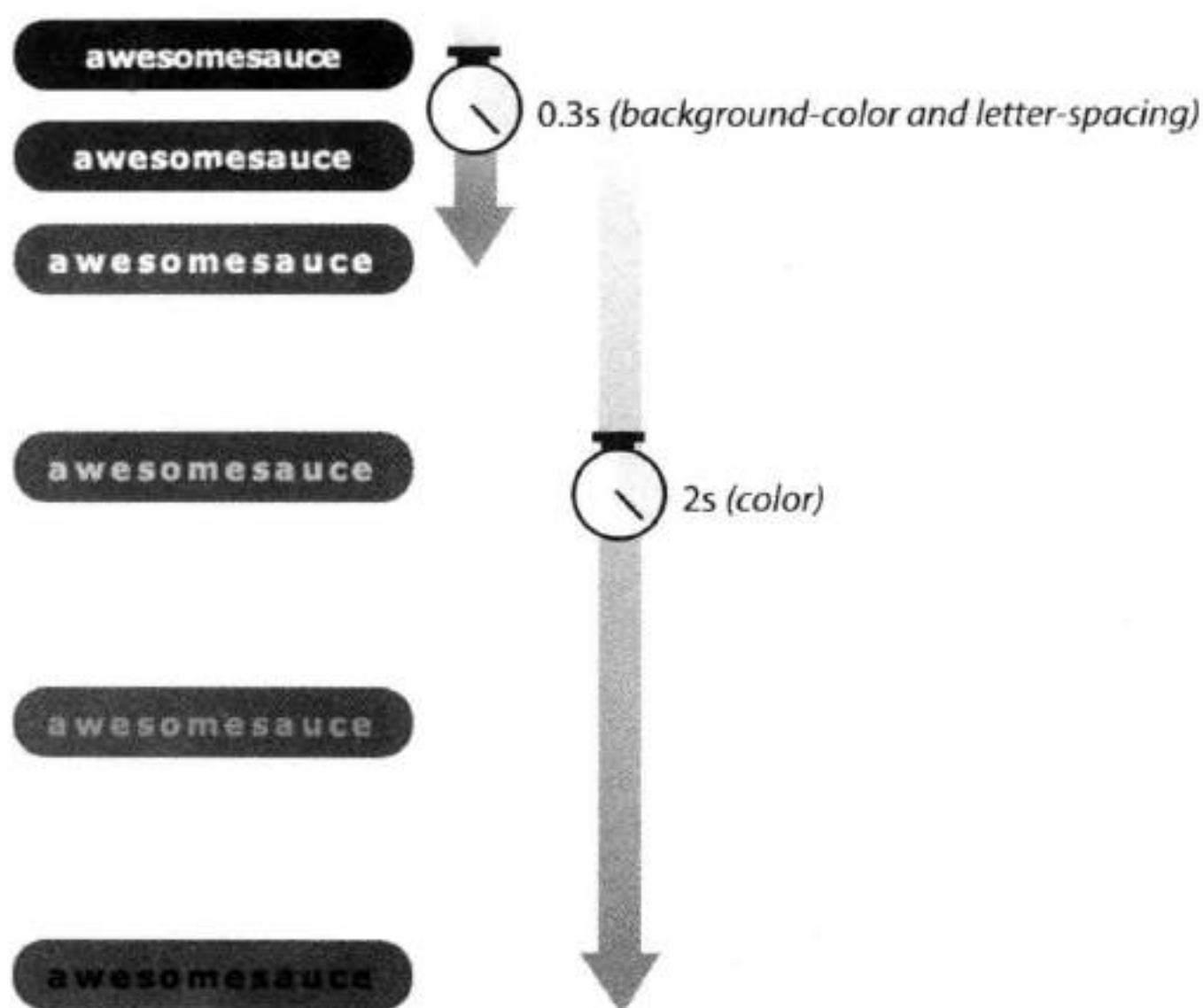
```
a.smooth {
 ...
 transition: background-color 0.3s ease-in-out 0.2s;
}
```

带有所有供应商前缀的版本也可以从20行减为5行：

```
a.smooth {
 ...
 -webkit-transition: background-color 0.3s ease-in-out 0.2s;
 -moz-transition: background-color 0.3s ease-in-out 0.2s;
 -o-transition: background-color 0.3s ease-in-out 0.2s;
 -ms-transition: background-color 0.3s ease-in-out 0.2s;
 transition: background-color 0.3s ease-in-out 0.2s;
}
```

图17-4：颜色，背景色和字符间距以不同的步调变化

这绝对是一个进步！



## 应用多个过渡

到目前，我们每次都只改变一个属性，但是一次设置多个过渡属性也是可以的。让我们回到“awesomesauce”的例子。这一次，除了从蓝色转变为红色，我还想让字母间距增加一点，我也希望文本颜色更改为黑色，但是速度慢于其他动画。图17-4在纸面上显示了这些过渡。

做到这一点的方法之一是：为每个属性列出所有的值，并用逗号分隔，就如本例显示的那样。

```

a.smooth {
 ...
 transition-property: background-color, color, letter-spacing;
 transition-duration: 0.3s, 2s, 0.3s;
 transition-timing-function: ease-out, ease-in, ease-out;
}
a:hover, a:focus {
 background-color: red;
 letter-spacing: 3px;
 color: black;
}

```

这些值根据它们在列表中的位置进行匹配。例如，过渡的color属性（列表中第二个）的持续时间为两秒（2s），并且使用ease-in的定时功能。如果一个列表中的值比其他列表少，浏览器就重复该列表开始的值。在前面的例子中，如果我省略了第三个值：0.3秒的transition-duration，该浏览器将重新回到列表开头，并使用第一个值（0.3）作为letter-spacing的值。在这种情况下，效果将是相同的。

你同样可以将快捷过渡属性的值排成一行。之前看到的样式也可以写为：

```

a.smooth {
 ...
 transition: background-color 0.3s ease-out,
 color 2s ease-in,
 letter-spacing 0.3s ease-out;
}

```

这似乎是一个很好的方式，尤其是当你考虑到你需要将4个供应商前缀版本添加到每个过渡声明的时候。

## 所有场合的过渡

但是，如果你只是想让状态改变更加平滑，而不关心到底改变哪个属性，该怎么做呢？比如，在一些情况下，当你想要将持续、定时功能和延迟同时应用于元素的所有过渡时，可以为transition-property使用all值。在后续的例子中，我会指定a.smooth元素的任何属性变化持续0.2秒，并且使用ease-in-out功能来实现动画。

```

a.smooth {
 ...
 -webkit-transition: all 0.2s ease-in-out;
 -moz-transition: all 0.2s ease-in-out;
 -o-transition: all 0.2s ease-in-out;
 -ms-transition: all 0.2s ease-in-out;
 transition: all 0.2s ease-in-out;
}

```



对于用户界面的变化，一个很短的、微妙的过渡往往就是你所需要的，所以使用all值就可以了。

好了，这就是CSS3过渡的课程。现在，请尝试练习17-1。

### 练习17-1 试试过渡

在本练习中，我们将使用过渡属性实现一个过渡和有效的菜单链接（如图17-5所示）。我在本章的materials文件夹中为你准备了一个初步的文档（*exercise1.html*）。结果代码在附录A中。我建议你使用基于Webkit的最新浏览器（如Chrome和Safari）来查看结果。

1. 首先，看看已经使用的样式。通过使用浮动，列表已经转化为水平菜单。a元素已经像块元素一样显示，下划线也去掉了，并且应用了尺寸和填充、颜色、背景色、边框也已经建立。我使用了box-shadow属性，使链接看起来浮在页面上。
2. 我们会为hover和focus状态定义样式。当用户把鼠标放在链接上（或者用键盘选中链接）时，使背景色变为金色（#fdca00），边框颜色也变为橙色（#fda700）。

```
a:hover, a:focus {
 background-color: #fdca00;
 border-color: #fda700;
}
```

- 3 当用户单击链接（:active），使链接下移3个像素，使其看起来像按下去一样。通过设置a元素的position为relative来实现，然后为active状态修改top属性的值。这将使链接下移到距离顶部边缘3个像素的地方。

```
a {
 ...
 position: relative;
}
a:active {
 top: 3px;
}
```

4. 正常来说，如果按钮被按下去，那么阴影看起来就会小一些，所以也要减少box-shadow的大小。

```
a:active {
 top: 3px;
 box-shadow: 0 1px 2px rgba(0,0,0,.5);
}
```

5. 保存文件，并在浏览器中试试。当你单击链接时，它会变黄并且下移。我感觉这种效果很好。即便没有盒子阴影——IE8和之前版本的用户也确实看不到这种阴影——也没什么影响。现在，可以通过增加平滑过渡来提高用户体验。
6. 在0.2秒内，使背景色和边框色过渡为ease-in，看看菜单的用户体验如何。我使用了快捷transition属性来使代码简洁。我也首先使用了默认的ease定时功能，所以可以从样式中忽略它。

我将在第一个例子中显示所有浏览器前缀，但是如果你使用Chrome或者Safari，你可以只使用-webkit-来节省编写时间。在接下来的例子中，我将只使用标准的无前缀属性来节省空间（其实前缀版本也是类似的）。

```
a {
 -webkit-transition: background-color 0.2s, border-color 0.2s;
 -moz-transition: background-color 0.2s, border-color 0.2s;
 -o-transition: background-color 0.2s, border-color 0.2s;
 -ms-transition: background-color 0.2s, border-color 0.2s;
 transition: background-color 0.2s, border-color 0.2s;
}
```

- 保存文档，在浏览器中打开，然后试着把鼠标移到链接上（见“注意”）。你觉得这样漂亮吗？接下来，我觉得你应该试试其他一些持续时间。看看你是否可以看出0.1秒的持续有什么不同。再试试1秒。我想你会发现持续1秒看起来太慢了。为了看得更清楚，可以设置成几秒，并且试试不同的timing-function值（只需在持续时间后添加即可）。你能看出区别吗？你喜欢哪个？当你完成实验后，将持续时间设置回0.2秒。
- 现在看看当单击链接时，如果给链接的下移动作添加一个过渡会发生什么。对top和box-shadow属性都应用过渡，因为这两个动作是紧跟着的。像前面一样，设置0.2秒的持续。

```
a {
 transition: background-color 0.2s, border-color 0.2s, top .2s,
 box-shadow 0.2s;
}
```

保存文件，在浏览器中打开，然后试着单击链接。过渡确实改变了菜单的用户体验，难道不是吗？按钮看起来很难按下去。试着增加持续时间。是否感觉更加难按下去了？我发现时间对用户体验的影响很有趣。选择合适的时间很重要，否则页面就会感觉呆滞缓慢。我可以告诉你个经验，一个很短的过渡，（比如0.1秒，甚至不用过渡）会让这些按钮用起来感觉更利落。

- 如果你认为增加持续时间后，菜单用起来很不舒服，可以试着给top和box-shadow属性添加0.5秒的延时。

```
a {
 transition: background-color
 0.2s, border-color 0.2s, top 0.2s
 0.5s, box-shadow 0.2s 0.5s;
}
```

我想你会发现这一点点的额外时间，都会破坏整个页面的感觉。时间就是一切啊！

**注意：**如果你使用触摸屏体验这个练习，你会错过这种效果，因为在触摸屏上没有: hover状态。

图17-5：在这个练习中，我们会在这些状态中创建过渡

普通状态



:hover, :focus  
背景和边框色改变



:active  
链接看起来像被按下去了





## CSS变换

transform

属性值: 变换函数 (s) | none

默认值: none

适用对象: 变换元素 (见侧栏)

是否可继承: 否

CSS3变换模块为作者提供了旋转方式、重新定位、调整大小以及在二维和三维空间中倾斜HTML元素。然而本章专注于更简单的2-D类型，因为他们有更实际的用途。变换使用供应商前缀，可以被主流浏览器所支持。IE 8及以前的版本并不支持，Firefox 3及更早版本和Opera 10.1及更早版本也不支持。

你可以对正常状态的元素应用变换，它将会在加载页面时出现变换状态。只是要确保页面在不支持的浏览器上仍然是可用的。只有当用户通过翻转或JavaScript事件与元素交互时，才需要变换（在CSS专家Dan Cederholm的CSS for Web Designers书中“体验层”里有讲述）。无论哪种方式，变换都是一个很好的备选方案，即便一个IE 8的用户看到的是一个笔直的元素，而不是有一定变换效果的元素，也没什么影响。

图17-6列出了四种类型的二维变换：旋转、平移、缩放和倾斜。虚线轮廓表示元素的初始位置。

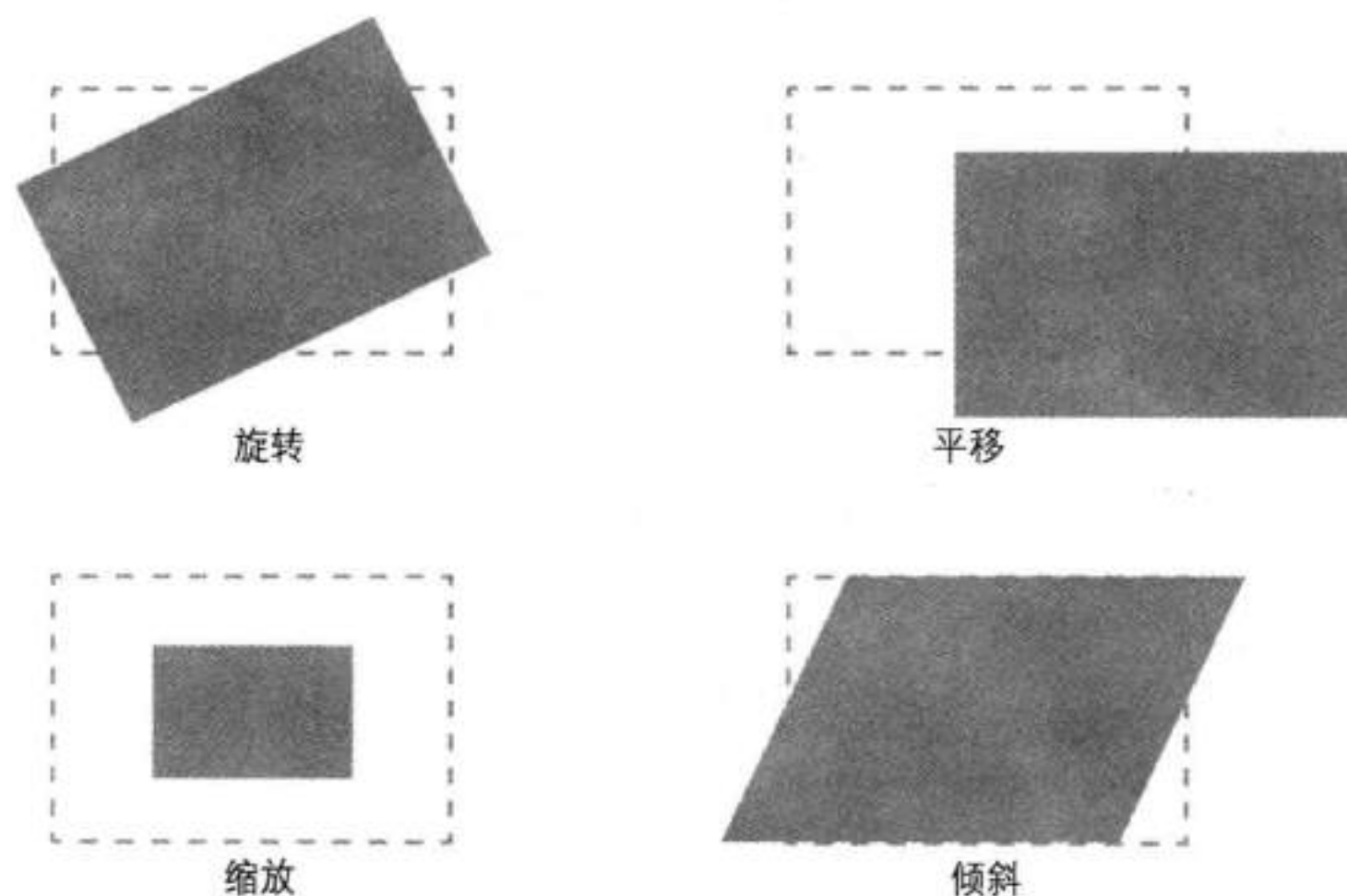


图17-6: 四种类型的变换：旋转、平移、缩放和倾斜

**注意：** 2D变换、3D变换以及SVG变换模型都包含在2012年的一个CSS变换草案文档中。可以从[www.w3.org/TR/css3-transforms/](http://www.w3.org/TR/css3-transforms/)获取规约。

当一个元素变换时，它的元素框保持其原来的位置，并且影响周围的布局，通过相对定位元素，空白也以同样的方式留在后面。这是因为虽然变换神奇的拾取渲染元素的像素，并在页面上变换这些像素。所以，如果你移动一个变换的元素，你只能移动它的的图片。该图片对周围的布局没有影响。

让我们一个一个地学习变换，从旋转开始。

## 变换角度（旋转）

如果你想一个元素有一点角度，可以使用旋转变换功能。`rotate`变换函数的值是一个角度，以正的或负的度数来指定。图17-7中的图像已被旋转-10度（350度），它使用的是下面的样式规则。微白的阴影图像显示元素的初始位置，以供参考。

```
img {
 width: 300px;
 height: 400px;
 transform: rotate(-10deg);
}
```



图17-7：使用`transform:rotate()`旋转一个元素

注意，图像围绕其中心点旋转，这是所有变换的默认点，但你可以很容易使用`transform-origin`属性来改变它。

### transform-origin

属性值： 百分比|长度|left|center|right|top|bottom  
 默认值： 50% 50%  
 适用对象： 可变换元素  
 是否可继承： 否

**注意：**事实上，在CSS规约中有5种2D变换函数。第五个是`matrix`，可以让你使用6个值和一些可怕的三角函数来实现自己的变换。理论上很美妙，但就我而言，我不喜欢这么做。如果你有兴趣，并且还记得三角函数，可以看看这个`matrix`变换的资料：  
[www.w3.org/TR/SVG/coords.html#InterfaceSVGMatrix](http://www.w3.org/TR/SVG/coords.html#InterfaceSVGMatrix).

### 可变换元素

你可以给下面的元素应用`transform`属性。

- 有替代内容的HTML元素，比如、`canvas`、表单输入以及嵌入式媒体。
- 有`display:block`的元素
- 有`display:inlineblock`的元素
- 有`display:inlinetable`的元素（或者任何一种`table-*`显示类型）



`Transform-origin`的值有两个关键字或者一个，可以是长度测量值，或百分比值。第一个值是水平偏移量，第二个是垂直方向的偏移量。如果只提供一个值，它会用于两者。如果我们想使红木森林图像以它顶部边缘的点为中心旋转，可以编写如下代码：

```
transform-origin: center top;
transform-origin: 50%, 0%;
transform-origin: 150px, 0;
```

图17-8中的图像都旋转25度，但是都是以不同的原点为中心。

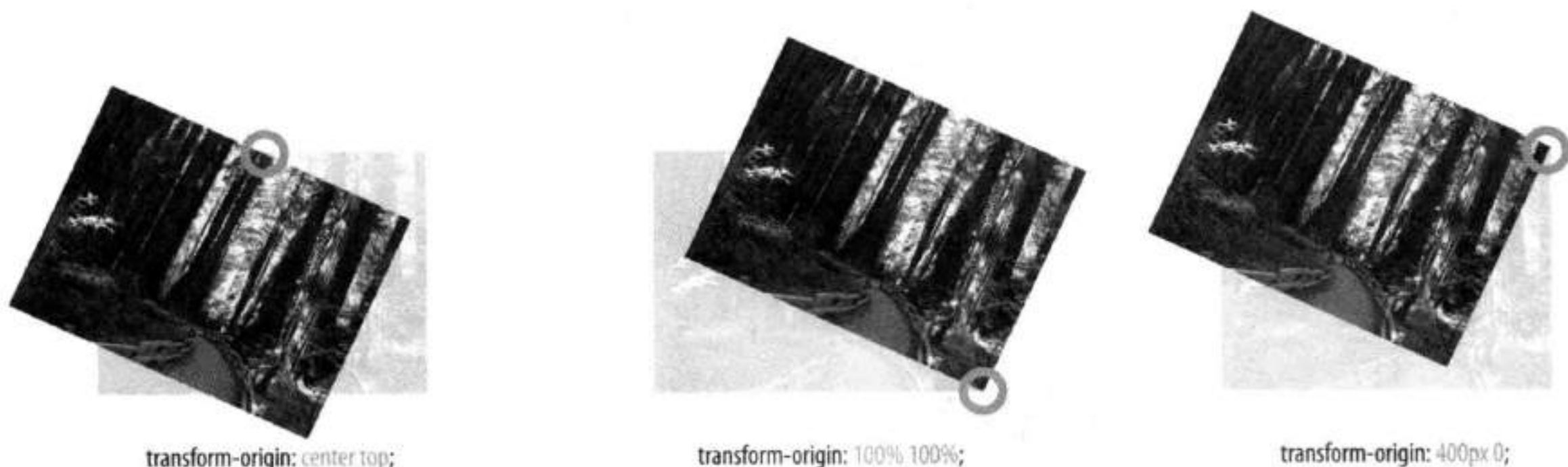


图17-8：使用`transform-origin`改变旋转所围绕的中心

使用旋转函数，可以很容易地证明所围绕的原点，但是要知道，你可以为任何变换功能设置一个原点

### 变换位置（平移）

另一件你可以用`transform`属性实现的是：使用三个`translate`函数之一，在页面上给元素渲染一个新的位置，就像图17-9所示的例子一样。`translateX`函数允许你在水平轴上移动元素；`translateY`是沿着垂直轴移动；`translate`结合了 X和Y两个的值（`translate(translateX, translateY)`）。

```
transform: translateX(50px);
transform: translateY(25px);
transform: translate(50px, 25px);
```



图17-9：使用translate函数来移动一个元素

你可以提供一个长度值，可以是任何CSS单位或百分比值。百分比以边界框的宽度来计算，边界框宽度就是指从边框的一边缘到另一边缘的宽度（顺便说一句，这也是在SVG中计算百分比的方式）。如图17-9所示，你可以提供正值或负值。

如果你为translate函数提供一个值，它会假定为translateX值，而translateY将设置为零。因此，translate(20px)相当于同时应用translateX(20px)和translateY(0)。

到目前，你感觉transform属性怎么样？接下来还有两种函数需要学习。

## 变换的大小（缩放）

使用下面三个函数中的任何一个，都可以使元素显示得更大或更小，这三个缩放函数是：将scaleX（水平）、scaleY（垂直）和快捷scale。该值是无单位的数字，表示尺寸比例。下面的例子使影像是其原始宽度的150%：

```
a img {
 transform: scaleX(1.5);
}
```

快捷scale属性列出了scaleX和scaleY的值。这个例子中，元素的宽是原来的两倍，但高只有原来的一半。

```
a img {
 transform: scale(2, .5);
}
```

然而，与translate不同，如果你只提供一个值，它会将其视为两个方向上的缩放比例。因此，scale(2)与scaleX(2)和scaleY(2)一样。



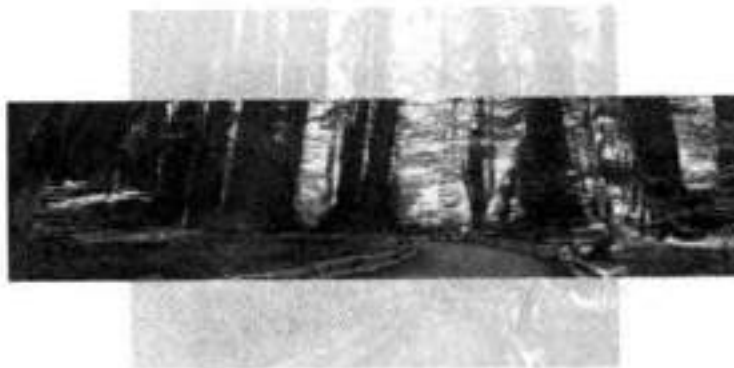
图17-10显示了缩放的结果。



`transform: scale(1.25);`



`transform: scale(.75);`



`transform: scale(1.5, .5);`

图17-10：通过scale函数改变元素的大小

### 使它倾斜 (skew)

这个古怪的skew属性 (skewX、skewY和快捷skew) 通过指定度数来改变水平或垂直方向的角度 (或两个方向角度都改变)。与translate类似，如果你只提供一个值，它会用于skewX，而skewY将设置为零。

得知skew如何工作的最好方式是看一些例子 (图17-11)。

```
a img {
 transform: skewX(15deg);
}

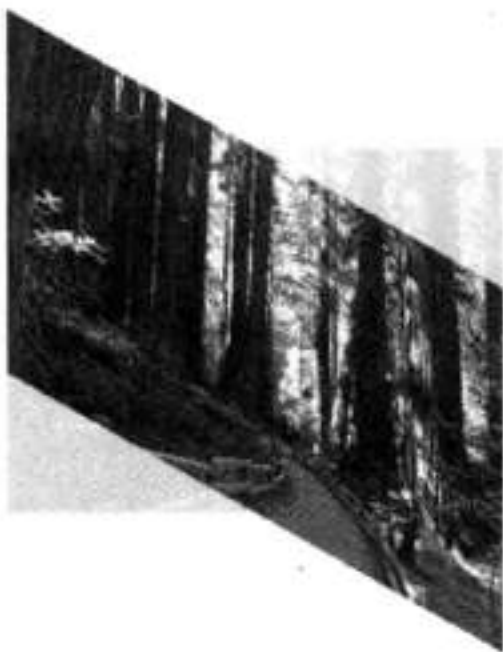
a img {
 transform: skewY(30deg);
}

a img {
 transform: skew(15deg, 30deg);
}
```

图17-11：使用skew函数来倾斜一个元素



`transform: skewX(15deg);`



`transform: skewY(30deg);`



`transform: skew(15deg, 30deg);`

## 应用多个变换

当然，可以对一个元素应用一个以上的变换。只要列出的函数和它们的值，用空格隔开即可，例如：

```
transform: function(value) function(value);
```

在图17-12的例子中，当鼠标移到图片上或者当图片是焦点时，森林图像会放大、倾斜，并向右下方移动。

```
img:hover, img:focus {
 transform: scale(1.5) rotate(-5deg) translate(50px,30px);
}
```

Normal state



:hover, :focus  
(rotate, translate, and scale applied)



图17-12：对一个元素应用缩放、旋转和平移

值得注意的是，变换执行的顺序和它们所列的顺序相同。例如，如果你应用一个平移，然后应用旋转，与先旋转再平移的效果并不相同。

另外一点要注意的是，如果你想给不同的状态（如hover、focus和active）应用额外的变换，你需要重复元素已经应用的所有的变换。例如，a元素在其正常状态下旋转45度。如果给hover状态应用scale变换，我就会失去旋转效果，除非我明确地再次声明旋转。

```
a {
 transform: rotate(45deg);
}
a:hover {
 transform: scale(1.25); /* rotate on a element would be lost */
}
```

为了同时达到旋转和缩放的效果，同时提供变换值：

```
a:hover {
 transform: rotate(45deg) scale(1.25); /* rotates and scales */
}
```



### 不要忘了前缀

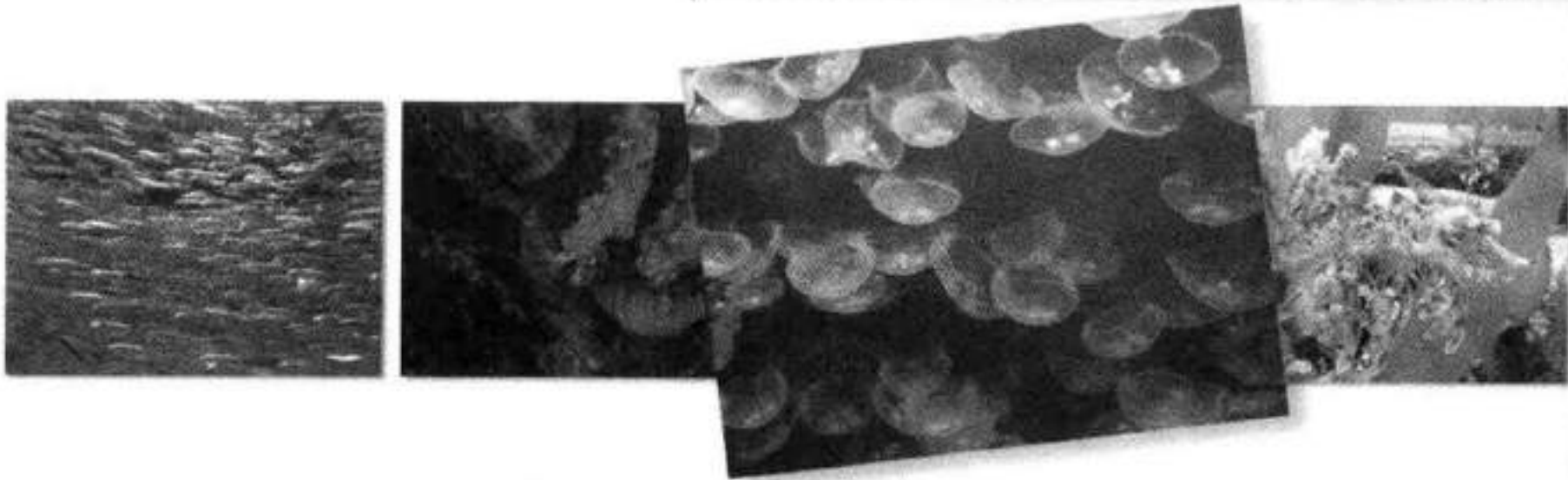
为了看起来清晰，我只使用了标准语法来完成transform的例子。但是在实际中，transform属性在所有支持它的浏览器中都需要供应商前缀。下面是真实发布的站点中，应用多个变换的例子：

```
a:hover img, a:focus img{
 -webkit-transform: scale(1.5) rotate(-5deg) translate(50px,30px);
 -moz-transform: scale(1.5) rotate(-5deg) translate(50px,30px);
 -o-transform: scale(1.5) rotate(-5deg) translate(50px,30px);
 -ms-transform: scale(1.5) rotate(-5deg) translate(50px,30px);
 transform: scale(1.5) rotate(-5deg) translate(50px,30px);
}
```

### 非常平滑的变换

多个变换应用于红木森林图像看起来很有意思，但是如果使用流畅的动画，它的效果可能会更好！现在，你知道的过渡和变换，让我们把他们结合在一起，来见证奇迹的发生。这里的“奇迹”，意思是两个状态之间的一些基本动画效果。在练习17-2中，我们来一步一步实现它。

图17-13：照片在:hover和:focus状态时变大并且倾斜。可以使用过渡来让它更流畅。当你完成本练习，你可以看到它的效果，或者你也可以直接到 [earningwebdesign.com/4e/chapter17/figures.html](http://earningwebdesign.com/4e/chapter17/figures.html) 查看



### 练习17-2 过渡变换

在本练习中，当用户的鼠标放在17-13所示的图片上时，我会使这些照片变大，并且倾斜。我会使用过渡来更平滑地实现这个效果。你可以在本章的 *materials* 文件夹下找到初始文档 (*acquarium.html*) 和所有图像。

1. 在文本编辑器中打开 *acquarium.html*，你会看到其中已经有使列表项水平排列的样式，并且已经给每个图片应用了阴影。（注意：如果你看不到阴影，说明你使用的不是最新的浏览器。）我们要做的第一件事是给每个图片添加transform属性。

2. 我们只希望鼠标移动到图片上或者图片获取焦点时，才发生变换效果，所以transform

属性应该应用于:hover和:focus状态。因为我希望每个图像的倾斜都不相同，所以需要为每个图像都写一个倾斜规则，可以使用图像的唯一id作为选择器来实现。你可以保存文档，并在浏览器中查看结果。

```

a:hover #img1, a:focus #img1 {
 transform: rotate(-3deg);
}
a:hover #img2, a:focus #img2 {
 transform: rotate(5deg);
}
a:hover #img3, a:focus #img2 {
 transform: rotate(-7deg);
}
a:hover #img4, a:focus #img2 {
 transform: rotate(2deg);
}

```

3. 接下来实现放大功能，使用户能看得更清晰。给每个transform值添加scale(1.5)。下面是第一个，你来实现其他的。

```

a:hover #img1 {
 transform: rotate(-3deg) scale(1.5);
}

```

值得注意的是，我的图像文件都很大，然后我把它们缩小到了缩略图。如果开始使用小图片，然后再放大，图片会很模糊很难看。

4. 当想给照片一种在屏幕上抬起的效果时，可以使下阴影看起来远一些，只要增加阴影的偏移量，并且模糊淡化阴影灰度即可。所有的图像看起来应该有相同的效果，所以使用a:hover img作为选择器，添加一条规则。

```

a:hover img {
 box-shadow: 6px 6px 6px rgba(0,0,0,.3);
}

```

保存文件，并在浏览器中查看。当鼠标放在图像上时，图像会稍微倾斜并且放大。但是这种效果似乎不太和谐。现在添加一个过渡来改善一下。

5. 给img普通状态添加transition快捷属性（也就是非:hover和:focus状态）。在这里，想添加过渡的属性是transform。把持续时间设置为0.3秒，并且使用linear定时功能。

```

img {
 ...
 transition: transform 0.3s linear;
}

```

注意供应商前缀，transform属性也需要这些前缀。比如，Webkit版本应该是：

```
-webkit-transition: -webkit-transform .3s linear;
```

到此就完成了。你可以试试不同的持续时间和定时功能，也可以试着修改变换，或者旋转的原点，看看效果怎么样。

注意：注意，我省去了供应商前缀，但是如果你使用Chrome和Safari浏览器，你需要使用-webkit-前缀。

### 3-D变换

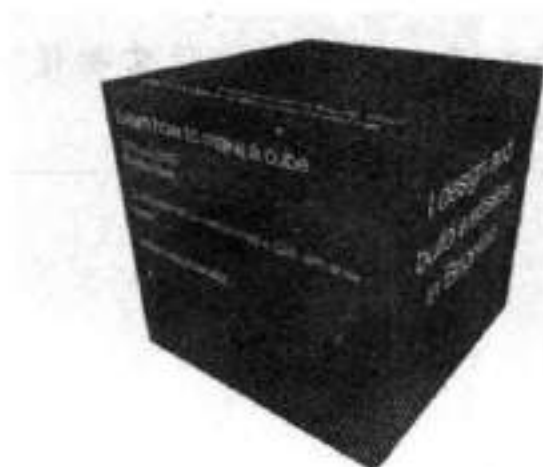
除了两维的变换函数，CSS的变换规范还描述了一种可以营造空间和角度的系统。结合与过渡，你可以使用3-D变换创造丰富的交互接口，如图



像旋转木马、翻动卡或旋转立方体！（Web上的CSS立方体现在不存在问题，它是一个很好的学习项目。）图17-14显示了一些使用3-D变换创建接口的示例。在过去，如果你看到像这样的3-D接口，你肯定会认为它是Flash。现在，浏览器原生的功能和老版的CSS3就可以做到。

3-D变换并不是网页设计初学者需要了解的技能，所以我不打算在这里探讨全部细节，但我会给你一个初步尝试，了解需要如何做才能添加三维效果。如果你想了解更多，下面的教程是开始的好地方：

图17-14：3-D变换的一些例子



**Paul Hayes' 3D cube**  
([www.paulhayes.com/experiments/cube-3d/touch.html](http://www.paulhayes.com/experiments/cube-3d/touch.html))



**Safari Technology Demos: Web Gallery**  
([developer.apple.com/safaridemos/showcase/gallery/](http://developer.apple.com/safaridemos/showcase/gallery/))

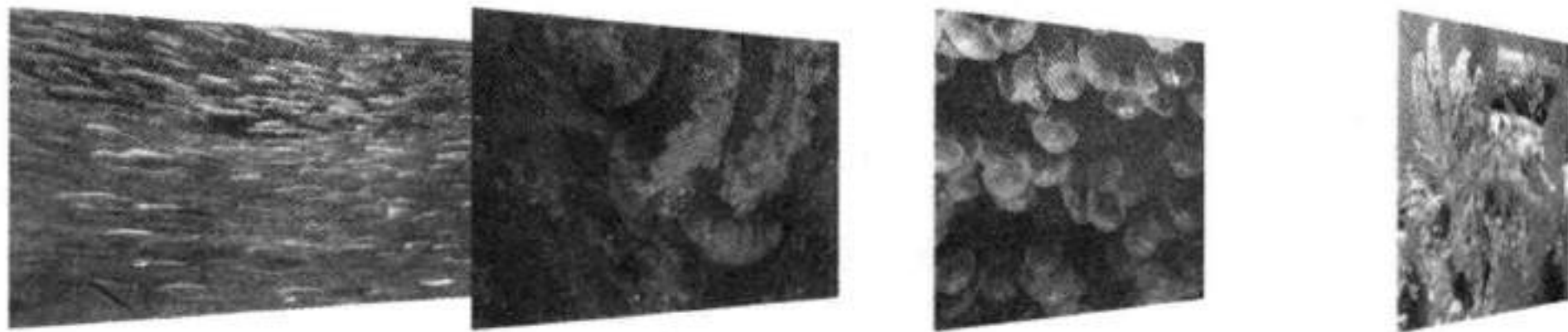


**Snow Stack by Charles Ying**  
([www.satine.org/research/webkit/snowleopard/snowstack.html](http://www.satine.org/research/webkit/snowleopard/snowstack.html))

- “Adventures In The Third Dimension: CSS 3D Transforms”，作者 Peter Gasston ([coding.smashingmagazine.com/2012/01/06/adventures-in-the-third-dimension-CSS-3-d-transforms/](http://coding.smashingmagazine.com/2012/01/06/adventures-in-the-third-dimension-CSS-3-d-transforms/))
- “Intro to 3D Transforms”，作者 David DeSandro ([desandro.github.com/3dtransforms/](http://desandro.github.com/3dtransforms/))

为了给你一个非常简单的例子，我打算使用练习17-2中的图像，并把它们放在一个3-D旋转木马式的画廊中（如图17-15所示）。

图17-15：放在立体空间中的水族馆图像



下面的标记文本是与练习17-2相同的无序列表。

```



```

第一个步骤是使用`perspective`属性添加一定量的“立体”的元素。这告诉浏览器子元素应该表现得好像他们是在3-D空间中似的。`perspective`属性是某个整数大于零值，表示的距离元素在z轴的原点有多远。该值越小，角度越大。我已发现，在300~1500之间的值是合理的，但也不必这么拘束，你可以大胆尝试，直到你得到所需的效果。

```
ul {
 width: 1000px;
 height: 100px;
 list-style-type: none;
 padding: 0;
 margin: 0;
 -webkit-perspective: 600;
 -moz-perspective: 600;
 perspective: 600;
}
```

`perspective-origin`属性（上面代码中没有显示）描述你的眼睛相对于变换项的位置。该值是一个水平位置（`left`、`center`、`right`，长度或百分比值）和垂直位置（`top`、`bottom`、`center`，长度或百分比值）。默认值（如图17-15所示）是水平和垂直居中的（`perspective-origin: 50% 50%`）。最后与变换相关的属性是`backface-visibility`，它控制元素旋转时，元素的侧翻面是否可见。

3-D空间的建立，给每个`ul`中的`li`元素应用一个3-D变换函数。3-D函数包括`translate3d`、`translateZ`、`scale3d`、`scaleZ`、`rotate3d`、`rotateX`、`rotateY`、`rotateZ`和`matrix3d`。你应该知道其中一些函数的作用。`*Z`函数定义对象相对于z轴的方向（z轴就是从你的鼻子到页面的这条线，而x-和y-轴就在页面上）。

在这个例子中，如图17-15所示，每个`li`使用的`rotateY`函数围绕其y轴（垂直轴）旋转45度。

而图17-16中，每个`li`是使用`rotateX`函数围绕其x轴（水平轴）的上旋转。

```
li {
 float: left;
 margin-right: 10px;
 -webkit-transform: rotateX(45deg);
 -moz-transform: rotateX(45deg);
 transform: rotateX(45deg);
}
```



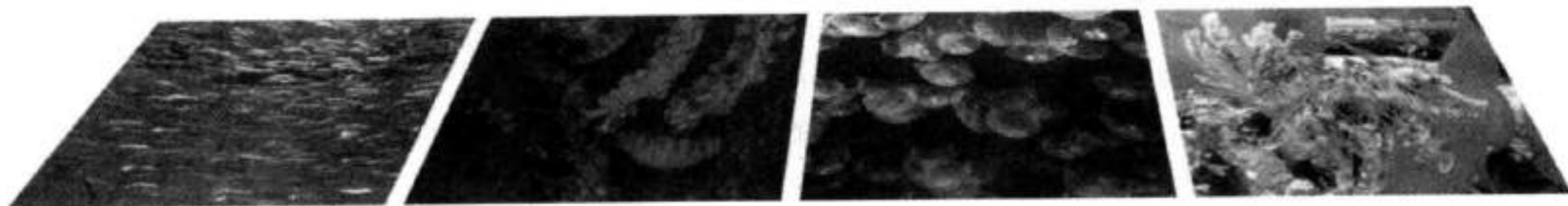


图17-16：相同的一组图片，使用rotateX函数以水平轴旋转

很显然，我几乎没有深入探讨3-D变换，但你应该已经有一个大体的了解。接下来，我将向你介绍一个更复杂的方式，它可以使你的网页动起来。

## 关键帧动画

CSS动画模块允许作者创建真实的关键帧动画。从一个状态到另一个状态的变换不同，关键帧动画允许你在过程中的某些时点明确指定其他的状态，并且允许更精细的控制动作。

过程中的某些“时点”使用keyframes来建立，keyframes定义了动画的开头或结尾。CSS过渡就是有两个关键帧的动画：开始状态和结束状态。更复杂的动画需要许多关键帧来控制序列中的属性变化。

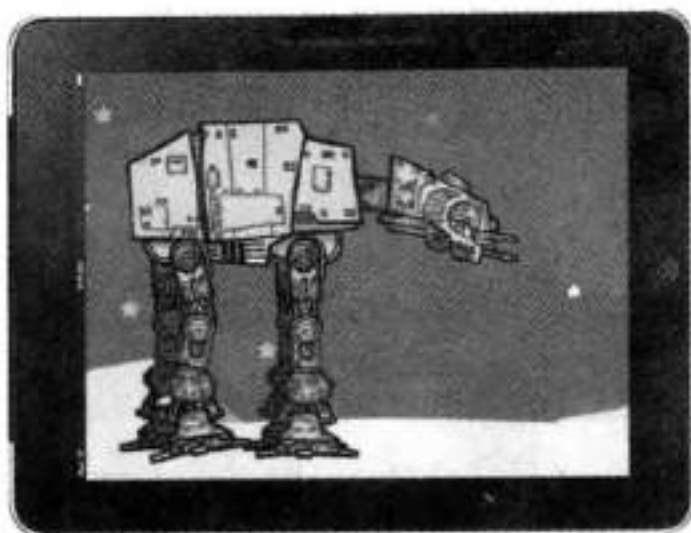
创建关键帧动画是复杂的，远远超过了我这里所讲的内容。不过，我希望你可以大体了解它是如何工作的，所以我会讲最核心的细节。下面的资源都是很好的材料，有助于你了解更多信息：

- “A Masterclass in CSS Animations”，作者Estelle Weyl ([www.netmagazine.com/tutorials/masterclass-css-animations](http://www.netmagazine.com/tutorials/masterclass-css-animations))
- “The Guide to CSS Animation: Principles and Examples” ([coding.smashingmagazine.com/2011/09/14/the-guide-to-css-animation-principles-and-examples/](http://coding.smashingmagazine.com/2011/09/14/the-guide-to-css-animation-principles-and-examples/))
- Rich Bradshaw的教程“Using CSS3 Transitions, Transforms and Animation” ([css3.bradshawenterprises.com](http://css3.bradshawenterprises.com))
- [Anthonycalzadilla.com](http://Anthonycalzadilla.com)。我的朋友Anthony Calzadilla在CSS动画方面做了开创性的工作，包括The Walking At-At和CSS3-Man动画（如图17-17所示），这都是很超前的。他的网站一些链接，指向动画实例和CSS一般新闻。他还建立了一个CSS场景动画教程。

---

**注意：**关键帧动画就是显式动画（explicit animation），因为你可以控制它的行为。相反，过渡是隐式动画implicit animation，因为只有当属性改变时，他们才变化。

---



纯粹的CSS3 AT-AT Walker  
Anthony Calzadilla 提供  
([www.anthonycalzadilla.com/css3-ATAT/](http://www.anthonycalzadilla.com/css3-ATAT/))



CSS3-Man  
Anthony Calzadilla 提供  
([www.optimum7.com/css3-man/](http://www.optimum7.com/css3-man/))



如何学习走路  
Andrew Hoyer 提供  
([andrew-hoyer.com/blog/2010/10/21/walking/](http://andrew-hoyer.com/blog/2010/10/21/walking/))



CSS3制作的星球大战介绍  
Guillermo Esteves 提供  
([www.gesteves.com/experiments/starwars.html](http://www.gesteves.com/experiments/starwars.html))

图17-17：使用CSS的动画例子

## 建立关键帧

动画过程有两个部分：第一，建立关键帧，使用@keyframes规则，然后给需要动画效果的元素添加动画属性。

这里是一个非常简单的关键帧，随着时间推移，改变一个元素的背景颜色。这并不是一个动感十足的动画，但它应该可以让你基本了解什么是@keyframes规则。

```
@keyframes colors {
 0% { background-color: red; }
 20% { background-color: orange; }
 40% { background-color: yellow; }
 60% { background-color: green; }
 80% { background-color: blue; }
 100% { background-color: purple; }
}
```

@keyframes的规则是说：创建一个名叫“colors”的动画序列。在动画开头，background-color元素应该是红色的，通过动画运行20%后背景颜色是橙色的，依次类推，直到动画结束。浏览器填补了所有关键帧中的幻影颜色（或补间动画tweens一术语）。如图17-18所示。

### 动画工具

如果你想要使用CSS动画，但是没有精力来学习，这里有一些工具可以给你提供创建动画的时间线界面，并且会生成HTML和CSS代码。下面是写本书时，已有的工具：

- Tumult Hype, [tumultco.com/hype/](http://tumultco.com/hype/) (Mac only)
- Sencha Animator, [www.sencha.com/products/animator/](http://www.sencha.com/products/animator/)
- Adobe Edge, [labs.adobe.com/technologies/edge/](http://labs.adobe.com/technologies/edge/)

注意：@keyframes规则需要供应商前缀，如下所示：

```
@-webkit-keyframes
```



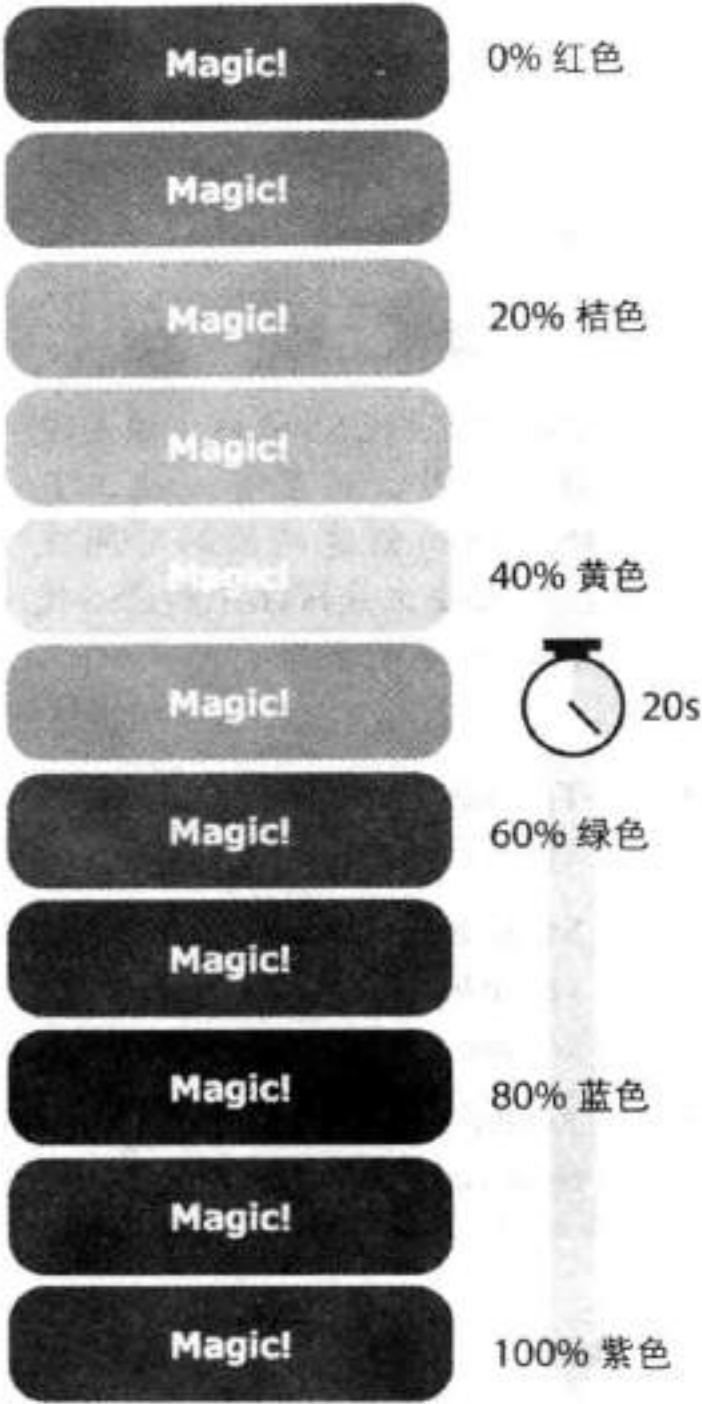


图17-18：使用关键帧的彩虹色彩变换动画

每一个百分比值和属性/值声明，定义了动画序列中的一个关键帧。除了百分比，你也可以在动画序列开始时使用from关键字，在结束时使用to。这里有一个@keyframes规则的一般语法。

```
@keyframes animation-name {
 keyframe { property: value; }
 keyframe { property: value; }
}
```

### 添加动画属性

现在，可以使用动画属性集，对文档中的一个或多个元素应用动画序列，这与你已经知道的过渡属性用法非常相似。

我要给文档中的#magic div应用彩虹动画。

```
<div id="magic">Magic!</div>
```

在CSS规则的#magic处，我需要决定应用哪些动画：

- 使用哪个动画 (animation-name)
- 应采取多长时间 (animation-duration)
- 它应该如何加速 (animation-timing-function)
- 是否在开始之前暂停 (animation-delay)

看起来很熟悉？还有一些动画特定的属性。

- 多少次重复 (animation-iteration-count)
- 是否向前播放，或者后退，或者前后来回播放 (animation-direction)
- 是否应该运行或暂停。可以使用JavaScript切换播放状态为开启和关闭，或者用：hover状态来切换 (animation-play-state)
- 是否覆盖默认值，以防止属性应用外部的运行时 (animation-fill-mode)

animation-name属性告诉浏览器对#magic div应用哪个关键帧序列。我还设置了持续和计时功能，使用animation-iteration-count来使动画无限地重复。我也可以提供一个具体的数值，如设定为2，表示播放两次，两个彩虹好玩吗？为了好玩，我就设置animation-direction为alternate，这使得动画正向播放后，就反向播放。其他的选择是简单的forward或者reverse。下面是动画div的规则。

```
#magic {
 ...
```

```

 animation-name: colors;
 animation-duration: 5s;
 animation-timing-function: linear;
 animation-iteration-count: infinite;
 animation-direction: alternate;
}

```

这样做有点冗长，尤其是还要添加供应商前缀。你也可以用快捷动画属性来把值结合在一起，就像对过渡所做的。

```

#magic {
 animation: colors 5s linear infinite alternate;
}

```

这些都是创建关键帧并对一个页面上的元素应用动画的大概方法。为了使元素左右移动（通常认为这就是“动画”），可以使用关键帧和 `translate` 变换（或者 `top`、`bottom` 和 `left` 属性）来改变元素在屏幕上的位置。当关键帧补间时，该对象将平滑地从一个位置移动到另一个位置。你也可以用其他的变换方法。

我希望我已经帮你大概了解如何使用CSS来给你的页面添加一点运动和平滑效果。这是很酷的东西，但要记住有节制地使用它，并仅将它作为一个增强效果。如果你正在展示你的动画，最要让你的访问者更新他们的浏览器。

现在，看看下面的小测验！

## 自我测验

我想你应该掌握过渡、变换和关键帧动画了吧？这里有几个问题，试试看。

1. 什么是“补间动画”？
2. 如果一个过渡有关键帧，它会有多少？
3. 写出过渡的声明（属性和值），来完成以下任务：
  - a. 过渡开始之前等待0.5秒
  - b. 使过渡以恒定的速度发生
  - c. 使过渡持续0.5秒

**注意：**在这个规约中，`@keyframe`和`animation`属性中`animation-name`的值需要包含在单引号中。便捷的写法是：

```

animation: 'colors' 5s linear
 infinite reverse;

```

然而开发者们现在都忽略了引号以避免在Firefox中出现问題。



- d. 使文本的行慢慢地隔得越来越远
- 4. 下列哪项你不能设置动画?
  - a. Width
  - b. padding
  - c. text-transform
  - d. word-spacing
- 5. 如果你省略了transition-timing-function属性, 你会使用哪个定时函数? 详细说明它的作用。
- 6. 在下面的过渡中, 0.2s表示什么?  
`transition:color .2s linear`
- 7. 哪个过渡会先完成?
  - a. `transition: width 300ms ease-in;`
  - b. `transition: width 300ms ease-out;`
- 8. 编写变换声明, 完成以下任务:
  - a. 七度倾斜元素
  - b. 元素上方复位25像素, 左边50像素
  - c. 从右下角旋转元素
  - d. 一个将400像素宽的图像以500像素宽来显示
- 9. 在下面的变换声明中, 值3表示什么?  
`transform:scale(2,3)`
- 10. 哪个3-D变换更有角度和戏剧性呢?
  - a. `perspective: 250;`
  - b. `perspective: 1250;`

11. 这个动画中途会发生什么事？

```
@keyframes border-bulge {
 from { border-width: 1px; }
 25% { border-width: 10px; }
 50% { border-width: 3px; }
 to { border-width: 5px; }
}
```

12. 编写动画的声明，完成以下任务：

- a. 反向播放动画
- b. 使整个动画持续5秒钟
- c. 等待2秒钟，然后运行动画
- d. 动画重复3次，然后停止

## CSS回顾：过渡、变换和动画

下面是本章讲述的属性的总结，以字母顺序排列。

属性	描述
animation	结合动画属性的快捷属性
animation-name	为动画序列指定一个名字
animation-duration	动画持续的时间
animation-timing-function	描述动画的加速
animation-iteration-count	动画重复的次数
animation-direction	动画是否正向播放、反向播放，或者前后交替播放
animation-play-state	动画播放还是暂停
animation-delay	动画开始之前延迟的时间
animation-fill-mode	当指定动画属性时，覆盖默认值
backface-visibility	指定一个元素的翻转面在3D变换中是否可见
perspective	以3D空间建立元素，并指定视角深度
perspective-origin	指定你在3D空间中的视点
transform	使用一个2D或3D函数，指定一个元素的渲染需要改变



属性	描述
transform-origin	元素变换围绕的原点
transform-style	当变换元素嵌套时，使用它来阻止3D效果
transition	结合过渡属性的快捷属性
transition-property	定义哪个CSS属性会过渡
transition-duration	过渡持续的时间
transition-timing-function	描述过渡发生的方式（以加速的方式改变）
transition-delay	过渡开始前延迟的时间

# 第18章 CSS技术

到现在，你已经具有编写样式表的坚实基础了。你可以给文本和元素盒子创建样式，甚至还可以使用浮动元素和定位来创建网页布局。但是在到第四部分学习JavaScript之前，我还希望你了解一些常用的CSS技巧。

本章是各种各样类型的大杂烩。它从一些技术开始，这些技术是Web开发者的基本工具包的一部分：使用CSS重置，清除浏览器的样式，使用图像代替文字（只在必要时），并使用CSS的sprite，减少服务器请求的数量。接下来学习一般方法和特殊属性，给表单和表格设置样式。最后（我把最好的留在了最后），通过一步一步地练习，你会使用媒体查询创建一个自适应的网站。

## 一个干净的石板（CSS重置）

如所知，浏览器有自己的内置样式表（称为用户代理样式表）来渲染HTML元素。如果你没有为h1提供样式，可以肯定的是，它会显示为上方的黑体文本，文本上下方还有空白。不同的浏览器，文本和空白大小也不同。此外，即使你提供了自己的样式表，文档中的元素也可以继承从用户代理样式表的某些样式，从而导致意外的结果。

出于这个原因，很多设计师使用CSS重置（CSS Reset）技术，它是一系列样式规则，可以覆盖所有的用户代理样式的样式规则，并创建一个尽量中性的初始样式。之后，你必须明确地指定文档中每一个元素的字体、文本、边距和填充样式，而且可以肯定，浏览器的样式不会干扰它们。

最流行的重置是由Eric Meyer（很多CSS书的作者）写的。我把它呈现在这里，我也在本章的materials文件夹包含了它的一个副本，以便复制和粘贴。

### 本章内容

应用CSS重置

使用图像替代文本

使用CSS Sprite

样式化表单

样式化表格

使用媒体查询进行自适应设计



```

/* http://meyerweb.com/eric/tools/css/reset/
 v2.0 | 20110126 License: none (public domain)*/
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center, dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
 margin: 0;
 padding: 0;
 border: 0;
 font-size: 100%;
 font: inherit;
 vertical-align: baseline;
}
/* HTML 5 display-role reset for older browsers*/
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
 display: block;
}
body {
 line-height: 1;
}
ol, ul {
 list-style: none;
}
blockquote, q {
 quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
 content: '';
 content: none;
}
table {
 border-collapse: collapse;
 border-spacing: 0;
}

```

注意: Yahoo!的开发者也开发了一个重置。为了使用它,可以简单地把下面的代码粘贴到HTML文档的head部分。

```

<link rel="stylesheet"
 type="text/css" href="http://
yui.yahooapis.com/3.5.1/
build/cssreset/cssreset-min.
css">

```


在你这么做之前,务必先阅读它做了什么: [yuilib.com/yui/docs/cssreset/](http://yuilib.com/yui/docs/cssreset/)。

要使用重置,就将这些样式放在自己的样式表的顶部。你可以照搬使用,也可以根据需要做一些定制。我也建议你阅读Eric的帖子,了解他的思想,请访问[meyerweb.com/eric/tools/css/reset/](http://meyerweb.com/eric/tools/css/reset/)和[meyerweb.com/eric/thoughts/2007/04/18/reset-reasoning/](http://meyerweb.com/eric/thoughts/2007/04/18/reset-reasoning/)。

CSS重置并不适合每一个人。你可以决定你想依靠的浏览器的一些基本样式,而不必为每个小东西写样式。但如果你想确保所有的样式都按你的要求显示,重置也是一种途径。

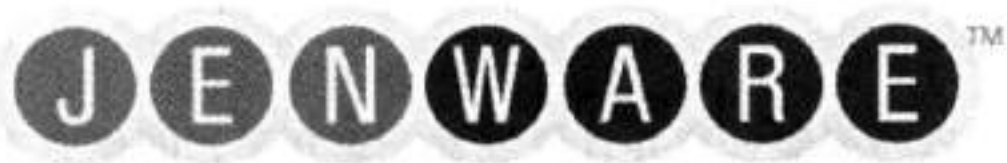
## 图像代替技术

在网页字体是一个可行的选择之前，如果想使用比Times或Helvetica更漂亮的字体，那么就要使用图像。幸运的是，现在不必如此了，无须图像，我们就可以有非常时尚的标题和文字处理。然而，有时候，如果一个web字体失效，也有必要使用图像来替代一些文字。例如，你可能想使用一个风格迥异的标识作为你的公司名称，或使用熟悉的图标代替文本链接。

完全删除文字，然后使用元素来替代是一个糟糕的主意，因为有价值的内容将丢失殆尽。解决的办法是使用一个基于CSS的图片替换技术，这个技术使用图像作为元素的背景，然后将文本转移，从而不在页面上呈现。可视化浏览器会看到背景图片，文本也依然保留在文件中，以便于搜索引擎搜集，屏幕阅读器和其他辅助设备也可以使用这些文本。这是一个多赢的方案！

一个优雅的图像替换技术来自：Scott Kellum（Jeffrey Zeldman将其命名为“The Kellum Technique”）。它使用text-indent属性将文本内容向右推出可视元素盒子的范围（如图18-1所示）。

用户看到的：



实际发生的：



图18-1：Kellum图像替换技术隐藏了HTML文本，使用text-indent把文本推出了可视元素盒子的范围

在这个例子中，我将使用花哨的Jenware标志来替换h1的“Jenware” HTML文本。该标记是很简单的：

```
<h1 id="logo">Jenware</h1>
```

这个样式的规则如下：



```
h1#logo {
 width: 450px;
 height: 80px
 background: url(jenware.png) no-repeat;
 text-indent: 100%;
 white-space: no-wrap;
 overflow: hidden;
}
```

这里有几件事情需要注意。首先，h1元素默认显示为块的情况下，就可以指定它的width和height，以匹配用作背景的图像的尺寸。text-indent属性通过设置元素宽度为100%，将文字“Jenville”推到了最右面。white-space属性设置为no-wrap，确保长字符串文字不会自动换行，从而再次出现在元素中。最后，overflow:hidden指示浏览器，任何超出了元素框的内容都不显示（如h1文本）。

其实这几年出现了很多的图片替换技术。其中最流行的是Phark技术，它使用一个非常大的负text-indent（通常为-9999px）来把HTML文本拉到视口区域的左侧。

```
h1#logo {
 width: 450px;
 height: 100px
 background: url(jenware.png) no-repeat;
 text-indent: -9999px;
}
```

**注意：**你可以使用图像替换技术来替换图像，例如，在高分辨率屏幕上（Retina）使用高分辨率图像来替换标准图像。Aaron Gustafson在他的博客上记录了这些技术（[v2.easy-designs.net/articles/iIR](http://v2.easy-designs.net/articles/iIR)和[blog.easy-designs.net/archives/2012/04/16/iir-redux](http://blog.easy-designs.net/archives/2012/04/16/iir-redux)）。

这种方法的缺点是，浏览器被强制计算并且绘制宽元素盒子，即使它不会被渲染，这会降低性能。但是，如果你遇到文本有背景图像的例子和-9999px的缩进，你就知道这是怎么回事。

任何图像替换方法的缺点是，它意味着每个图像都需要一个额外的服务器请求。在下面将学习一些方式来控制不必要的请求。

## CSS sprite

当我在第3章谈到性能时，我提醒过你可以减少页面请求（也称为HTTP请求），从而提高网站性能。减少图像请求数量的策略之一是将所有的小图像放到大图像文件中，从而只需要为一个图像获取请求。包含多个图像的大图像称为sprite，这是早期计算机图形和视频游戏行业的一个术语。图像使用background-position属性在元素中定位，这样只有用到的部分图像才是可见的。看看下面的例子会更清楚一些。

O'Reilly Media's Velocity Conference精选9种常见的社交媒体图标，如图18-2所示。为了努力改善网站的性能，Tony Quartorolo和Zebulon Young采用的策略之一是把那9个图标图形转换成一个sprite，并且减少相应的

HTTP请求数。他们将图标置入一个堆栈，并且在图标之间留有两个像素的空间。

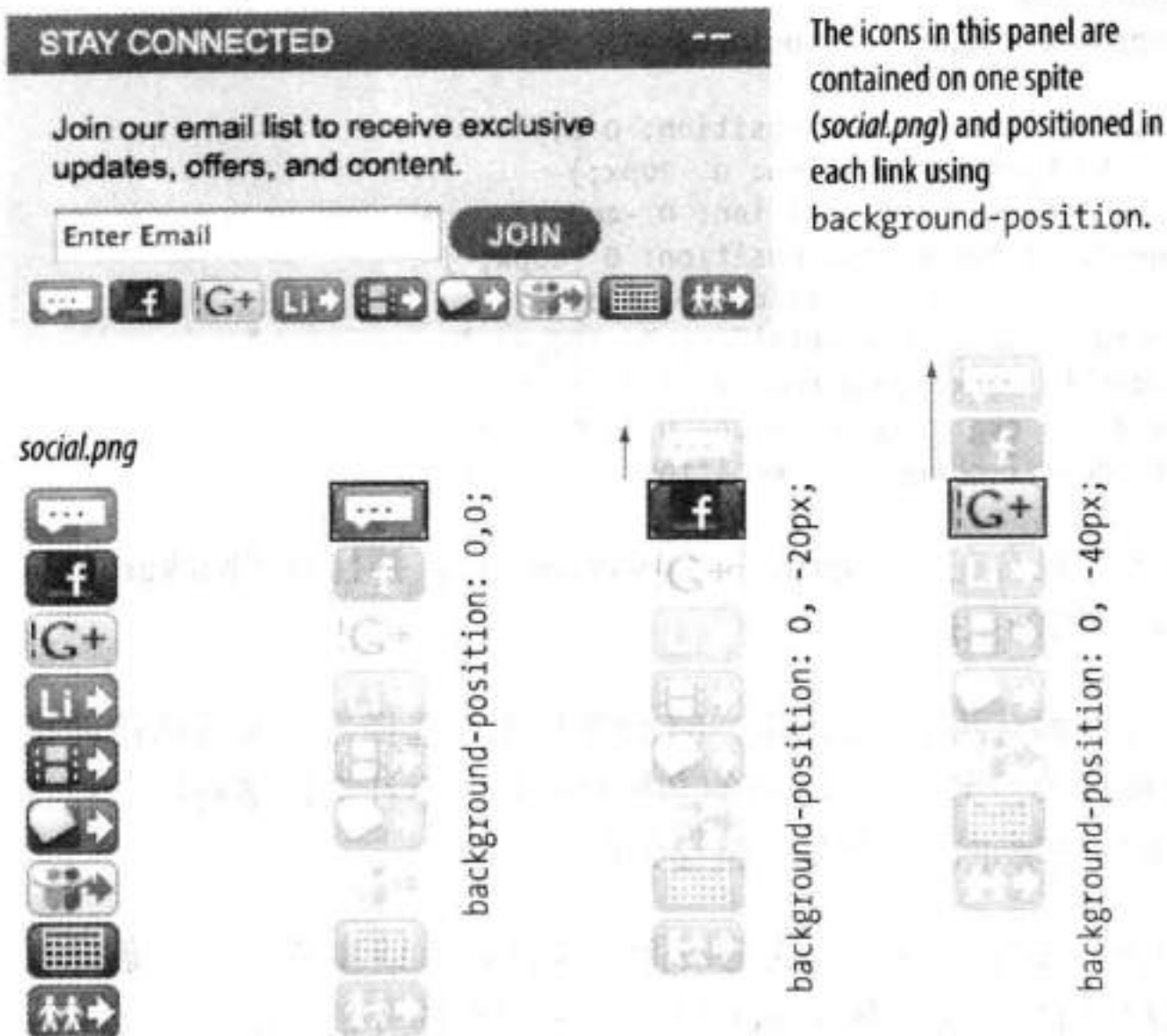


图18-2: 使用一个sprite图像来替换分隔的图像文件, 减少HTTP服务器请求数量, 提高站点性能

此处显示的样式和标记是用于Velocity网站的简化程序, 但结果是一样的。

### 标记

```

 Twitter
 Facebook
 Google+
 LinkedIn
 BlipTV
 Lanyrd
 Slideshare
 Schedule
 Attendee List

```

### 样式

```
.hide {
 text-indent: 100%;
 white-space: nowrap;
 overflow: hidden;
}
```

**注意:** 要了解更多的CSS sprite好处, 可以阅读Smashing Magazine的文章“The Mystery of CSS Sprites: Techniques, Tools, and Tutorials”, 作者是Sven Lennartz([coding.smashingmagazine.com/2009/04/27/the-mystery-of-css-sprites-techniques-tools-and-tutorials](http://coding.smashingmagazine.com/2009/04/27/the-mystery-of-css-sprites-techniques-tools-and-tutorials))。它包含了sprite的精彩例子, 包括Amazon、Facebook等使用的sprite。

**注意:** 阅读Tony和Zeb的优化过程在线文档: [radar.oreilly.com/2012/05/velocity-performance-makeover.html](http://radar.oreilly.com/2012/05/velocity-performance-makeover.html)。



### Sprite生成器

有一些工具可以创建sprite文件的工具，以及它们各自的自动样式。下面是一些工具：

- **SpriteMe** ([spriteme.org](http://spriteme.org))。SpriteMe是一个工具，可以将一个站点的图像转化到一个Sprite和样式规则中。只需要到你的站点，单击SpriteMe书签按钮，SpriteMe会分析页面，并且建议哪些图像应该合并到一个Sprite中。
- **CSS Sprite生成器** ([spritegen.website-performance.org](http://spritegen.website-performance.org))。CSS Sprite生成器是一个在线服务，可以让你上传你的单个图像，并把它们转换为Sprite和控制它的CSS。

```
li a {
 display: block;
 width: 29px;
 height: 18px;
 background-image: url(social.png);
}
li a.twitter { background-position: 0 0;}
li a.fb { background-position: 0 -20px;}
li a.gplus { background-position: 0 -40px;}
li a.linkedin { background-position: 0 -60px; }
li a.blip { background-position: 0 -80px; }
li a.lanyrd { background-position: 0 -100px; }
li a.slides { background-position: 0 -120px; }
li a.sched { background-position: 0 -140px; }
li a.attendees { background-position: 0 -160px; }
```

这个面板的图标包含在一个sprite中 (*social.png*)，并且使用background-position在每个链接中定位。

在标记中，每项都有两个class值。hide类被用作选择器，来应用前面提到的图片替换技术。另一个class的名称对应每一个社会网络链接。独特class值使我们能够正确地为每个链接定位sprite。

在样式表顶部，你应该能认出这是图像替换样式。请注意，下一条规则中，所有链接 (a) 元素都使用*social.png*作为它们的背景图像。

最后，我们来看看最重要的样式。列表中每个链接的background-position设置都不同，而可见元素盒就像一个露出部分背景图片的小窗口一样。第一项值为“0,0”，这个位置是元素框的左上角的图像。为了使Facebook图标可见，我们需要将图像上移20个像素，所以它的垂直位置设置为-20像素（水平位置0就可以了）。每个链接的图像都有20个像素的增量，表示sprite中更下面的图像。

在这个例子中，所有的图标具有相同的尺寸，堆叠得很好，但这不是必需的。你可以将不同尺寸的图像结合到一个sprite中。为元素设置尺寸的过程，以及使用background-position属性来定位的方法是一样的。

**警告：** CSS Sprite不能用于背景拼贴图。背景图像需要使用单独的图像。

### Sass和LESS

我知道你已经习惯以一般方式编写CSS规则，但是我希望你知道一些与众不同的方法。开发者Hampton Catlin和Nathan Weizenbaum发现普通CSS语法的重复，所以他们创建了一个新的样式表单语法，这种语法利用了节省时间的脚本语言工具的优点。这种方法称为新的语法Sass (Syntactically awesome style sheets)。其后的发布版就是SCSS(Sassy CSS)，SCSS就是基于原先缩进的Sass语法，但是也允许混合使用CSS语法。

在Sass样式文档中，你可以做脚本做的事情，比如为常用的值设置一个变量名。例如，O'Reilly使用了他们站点多次使用的红色阴影，所以就可以创建一个名为“oreilly-red”的变量，并且使用这个变量名来获取颜色值。通过这种方式，如果他们需要改变阴影，他们只需要在一个地方改变变量值。下面是Sass中的设置和变量用法：

```
$oreilly-red: #900;

a { border-color: $oreilly-red; }
```

你可以使用一个名叫mixins的约定来重用整个样式设置。下面的例子保存了背景、颜色和边框样式的组合，并将其称为“special”。为了应用样式的组合，在声明中@include，并且使用它的名字来调用。

```
@mixin special {
 color: #fff;
 background-color: #befc6d;
 border: 1px dotted # 59950c;
}

a.nav {
 @include special;
}
```

此外，Sass允许嵌套规则，处理数学操作，以数学方法调整色彩，只是借用了脚本语言中的一些函数名字。

浏览器并不知道如何理解.sass或者.scss文件的语法，所以你需要使用Sass编译器（Ruby编写），它在服务器上运行。这个编译器可以在把文件传递给浏览器之前，将Sass文件转换为标准的CSS语法。

LESS是另一个使用脚本类似功能的CSS语法。它很类似于Sass，但是它缺少了一些功能，在语法上有些不同（比如，LESS中的变量用符号@表示，而非\$，例如：@oreilly-red）。另一个主要的不同是，LESS文件通过JavaScript处理成规则的CSS语法（less.js），而不是Ruby。注意，把一个LESS文件编译为CSS是处理密集的，可能会拖垮浏览器。因此，最好在传到服务器之前进行CSS转化。可以完成这个功能的推荐工具是Codekit（[incident57.com/less/](http://incident57.com/less/)），当然还有其他的工具。

一旦你经过一些练习，并且认为你准备好进一步学习样式表，可以阅读下面的Sass和LESS文章和资料：

- Sass 站点 ([sass-lang.com](http://sass-lang.com))
- LESS 站点 ([lesscss.org](http://lesscss.org))
- Compass, a full-featured CSS authoring framework that uses Sass ([compass-style.org](http://compass-style.org))
- “GettingStartedwithSass”，作者David Demaree ([alistapart.com/articles/getting-started-with-sass](http://alistapart.com/articles/getting-started-with-sass))
- “An Introduction to LESS, and Comparison to Sass”，作者 David Hixon ([coding.smashingmagazine.com/2011/09/09/an-introduction-to-less-and-comparison-to-sass](http://coding.smashingmagazine.com/2011/09/09/an-introduction-to-less-and-comparison-to-sass))



## 样式化表单

没有风格样式的Web表单看起来会有一点乱（图18-3），所以你肯定想用CSS让它们看起来更专业。不仅是因为更好看，而且研究表明，当标签和输入一字排开时，使用表单是更方便快捷的。在这个章节中，我们将了解如何给多样的表单设置风格，以及在不使用表格的情况下排列表单元素。

现在，我想说的是：样式化表单是某种黑暗艺术，因为浏览器在处理表单元素时有各种形式。但努力是非常值得的，这样会使你的表单看起来和网页的其他部分一样专业。

### Contest Entry Information

Name

Email

Telephone

Your story

Size  Sizes reflect standard men's sizes

Color

☐ Red

☐ Blue

☐ Black

☐ Silver

Features

☐ Sparkley laces

☒ Metallic logo

☐ Light-up heels

☐ MP3-enabled

### Contest Entry Information

Name

Email

Telephone

Your story

Size  Sizes reflect standard men's sizes

Color ☐ Red ☐ Blue ☐ Black ☐ Silver

Features ☐ Sparkley laces

☒ Metallic logo

☐ Light-up heels

☐ MP3-enabled

图18-3：仅使用HTML会使表单看起来很丑很难用（左图）。使用一点CSS就可以带来很大改观。本节会带你一步一步学习样式化表单。

样式化表单没有任何特殊属性，只需使用标准颜色、背景、字体、边框、边距和填充属性，在前面的章节中你已经学到了。以下是每个表单控制类型适用属性的简要总结。

### 文本输入（text, password, email, search, tel, url）

用width, height, background-color, background-image, border, border-radius, margin, padding和box-shadow改变盒子本身的外观。你也可以使用color和不同的字体属性，来样式化输入的字段。

### textarea元素

它也像文本输入域一样进行样式化。textarea的元素默认情况下，使用等宽字体，所以你需要改变它，以便与你的其他文本输入域相匹配。由于有多行，你也可以指定line-height。请注意，有些浏览器的文本框的右下角有个拖动手把可以用来缩放，但是你可以使用resize:none来关闭它。

## 按键输入 (submit, reset, button)

将盒子属性应用到提交和复位按钮 (width, height, border, background, margin, padding和box-shadow)。但是请注意,默认情况下按钮被设置为border-box尺寸模型,这样width和height值可以应用于多个边框。大多数浏览器还默认添加了一点填充,它可以被你自己的padding值覆盖。你也可以对按钮显示的文本设置样式。

## 单选按钮和复选框按钮

对于单选按钮和复选框按钮的最佳做法就是不管它们。最好情况下,IE会在其周围显示一点颜色,这看起来很奇怪。如果你还不肯放弃,你可以使用JavaScript改变这些按钮。

## 下拉和选择菜单

你可以指定一个select元素的width和height,但请注意,默认情况下,它使用border-box尺寸模型。有些浏览器允许你对options设置color、background和字体属性,但也许最好让浏览器按照默认方式渲染。

## Fieldset和legend

你可以把fieldset看做其他的元素盒子,调整border、background、margin和padding。完全关闭边框是保持你的表单整洁的一种方法,同时保持语义和可访问性。默认情况下,legend元素的位置的缩进,fieldset的上边框垂直居中,遗憾的是,浏览器使它们难以改变。一些开发人员在legend中使用一个span或b元素,并且对包含元素应用样式,以产生更可预见的结果。

现在我们知道如何为单独的控件应用样式,但更大的目标是使表单更有条理和易于使用。在过去,我们使用表格来实现,但最好是坚持使用CSS以求良好的显示,比如对齐的问题。我将引领你一步步从A到B,为表单手动编写样式表,如图18-3所示。

## 标记文本

这是contest entry表单的标记文本。表单中每一个问题都被包含在一个列表项中,所有控件都提供了标签。还有两个fieldset,将单选按钮和复选框分组。

```
<form action="" method="">
<h2>Contest Entry Information</h2>

 <label for="form-name">Name</label>
```



注意：你可能注意到，这个例子中的表单类似于第9章完成的比赛输入表单。为了使本节的讲解简单一些，我简化了它。

```

 <input type="text" name="username" id="form-name" class="textinput">

 <label for="form-email">Email</label>
 <input type="email" name="emailaddress" id="form-email"
 class="textinput">

 <label for="form-tel">Telephone</label>
 <input type="tel" name="telephone" id="form-tel" class="textinput">

 <label for="form-story">Your story</label>
 <textarea name="story" maxlength="300" id="form-story" rows="3"
 cols="30" placeholder="No more than 300 characters long"></textarea>

 <label for="sizes">Size</label>
 <select name="size">
 <option>5</option>
 <option>6</option>
 <option>7</option>
 <option>8</option>
 <option>9</option>
 <option>10</option>
 <option>11</option>
 <option>12</option>
 <option>13</option>
 </select>
 Sizes reflect standard men's sizes

 <fieldset id="colors">
 <legend>Color</legend>

 <label><input type="radio" name="color" value="red"> Red
 </label>
 <label><input type="radio" name="color" value="blue">
 Blue</label>
 <label><input type="radio" name="color" value="black">
 Black</label>
 <label><input type="radio" name="color" value="silver">
 Silver</label>

 </fieldset>

 <fieldset id="features">
 <legend>Features</legend>

 <label><input type="checkbox" name="feature" value="laces">
 Sparkley laces</label>

 <label><input type="checkbox" name="feature" value="logo"
 checked>
 Metallic logo</label>

 </fieldset>


```

```


 <label><input type="checkbox" name="feature" value="heels">
 Light-up heels</label>

 <label><input type="checkbox" name="feature" value="mp3">
 MP3-enabled</label>

</fieldset>

<li class="buttons">
 <input type="submit" value="Pimp My Shoes!">
 <input type="reset">

</form>

```

## 第1步：添加基本样式

第一组样式负责一些基本的文档样式，包括body、h2和一些标准的ul样式来移除项目符号。我还为这个form元素创建了规则，设定了它的宽度、背景颜色、圆角、阴影和填充。作为一个浮动容器，由于将浮动很多的内容，所以添加了overflow:hidden。同样地，ul li规则为了浮动也包括了一个clear:both;声明。为了节省一点空间，这里只介绍表单相关的样式。结果如图18-4所示。

```

ul li {
 clear: both;
 ...
}
form {
 width: 40em;
 border: 1px solid #666;
 border-radius: 10px;
 box-shadow: .2em .2em .5em #999;
 background-color:#d0e9f6;
 padding: 1em;
 overflow: hidden;
}

```

图18-4：添加样式后的表单元素



## 第2步：对齐的标签和输入

现在是关键一步！请注意，对图18-3的表单设置样式后，在所有的标签和输入整齐的排为一列。为了让这种情况发生，需要给label元素一个特定的宽度，并把它们浮动到左侧，然后将标签文字右对齐，这样标签就在各自的输入附近。在label元素右边添加一点边距，可以使列看起来很美。你应该能够在label样式规则中看到所有这些样式。

**注意：**我已经给不同的文本输入类型（text、tel等）添加了一个class="textinput"属性。所以我才能只选取文本输入的内容。我也可以对每个文本输入使用属性选择器（比如，input[type="tel"]），但是有些IE版本无法支持。我之所以选择可靠的class方法，是因为我希望每个人都能看到这些样式。

```
label {
 display: block;
 float: left;
 width: 10em;
 text-align: right;
 margin-right: .5em;
 color: #04699d;
}
```

文本输入框和textarea设置了width和height值，以及1像素的边框，此外，类似的字体属性也都被两者应用。我通过设置resize为none，使textarea元素不能调整大小。图18-5的表单看起来又好了一点，但现在我们有一些关于单选按钮和复选框标签的问题。

图18-5：对齐后的标签和输入

```
input.textinput {
 width: 30em;
 height: 2em;
 border: 1px solid #666;
}
textarea {
 display: block;
 width: 30em;
 height: 5em;
 border: 1px solid #666;
 margin-bottom: 1em;
 line-height: 1.25;
 overflow: auto;
 resize: none;
}
input.textinput, textarea {
 font-family: Georgia, "Times New Roman", Times,
 serif;
 font-size: .875em;
}
```

## 第3步：修复Fieldset和小标签

接下来要做的就是覆盖fieldset元素的默认样式，使它们不那么突出。我也为每个fieldset的legend使用与label相同的样式。

完成步骤2中添加的样式后，fieldset中单选按钮和复选框的标签也与主标签的样式相同，但这并不是我想要的风格。我编写了一些规则，尤其是为名为Color和Feature的fieldset中的label元素编写了规则。最后，显示的Color部分的列表项为内联，这样它们就出现在同一行，并节省一些空间。Features复选框列表需要一些小的调整，如添加一些左边距，使复选框与其他的表单控件排成一行（margin-left:11em），并且重置clear属性，使第一个复选框列表项不低于浮动legend（clear:none）。结果如图18-6所示。

```

fieldset {
 margin: 0;
 padding: 0;
 border: none;
}
legend {
 display: block;
 width: 10em;
 float: left;
 margin-right: .5em;
 text-align: right;
 color: #04699d;
}
#features label, #colors label {
 color: #000;
 display: inline;
 float: none;
 text-align: inherit;
 width: auto;
 font-weight: normal;
 background-color: inherit;
}
#colors ul li {
 display: inline;
 margin-bottom: 0;
}
#features ul {
 margin-left: 11em;
}
#features ul li {
 margin-bottom: 0;
 clear: none;
}

```

图18-6：修正复选框和单选按钮旁的标签

#### 第4步：调节按钮

现在所要做的，是解决对齐，并且对提交和重置按钮应用样式（图18-7）。通过给提交按钮添加左边距，将按钮与其他表单控件对齐。我也给按钮设置了新的尺寸、背景颜色、圆角边框和下阴影。font-size:inherit;声明确保按钮使用的字体大小与表单其他部分相同（覆盖浏览器的默认值），从而使测量em值可预见。



```
input[type="submit"], input[type="reset"] {
 display: block;
 width: 10em;
 height: 2em;
 float: left;
 background: white;
 font-size: inherit;
 border: 1px solid #04699d;
 border-radius: 4px;
 box-shadow: 2px 2px 3px rgba(0,0,0,.5);
}

input[type="submit"] {
 margin-left: 10.5em;
 margin-right: 1em;
 color: #C00; /* submit按钮文本是醒目的红色*/
}
```

图18-7：使用样式和对齐按钮的最终表单

现在你应该明白了！我已经集中讲解了例子中对齐、色彩和文本处理的样式。对于你自己的表单，你可能想添加交互性的样式，比如对选中的按钮和文本输入分别添加: hover样式和: focus样式。

## 样式化表格

我们已经讲述了大多数的样式属性，你还需要样式化表格中的内容。你可以使用各种字体、文字和背景属性更改单元格中内容的外观和对齐方式，就像你对其他文本元素那样。此外，你还可以使用填充、边距和边框处理自己的表格和单元格。

但是有几个CSS属性专用于表格。其中有些比较深奥的属性在侧栏“高级表格属性”中作了简要介绍。本节侧重于直接影响表格显示的属性，尤其是边框处理的属性。

分散的和折叠的边框

CSS提供两种方法来显示单元格之间的边框：separated或collapsed。当边框分开时，各个单元格的边框在四个方向都向内退，你可以指定各个边框间的空白。在折叠边框模型中，相邻边的边框“折叠在一起”，这样只能看到一个边框，而且之间的空白也没有了（如图18-8所示）。

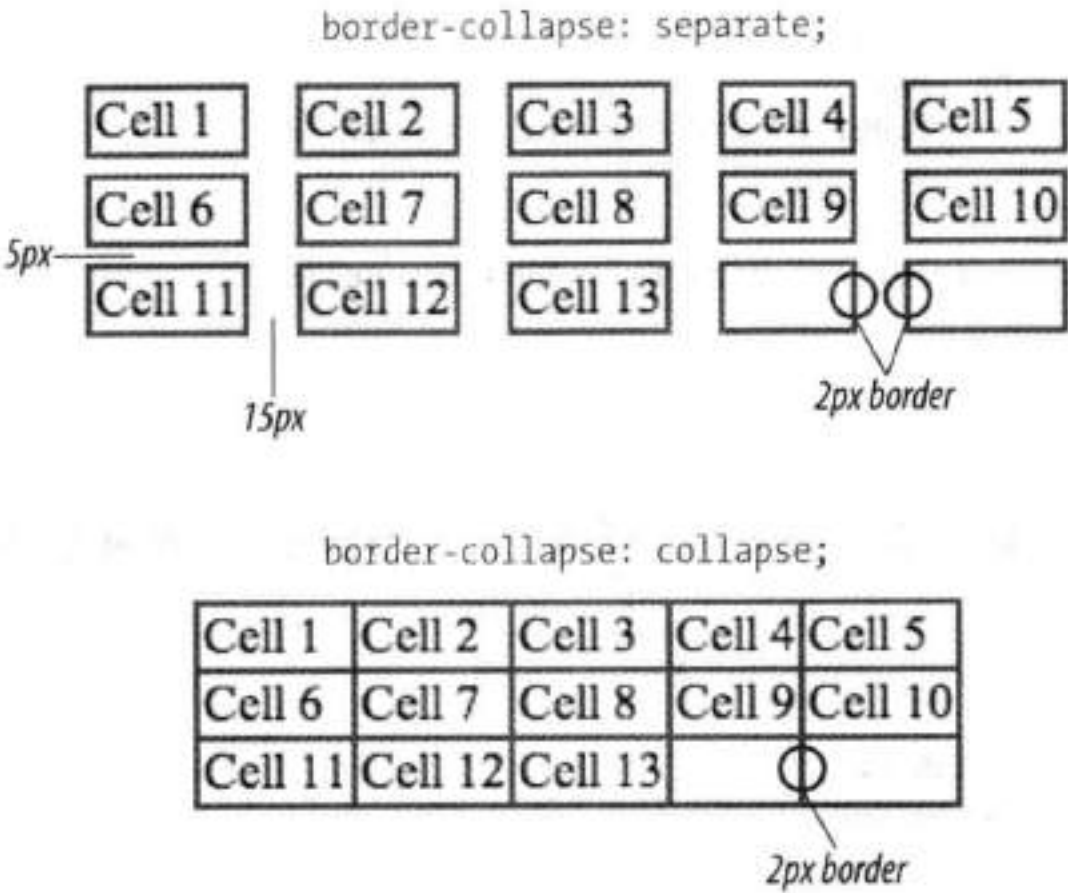


图18-8：分开的边框（上）和折叠的边框（下）

Border-collapse属性允许作者选择使用哪种边框显示方法。

border-collapse

- 属性值： separate|collapse|inherit
- 默认值： separate
- 适用对象： 表格和表格嵌入元素
- 是否可继承： 是

分散边框模型

表格默认是分散地显示的，如图18-8中的上图。你可以使用border-spacing属性指定你想显示的单元格间距值。

border-spacing

- 属性值： 长度值 长度值|inherit

高级表格属性

还有一些与CSS表格模型相关的属性。

表格布局

Table-layout属性允许作者选择两个计算表格宽度的方法之一。fixed值以表格、列或单元格的width属性值作为表格宽度。auto值以表格内容的最小宽度为表格宽度。auto布局方法名义上显示得比较慢，因为浏览器在知道表格宽度之前，必须先计算每个单元格的默认宽度。

表格display属性值

第14章介绍了display属性用来指定布局中产生的元素是哪种盒子。CSS在设计上与XML语言协作，而不仅是与HTML和XHTML协作。其他语言也需要表格布局，但是不需要table、tr或者td这些词汇。

最后，还有多种与表格相关的display属性值，允许XML语言的作者给任何元素指派表格布局行为。与表格相关的display属性值有table、inline-table、table-row-group、table-header-group、table-footer-group、table-row、table-column-group、table-column、table-cell和table-caption。你可以把这些显示角色指派给其他的HTML元素，但通常不鼓励这样做。

警告： IE 6并不支持border-spacing属性。



默认值: 0  
 适用对象: 表格和表格嵌入元素  
 是否可继承: 是

`border-spacing`的属性值是两个长度计量值。水平值在前，并应用于列间。第二个计量值用于行间。如果你只提供一个值，那么该值将用于水平和垂直两个方向。默认设置为0，导致边框重叠在表格的格子中。

有一些样式规则用于创建自定义的边框间距，如图18-8中的上图。

```
table {
 border-collapse: separate;
 border-spacing: 15px 3px;
 border: none; /* no border around the table itself */
}
td {
 border: 2px solid purple; /* borders around the cells */
}
```

### 压缩边框模型

当选择了压缩边框模型时，在单元格间只会显示一个边框。下面就是创建图18-8中的样式表。

```
table {
 border-collapse: collapse;
 border: none; /* no border around the table itself */
}
td {
 border: 2px solid purple; /* borders around the cells */
}
```

注意，虽然每个单元格都有2像素的边框，但单元格之间的边框值共为2像素，而不是4像素。单元格之间的边框在单元格中格子的中间，所以如果单元格设置4像素的边框，那么每个单元格分配2像素。如果像素值是个奇数，那么就由浏览器来决定多余的1像素分配给谁。

在相邻单元格边框样式不同的情况下，一个复杂的次序被引入，来决定显示哪个边框。如果任意一个单元格的`border-style`属性设置为`hidden`值，那么边框将不会显示。然后，还要考虑边框的宽度：宽边比窄边优先。最后，如果所有其他属性都相等，那么就要归结为样式的问题了。CSS创建者将边框样式按优先级从大到小排列：`double`、`solid`、`dashed`、`dotted`、`ridge`、`outset`、`groove`和（优先级最低的）`inset`。

**注意：**虽然`border-spacing`默认为0，但是浏览器默认会给`cellspacing`属性添加2像素的空间。如果你想看到这种效果，那么需要在`table`元素中把`cellspacing`属性设置为0。

## 空单元格

分散边框的表格中，利用empty-cells属性，可以决定是否让空单元格显示背景和边框。

### empty-cells

属性值： show|hide|inherit  
 默认值： show  
 适用对象： 单元格元素  
 是否可继承： 是

对于一个“空”的单元格，它可能不包含任何文本、图像或者非换行空白。它可能包含回车和空白字符。

图18-9显示了上一个分散表格边框的例子，空单元格（Cell 14 和 Cell 15）设置为hide状态。

```
table {
 border-collapse: separate;
 border-spacing: 15px 3px;
 empty-cells: hide;
 border: none;
}
td {
 border: 1px solid purple;
}
```

Cell 1	Cell 2	Cell 3	Cell 4	Cell 5
Cell 6	Cell 7	Cell 8	Cell 9	Cell 10
Cell 11	Cell 12	Cell 13		

图18-9：使用empty-cells属性来隐藏空单元格

## 简单的自适应Web设计

自适应Web设计（也称为响应式Web设计）是一种技术，它使用CSS，根据屏幕大小以适应页面的布局。这仅是我们采取的策略，用来应付令人发疯的各种尺寸的屏幕。

当然，自适应Web设计是一个非常庞大的主题，足够写整本书（而且有人已经写了）。在这里，我将给你介绍一些基本的自适应网页设计，使你有一些基本的感觉。这里介绍的方法是基于Ethan Marcotte在他的里程碑式的著作自适应网页设计（Responsive Web Design）中描述的方法。当你读到这本书的时候，我很确信关于这个主题会有很多的书，还不包括在线的很多信息（可以阅读本章最后的侧栏：进一步阅读）。需要说明的



是，在你完成本节的练习之后，你前往自适应网页设计殿堂的路才刚刚开始。

一个简单的例子

接下来将共同努力，使Jenware页面成为自适应网页设计的典范。图18-10显示了相同的Jenware HTML页面，但是分别显示在狭窄屏幕、纵向和横向的平板电脑屏幕，以及一个大型台式机显示器中。



图18-10：新的自适应Jenware站点。你可以在你自己的移动设备上查看：[www.learningwebdesign.com/rwd/](http://www.learningwebdesign.com/rwd/)

注意： 为了更多的自适应性，可以看看媒体查询站点 ([mediaqueries.com](http://mediaqueries.com))。

在智能手机的屏幕上，页面上有一列布局 and 很窄的侧边距。平板在纵向模式下，有更丰厚的空间来容纳边距和换行的文本。在1024像素宽的屏幕可以容纳第二列，并且在很宽的浏览器窗口中，使用max-width属性来限制内容的宽度，以确保控制行长。与专业的自适应设计相比，还有非常温和的调整，这些对于初学者应该是足够的。

它是如何工作的

Marcotte先生首先提出自适应设计时，包含三个核心组件。

流动布局

在第16章CSS布局中，你已经了解了全部的流动布局，幸运的是，Jenware站点已经被设计为流体布局。

灵活的图像

当布局按比例缩小时，图像和其他嵌入式媒体需要随之缩小，否则，他们会离开视域。我们会确保Jenware图像适度缩小。

CSS媒体查询

媒体查询是一个方法，它根据显示文档的介质来应用适当的样式。查询从问题开始，比如，“是要打印的文档吗？那就使用适合打印

的样式”。或者，“是在屏幕上显示文档吗？屏幕至少为1024像素宽吗？是在横向模式下吗？那就使用这些样式”。我会立刻告诉你如何用CSS语法来实现。

对于以上这些组件，我想添加视窗的meta元素，它使网页的宽度与屏幕宽度匹配，我们也将从这里开始自适应设计之旅。

## 设置视窗

为了标准的网站适应小屏幕，移动浏览器会在画布上渲染页面，画布称为视窗，然后缩小视窗，以适应屏幕宽度（设备宽度）。例如，在iPhone上，移动Safari浏览器设置视窗宽度为980像素，就好像是将桌面浏览器窗口设置为980像素宽，而页面就显示在这个视窗上。但是，当iPhone横过来就会被缩小为320像素宽，就需要把大量的信息压入这个狭小的空间中。

移动Safari浏览器引入了视窗meta标签，允许开发人员控制初始视窗的大小。其他移动浏览器很快效仿，这也是自适应设计必不可少的第一步。只需在HTML文档的head部分添加下面的meta元素：

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

这一行告诉浏览器，任何时候都把视窗宽度设置为设备屏幕的宽度（width=device-width）。initial-scale设置缩放级别为1（100%）。

现在可以开始对Jenware站点进行自适应设计了。我们会一步一步地练习，从练习18-1开始。

### 练习18-1 设置视窗大小

在本练习中，你将熟悉Jenware的材料并在编辑样式表之前设置视窗。*jenware-rwd.html*和*jenware.css*都在本章的*materials*文件夹中。经过前面的练习，你应该很熟悉这个页面了，但我已经修改了一些小样式，以便让你轻装上阵。

1. 首先，在浏览器中打开文件*jenware-rwd.html*。*jenware.css*处理基本样式，例如背景、颜色、边框和文字风格等，提供了良好的样式。将窗口的大小缩到很窄，以适应智能手机的宽度。在图18-10中，你应该会看到类似iPhone的屏幕截图，只Jenware标志图形挂在屏幕的右边缘。向下滚动，可以看到#products和#testimonials在窗口的右边缘。
2. 现在将窗口尽量拉宽。你应该感到该页面宽的不舒服，文本并没有沿着产品图像换行。这种设计在宽浏览器上显然需要忍耐。

**注意：** 视窗meta元素也支持maximum-scale属性。把它设置为1（maximum-scale=1）可以阻止用户缩放页面，但是强烈建议你不要这么做，因为缩放对于可访问性和可用性非常重要。

**注意：** 本章练习需要使用Chrome、Firefox、Safari或者IE 9及其以上版本。IE 8和之前的版本不支持将要使用的媒体查询。



### 自适应布局

作为替代方法，尤其是当没有预算进行真正的自适应设计时，一些设计师选择了创建自适应布局作为替代方法。自适应布局针对大多数常用设备，使用两个或三个不同的*fixed*布局。这做起来要容易一些，但却失去了流动布局的优势。一些人认为自适应布局顶多是权宜之计，而不是长期移动设计策略。

**注意：** 为了保持缩放视频的长宽比，你需要使用更多的技术。Thierry Koblentz在他的文章“*Creating Intrinsic Ratios for Video*”中介绍了这种策略，你可以在[www.alistapart.com/articles/creating-intrinsic-ratios-for-video](http://www.alistapart.com/articles/creating-intrinsic-ratios-for-video)获取。[fitvidsjs.com](http://fitvidsjs.com)上还有一个JavaScript方案。

**警告：** IE 6不支持max-width属性。

3. 让我们看看视窗的meta元素。在文本编辑器打开jenware-rwd.html，添加标准的meta元素，如下所示：

```
<meta name="viewport" content="width=device-width,
initial-scale=1">
```

保存文件，就完成了。因为这是一个移动的网站，当你在桌面浏览器上查看页面时，你不会注意到任何变化，但其实已经进行了改进。

## 流体布局

由于流体布局是自适应设计的基础。流体布局使用百分比宽度，这样的元素按比例调整大小以填充屏幕或窗口的可用宽度。

创建一个适应所有可能的设备宽度的页面是不可行的。网页设计师通常针对主要设备，如智能手机、平板电脑和桌面浏览器，创建两个或三个设计（有时甚至更多）。它们依靠流体布局来照顾所有可能的尺寸。

因为在前面练习中我已经用到了流体布局风格，我们不需要为Jenware做什么。对于你自己的项目，也一定要灵活的设计。说到了灵活的设计，让我们对这个标志图像做点什么！

## 使图像灵活起来

举个例子，需要一些样式规则，使图像缩小到适合其容器的大小：

```
img {
 max-width: 100%;
}
```

当布局变得越来越小时，它里面的图像也将缩小，以适应它们所在容器的宽度。如果容器是大于图像，例如，在平板或桌面的布局中，图像并不放大；它保持其原始大小的100%。当你应用max-width属性，在HTML文档中，要确保img元素没有宽度和高度属性，否则图像不能按比例缩放。

别急，事情永远不会这么简单，对不对？我很担心，虽然样式规则很简单，但是在移动设备上显示图像是更大的问题，且更加复杂。即使在简单的例子中，服务于智能手机的图像都比实际需要的大，这意味着不必要的数据传输。我要在本章的后面“自适应图像”中再次重新审视这一难题。现在，只需要记在心里。

在继续练习之前，我需要提醒，缩放其他的嵌入式媒体，如object、embed或video（见“注意”），这里还是要使用max-width。



图18-11: max-width属性使得图像与它所在的容器同比例缩小

## 练习18-2 伸缩这些图像

这也是一个简单的练习。打开 `jenware.css`，在样式表的 `body` 规则后，添加图像缩放样式。

```
img { max-width: 100%; }
```

保存文档并在浏览器中查看。当你把窗口缩小到很窄时，logo图像也会随之缩小（如图18-11所示）。产品图像也是如此，但是视窗如果太小，你也就看不清了。

## 媒体查询技术

现在，正式步入自适应设计殿堂：媒体查询。

媒体查询允许设计者提供基于媒体类型的样式。定义的媒体类型有：`print`、`screen`、`handheld`、`braille`、`projection`、`screen`、`tty`和`tv`。关键字`all`表示样式适用于所有类型的媒体。媒体查询还可以评估特定的媒体特征，如`device-width`、`orientation`和`resolution`。大多数属性可以使用`min-`和`max-`前缀来分别测试最小值或最大值。例如，`min-width:480px`测试显示设备是否至少480像素宽。768像素宽的显示设备会通过测试并且得到样式，而320像素宽的显示设备则无法通过测试，也无法得到样式。

你可以使用媒体查询检测的设备特征完整列表在表18-1中列出。

你可以在样式表中添加媒体查询。下面是一个媒体查询样式的例子，它可以决定媒体类型是否为屏幕，以及这个屏幕是否为至少480像素宽：

```
@media screen and (min-width: 480px;) {
```



```
/* Put styles for devices & browsers that pass this test inside the curly braces */

}
```

查询以@media开始，后面跟着目标媒体类型关键字（在例子中为屏幕screen）。所测试的媒体特征和值都包含在括号中。符合这些条件的浏览器的样式规则被放到大括号之间。

表18-1：能用媒体查询求值的媒体特征

特征	描述
width	显示区域（视窗）的宽度
height	显示区域（视窗）的高度
device-width	设备渲染面（整个屏幕）的宽度
device-height	设备渲染面（整个屏幕）的高度
orientation	设备的方向是横向还是纵向（不接受min-和max-前缀）
aspect-ratio	视窗的宽高比率（宽度/高度）
device-aspect-ratio	整个屏幕（渲染面）的宽高比率
color	显示器的位深度；例如，color:8可以测试设备是否至少为8位色
color-index	色彩查询表中的色彩号
monochrome	单色设备中每个像素的位数
resolution	设备的像素密度。它与检测高分辨率显示器密切相关
scan	一个tv媒体类型是否使用前进或隔行扫描（不接受min-和max-前缀）
grid	设备是否使用基于网格的显示，比如一个固定宽度的字体（不接受min-和max-前缀）

下面是另一个例子，测试两个特征值：屏幕宽是否小于700像素，屏幕是否为横向。请注意，每对特征和值都被放置在括号内。字符串“and”将各种要求连在一起。该设备必须通过所有的要求，才能使用大括号中的样式。

```
@media screen and (max-width: 700px;) and (orientation: landscape;) {
 /* put styles for devices & browsers that pass this test here */
}
```

最后，在这个例子中，媒体查询测试设备是否具有高密度显示，例如，视网膜屏幕的iPhone、iPad以及新的MacBook Pro。这个例子有供应商前缀的查询，以及一个标准的查询。在这里，不同的查询在一个逗号分隔的列表中。当所有查询条件都满足时，就可以使用括号中的样式。

```
@media screen and (-webkit-min-device-pixel-ratio: 2),
 screen and (-moz-min-device-pixel-ratio: 2),
 screen and (-o-min-device-pixel-ratio: 2),
 screen and (-ms-min-device-pixel-ratio: 2),
 screen and (min-device-pixel-ratio: 2) {

 /*style refererencing high-resolution images here */

}
```

## 文档head中的媒体查询

@media的查询就在样式表中。媒体查询，也可以在link元素中使用media属性来实现，当条件满足时，就载入相应的.css文件。

在这个例子中，首先提供了网站的基本样式，然后就是一个样式表，这个样式表只有当设备超过780像素宽时才使用（当然，设备的浏览器需要支持媒体查询）。

```
<head>
 <link rel="stylesheet" href="styles.css">
 <link rel="stylesheet" href="2column-styles.css" media="screen and
 (min-width:780px)">
</head>
```

一些开发人员发现这种方法有利于管理模块化的样式表，但它的缺点是需要为每个.css文件进行额外的HTTP请求。请务必只提供必要的链接，并在样式表中依靠@media规则轻微地调整尺寸（注1）。

## “移动为先”的媒体查询

上面讲述了实现的机制，现在来谈一下策略。自适应网站最重要的是采取“移动第一”的思想。这意味着，随着更多的设备和特征的到来，你首先需要照顾最小、最简单的设备，并使用媒体查询来引入相应的样式。（如果你觉得这听起来像一种逐步增强的形式，那你理解对了。）

“移动为先”的媒体查询，往往从min-前缀开始，当宽度至少为指定的宽度或更大时，就引入新的样式。这允许开发者在更简单的已用样式基础上不断增加新的样式。

注1： 关于这项技术详见Stephanie Rieger的文章“Pragmatic Responsive Design”。你可以在[www.slideshare.net/yiibu/pragmatic-responsive-design](http://www.slideshare.net/yiibu/pragmatic-responsive-design)上了解更多信息。

---

**警告：** IE 8和之前的版本不支持媒体查询。我将在练习18-3中展示一个弥补方法。

---



---

**注意：** 为了更好地总结“移动为先”的设计方法，可以阅读Brad Frost的文章“Creating a Mobile-First Responsive Web Design” ([www.html5rocks.com/en/mobile/responsivedesign/](http://www.html5rocks.com/en/mobile/responsivedesign/)) 以及他的帖子“Anatomy of a Mobile-First Responsive Web Design” ([bradfrostweb.com/blog/mobile/anatomy-of-a-mobile-first-responsive-web-design](http://bradfrostweb.com/blog/mobile/anatomy-of-a-mobile-first-responsive-web-design))，他在这里深入描述了这个思想。这是了解移动网页设计师思想的机会。

---



**警告：** 确保正确地嵌套并关闭了花括号。媒体查询的最后一个花括号很容易就忘记写了。

请记住，无论是一个样式表的规则，还是链接元素的规则，后来的样式都会覆盖之前的样式。所以，基准样式应该放在最前，其次是小设备的样式，然后是大浏览器的增强样式。这正是在练习18-3中要完成的工作。

### 练习18-3 添加媒体查询

现在会添加一些样式，它们会根据显示区域的宽度来改变布局。我已经为你完成了设计工作。你可以复制这里完成样式，也可以从`materials`文件夹中复制`jenware-final.css`。

1. 在文本编辑器中打开`jenware.css`。当前的样式表创建了边对边的一列设计，对于窄屏幕来说，它工作良好，但是当屏幕比较宽时，看起来就会很糟糕。我认为应该支持横向或纵向使用的智能手机（最大480像素宽），此外，还需要给所有元素一些呼吸的空间。
2. 我首先至少为481像素宽的设备添加了样式。通过一点额外的空间，可以把产品图像浮动到左边，并清除其后的“More about...”链接。我也在白色的`#products`盒子周围添加了边距，并且对`#testimonials`盒子应用了圆角和边距，就像在第14章和第15章所做的练习那样（如图18-12所示）。这里的媒体查询样式在样式表的最后部分，这样它就可以选择性地覆盖前面的属性。

```
media screen and (min-width: 481px) {
 #products img {
 float: left;
 margin: 0 6px 6px 0;
 }
 #products .more {
 clear: left;
 }
 #products {
 margin: 1em;
 }
 #testimonials {
 margin: 1em 5%;
 border-radius: 16px;
 }
}
```

注意，我们会检测显示区域的宽度（`width`），而不是整个屏幕的宽度（`device-width`），因为在移动设备上显示时，页面周围往往会有应用镀边。使用`width`检测的结果更准确。

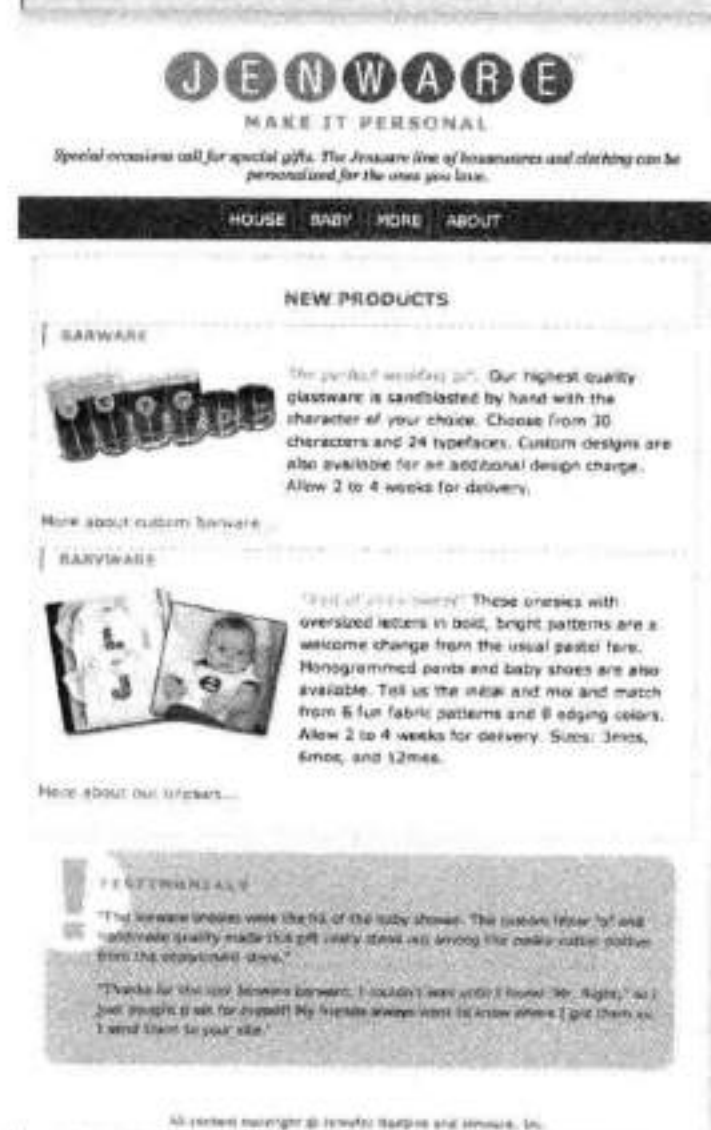


图18-12：应用平板样式后的Jenware站点



3. 下一步是当显示区域至少为780像素宽时设置样式。媒体查询中的样式通过浮动#products div到左侧，并且对#testimonials左侧应用宽边距，创建了一个两列布局。版权段落被清除了，所以它只能显示在页面底部。最后在#content div中设置了max-width，使得内容显示不会超过1024像素宽，即便浏览器超过了这个宽度（图18-13）。

在样式表单中，这一步的媒体查询应该在刚刚添加的样式之后。

```
@media screen and (min-width: 780px) {
 #products {
 float: left;
 margin: 0 2% 1em;
 clear: both;
 width: 55%;
 overflow: auto;
 }
 #testimonials {
 margin: 1em 2% 1em 64%;
 }
 p#copyright {
 clear: both;
 }
 #content {
 max-width: 1024px;
 margin: 0 auto;
 }
}
```

图18-13：宽屏幕上的Jenware站点

4. 现在你可以保存文档，并在浏览器中查看（使用Chrome、Safari、Firefox或者IE 9）。试着调整窗口，看看布局如何调整。你觉得你的第一个自适应网页看起来如何？

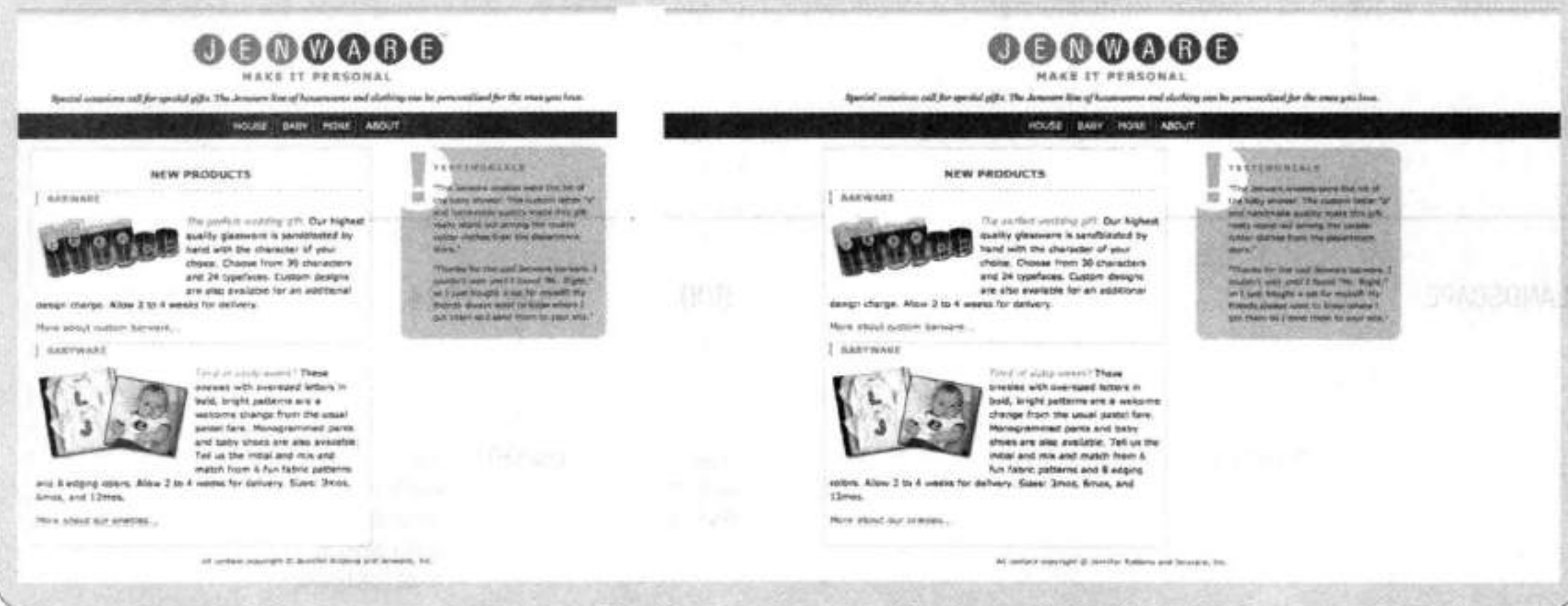
但是IE 8及其之前版本的浏览器该怎么办呢？

就像我前面提到的，IE 8及其之前的版本不支持媒体查询，所以这些媒体查询样式会被忽略。这意味着使用IE 8的桌面用户也只能看到一列布局，最小比率的页面。这样感觉很不好。解决方案是把适用于桌面浏览器的样式提取出来，并保存在单独的样式表中，这个样式表只对非移动的IE 9之前的版本起作用（lt IE 9）&(!IEMobile)。

如果你想简单一些，可以复制媒体查询中的样式（但是不包括媒体查询符号），并且把它们粘贴到一个名为ie-layout.css的文件中。从第一个媒体查询开始，包含浮动图像和圆角#testimonials盒子的样式。从第二个媒体查询开始的样式都适用于桌面，所以也把它们都粘贴过来。

IE特有的条件注释提供了到这个特殊样式表的链接，它必须要在其他样式表链接之后。你可以把下面的代码添加到jenware-rwd.html的head部分。

```
<link rel="stylesheet" href="jenware.css">
<!--[if (lt IE 9)&(!IEMobile)]>
 <link rel="stylesheet" href="ie-layout.css">
<![endif]-->
```





棘手的位

Jenware称得上是自适应设计，但它进行了简化，并只代表一些最好的情况。要成为自适应网站，需要一定的规划和工作。因为移动Web是相对较新的，开发社区仍然面对手机设计所面临的挑战和工作。接下来将介绍RWD和移动设计所带来的一些棘手问题和限制。

选择断点

自适应设计中一个主要的设计决策是确定在哪个宽度引进重大设计变化。媒体查询提供了一组新的样式，称为一个断点 (breakpoint)。有些站点只在一个单一的断点触发两列布局。更普遍的是，自适应网站使用三种设计来针对典型的电话、平板电脑和桌面宽度，我甚至看到过多达五个。选择多少取决于站点的设计本质。

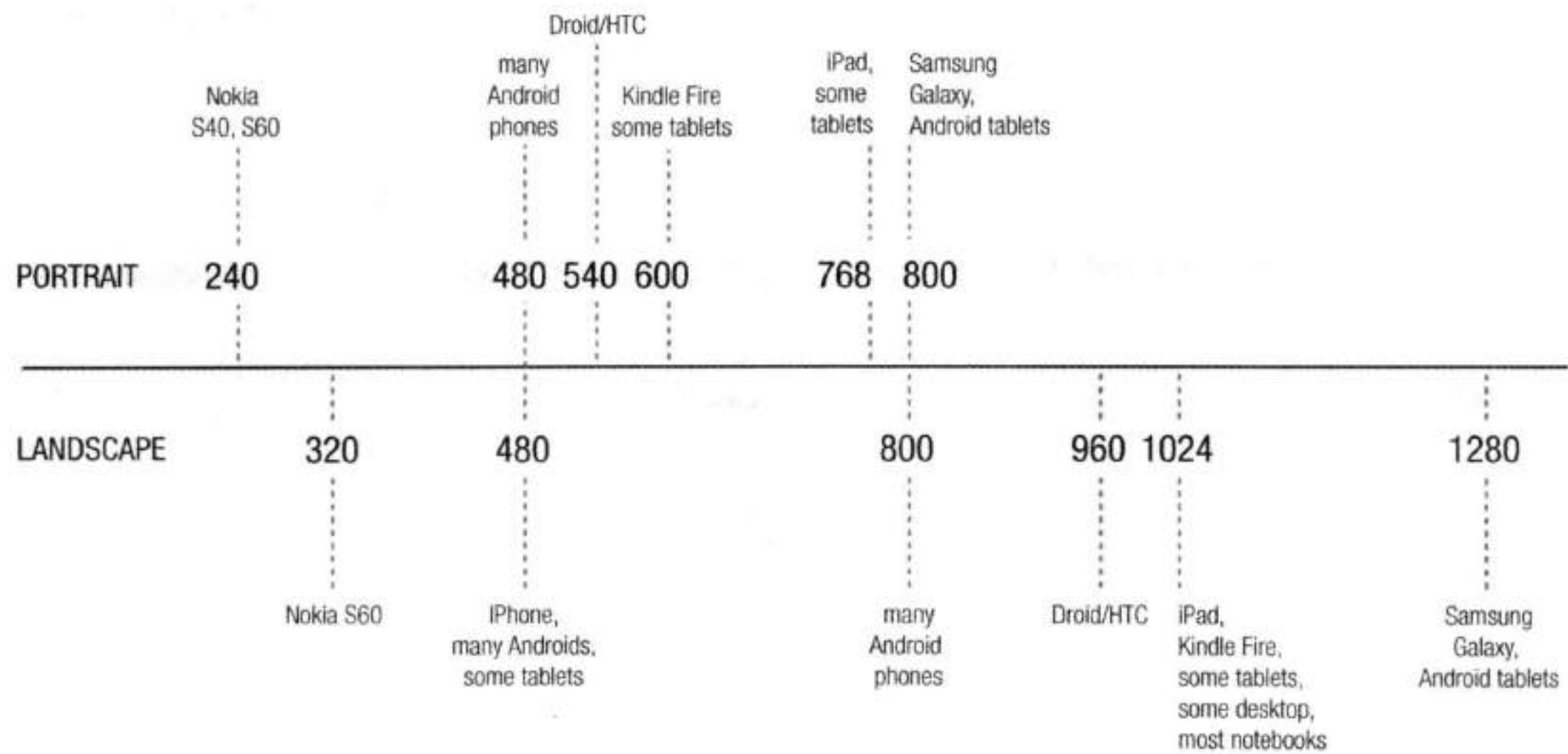
但该如何选择断点？一种方法是使用流行设备的像素尺寸，就像对Jenware所做的一样。图18-14显示了一个断点图表，列出了当前最流行的设备在横向和纵向时的尺寸。

现实情况是，新的设备宽度一直在不断涌现，我们不能指望创建一个万能的设计。出于这个原因，媒体查询从像素值走向了更利于开发的em值。许多开发者让他们的内容自行确定断点应该在哪里发生，这也让事情开始看起来非常糟糕！

工具提示

站点ResizeMyBrowser.com可以调整你的浏览器，就像它的域名描述的一样。单击列在屏幕的一个尺寸（如320×480），这个网站就会把你的浏览器窗口调整为你选择的大小。它可以让你在不同设备宽度下测试你的网站。然后你得先知道，如果在实际设备测试，那么没有代替的方法。这只是设计过程中的一个好用的工具而已。

图18-14：断点图显示了一些流行设备的像素宽度



Breakpoint chart by Viljami Salminen (with adaptations) from his article "Responsive Workflow" (viljamis.com/blog/2012/responsive-workflow/).

根据单列、宽单列和多列来思考，然后以em值确定逻辑断点，这是一个比较好的方法。要了解更多信息，我建议你阅读文章“The Ems have it: Proportional Media Queries FTW!” 作者 Lyza Gardner ([blog.cloudfour.com/the-ems-have-it-proportional-media-queries-ftw/](http://blog.cloudfour.com/the-ems-have-it-proportional-media-queries-ftw/))。

## 自适应图像

移动设备的Web开发人员最棘手的问题之一是如何得到正确的图像。在理想的情况下，移动设备下载的图像应该与其尺寸和网络速度相匹配。我们的目标是避免下载不必要的数据，当只需要一个图像时，既不下下载超过屏幕大小的图像，也不下载两个版本的图像（低分辨率和高分辨率）。

由于知道设备的大小并不能了解网络速度，所以图像问题比较复杂。小手机可能会使用较慢的EDGE网络或速度更快的WiFi。Retina iPad又迫切需要较大的图像，但可能会在3G网络下载。而且，你可能并不希望简单地小屏幕缩小图像。在某些情况下，它可能优先使用已经裁剪得完全不同的图像，以在小屏幕显示重要细节。

在撰写本书时，似乎有更多的辩论，但没有解决办法。有些人提出了新的HTML标记，使在屏幕分辨率的基础上，更容易指定图像文件尺寸。有些人认为服务器需要发挥更大的作用，特别是议定网络速度。另一些人的解决办法是，新的图像格式可以包含多个版本的相同图像。移动网络的使用爆发使网络高科技技术落后了。

在网络上搜索为“自适应图像”应该可以帮助你了解最新的情况。Jason Grigsby写了几个高知名度的文章，确切地描述了2012年的困境。为了解面临的挑战和可能的解决方案，可以先看看这些文章。

- “Responsive IMGs—Part 1” ([blog.cloudfour.com/responsive-imgs/](http://blog.cloudfour.com/responsive-imgs/))
- “Responsive IMGs Part 2—In-depth Look at Techniques” ([blog.cloud-four.com/responsive-imgs-part-2/](http://blog.cloud-four.com/responsive-imgs-part-2/))
- “Responsive IMGs Part 3—Future of the IMG” ([blog.cloudfour.com/responsive-imgs-part-3-future-of-the-img-tag/](http://blog.cloudfour.com/responsive-imgs-part-3-future-of-the-img-tag/))
- “The Real Conflict Behind <picture> and @srcset” ([blog.cloudfour.com/the-real-conflict-behind-picture-and-srcset/](http://blog.cloudfour.com/the-real-conflict-behind-picture-and-srcset/))

## 一个尺寸无法适合所有情况

CSS可以很好地更换样式，在屏幕上（甚至隐藏）左右移动元素。但是，在许多情况下，不同的内容或者不同顺序的相同内可以更好地适应较小

### 在服务器上调整图像尺寸

Sencha.io是个服务，可以在传递过程中缩小图像，然后以合适的尺寸发送到设备上。你所需做的是给img添加一些额外的标记，使其指向Sencha.io服务器。

Sencha.io服务器使用用户代理字符串（浏览器用来识别的文本）来在数据库中查询设备。一旦确定了宽度，Sencha.io就会把图像缩小到这个宽度，然后发回小文件。

如果要了解更多，可以参考[docs.sencha.io/0.1.3/index.html#!/guide/src](http://docs.sencha.io/0.1.3/index.html#!/guide/src)。



### 进一步阅读

由于技术在不断地进步，所以Web是了解自适应设计开发的最好平台。作为一本全面介绍的书籍，我推荐Brad Frost的移动网页最佳实践站点 ([mobilewebbestpractices.com](http://mobilewebbestpractices.com))。在资源页面上，Brad搜集了优秀文章、书籍、画廊、显示、脚本以及移动相关开发站点的列表。

关于这个主题还有很多书，但是我认为最有用的是*Head First Mobile Web* (O'Reilly)，作者是Lyza Danger Garden和Jason Grigsby，还有一本书*Implementing Responsive Design* (Peachpit/New Riders)，作者是Tim Kadlec。

的设备。JavaScript可以处理一些重新安排，并提供了一种方法来有条件地加载内容。使用JavaScript的自定义内容不是本节的范围，但你应该知道，内容的调整是可能的，而且在移动环境设计中需要考虑。

### 自适应的限制

对于一些网站，特别是文字为主的网站，如博客，自适应设计就是要使内容在小屏幕上可以愉悦地浏览。然而，对于其他网站，只调节样式是不够的。当一个网站或服务在移动情况下的使用与桌面上使用明显不同，那么可能有必要建立一个独立的移动网站。

但即使是单独的移动网站也可以利用这里介绍的自适应设计方法。自适应技术是每一个网页设计师必备的技能。

## 小结

现在已经基本讲完了样式表。目前使用CSS格式化文本和完成一些基本的布局应该没什么难度。精通样式表的窍门当然是大量的实践和测试。如果遇到了困难，你可以发现网上有很多的资源，可以帮助你找到需要的答案。

在第四部分，我将我的工作交给JavaScript大师Mat Marquis，他将向你介绍JavaScript和它的语法（某种程度上，他可以非常有趣地进行讲解）。我将在第五部分中谈论网络图形。

## 自我测验

利用这些问题，看看本章CSS技术你掌握得如何。答案还是在附录A中。

1. CSS重置的目的是什么？
  - a. 覆盖浏览器默认的风格
  - b. 使不同浏览器的显示更可预见
  - c. 阻止元素继承不希望的风格
  - d. 以上所有
2. CSS sprite的目的是什么？
  - a. 改进站点性能
  - b. 使用小图片替代大图像，减小文件大小
  - c. 减少HTTP请求的数量
  - d. a和c

- e. 以上所有
3. 说出LESS和Sass的两个区别。
4. 图像替代技术的目的是什么？
- 获取大的文本缩进
  - 使用装饰图像替代文本
  - 从文档中移除文本，并且使用装饰图像来替代
  - 保持文档的语义内容
  - b和d
  - 以上所有
5. 不使用表格，对齐表单控件和各自标签的秘诀是什么？做具体描述。
6. 设置视窗尺寸的重要性是什么？
7. 匹配媒体查询与各自的意义。
- |                                                                                     |                               |
|-------------------------------------------------------------------------------------|-------------------------------|
| a. @media screen and (max-width: 800px) { }                                         | _____以纵向模式打印时应用这些样式           |
| b. @media screen and (min-device-width: 800px) { }                                  | _____对黑白媒体应用这些样式              |
| c. @media print and (orientation: portrait) { }                                     | _____当显示区域至少为800像素宽时应用这个外部样式表 |
| d. <link rel="stylesheet" href="special.css" media="screen and (min-width: 800px)"> | _____当显示区域小于800像素宽时应用这些样式     |
| e. @media all and (monochrome) { }                                                  | _____当整个设备屏幕至少为800像素宽时应用这些样式  |
8. 把下面的样式规则与图18-15的表格匹配起来。
- table { border-collapse: collapse; }  
td { border: 2px black solid; }
  - table { border-collapse: separate; }  
td { border: 2px black solid; }
  - table {  
border-collapse: separate;  
border-spacing: 2px 12px; }  
td { border: 2px black solid; }



```
4. table {
 border-collapse: separate;
 border-spacing: 5px;
 border: 2px black solid; }
td { background-color: #99f; }

5. table {
 border-collapse: separate;
 border-spacing: 5px; }
td {
 background-color: #99f;
 border: 2px black solid; }
```

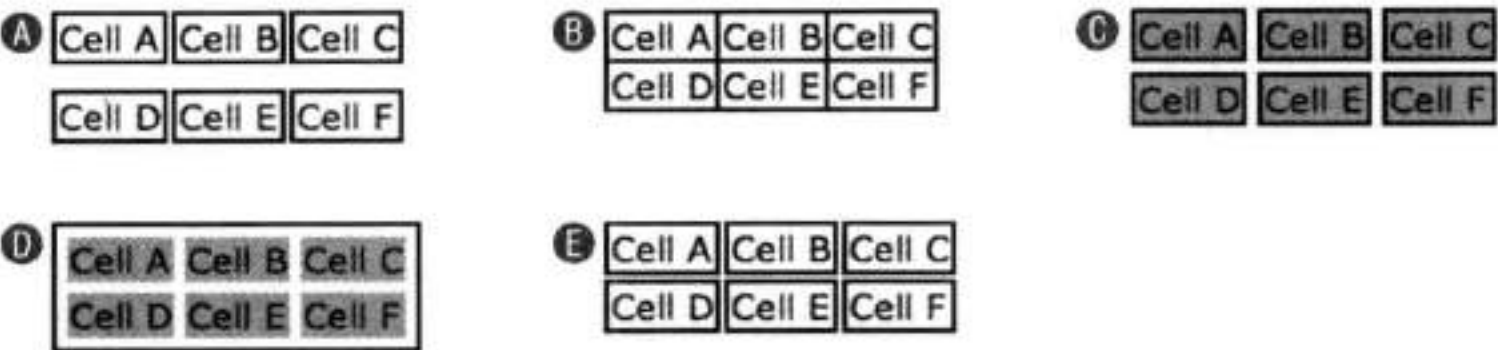


图18-15：将这些表格与问题8中的代码示例匹配

## CSS回顾：表格属性

下面是本章涵盖的属性的总结。

属性	描述
border-collapse	单元格的边框是分散的还是重叠的
border-spacing	分散时单元格的间距
empty-cells	单元格的边框和背景是否显示

# JavaScript行为

## 第四部分

---

### 本部分内容

第19章

JavaScript简介

第20章

使用JavaScript





# 第19章 JavaScript简介

作者Mat Marguis

在这章中，我将会向你介绍JavaScript。现在你也许有些担忧，我能理解。下面我们全面进入编程语言领域，也许会有点令人害怕，不过，我保证，并不会那么糟糕。

我们会从什么是JavaScript，什么不是JavaScript以及它所使用的方式来重温它。本章主要介绍JavaScript语法，包括变量、函数、操作符、循环等。本章结束时，你是否就会编程了呢？也许不会。但是当你看到一个脚本时你会对将要发生的事有一个好的理解。当你能使用脚本来操作浏览器窗口以及诸如单击或提交表单这类用户行为时，本章就结束了。

## 什么是JavaScript

如果你曾经用过，毫无疑问你已经知道了JavaScript是一种对网站添加互动以及自定义行为的程序语言。这是一个客户端脚本语言，意味着它在用户的电脑而不是像PHP和Ruby这类网络编程语言一样在服务器上运行。这意味着JavaScript（以及我们使用它的方式）依赖于浏览器的兼容性及设定。由于用户卸载又或是由于设备不支持，而导致JavaScript并不总是可用的，这一直是优秀的开发者想要解决的问题。JavaScript也被认为是一种动态的、松散类型的编程语言。不要对这个描述过于苦恼，之后我会解释它。

首先，我要说的是，JavaScript是一种误解。

## JavaScript不是什么？

从一开始，这个名字就是颇有争议的。尽管它的名字是JavaScript，但与Java没有任何关系。它是由Brendan Eich于1995年开发的，最初命名为LiveScript。因为当时Java风靡一时，因此为了营销的需要，将LiveScript改为JavaScript。如果你想在谈论JavaScript时听到比较酷的名字，可以将其简

### 本章内容

什么是JavaScript，什么不是  
变量和数组  
if/else语句和循环  
原生和自定义函数  
浏览器对象  
事件句柄



称为JS。

当然，JS也会有些不好的地方。它与各类肆意的网络恶作剧——如多余的重定向，令人讨厌的弹出式窗口，以及许多安全漏洞隐患——同义。曾经有段时间JavaScript允许小部分著名的开发商去做这些事，但是现代的浏览器运营商大部分已经知晓了JavaScript开发的弊端并且已经屏蔽了它。但是，在那个年代，我们不该指责JavaScript本身。因为正如一个古语所说，“能力越强，责任越大”。JavaScript允许开发商对网页构想以及浏览器如何运行进行大量的控制，同时这取决于我们如何负责任地使用这种控制权。

## JavaScript是什么？

现在我们知道了JavaScript不是什么：它与Java无关；也不是浏览器中潜藏的狰狞“反派”。让我们更多地谈论下JavaScript是什么。

JavaScript是一种轻巧但是非常强大的脚本语言。我们经常通过浏览器看见它，但是JavaScript的使用已经渗透到了很多方面，从自带程序到PDF再到电子书，甚至是网络服务器也需要JavaScript的技术支持。

作为一个动态编程语言，JavaScript并不需要任何形式的编译器，将可读的代码译为浏览器可读的代码。浏览器能如迅速有效地读取代码。

JavaScript也是松散型的。这就意味着在JavaScript中我们并不需要解释变量是什么。如果设定一个变量值为5，那么我们并不需要编程指定变量为一个数。正如你也许已经注意到的，5本身就是一个数，JavaScript也是这样识别它的。

现在你不需要记住这些术语，留心就可以了，说实话，我也没有记住，虽然我已经读过了。现在仅仅是向你介绍一些学习JavaScript时会听到的术语，随着学习的深入，你会了解得更多。这同样也为你的下一个鸡尾酒会提供了交谈的素材。“噢，我么？近来我已经熟悉了松散类型的脚本语言。”人们会安静地向你点头，我认为这就意味着在谈话中你做得很好。当然我不会参加很多的鸡尾酒会。

## JavaScript能做什么？

通常情况下，我们会在给网页添加互动性时用到JavaScript。网页的结构层是我们的标记；网页的表象层是由CSS构成的；第三个行为层是由JavaScript组成的。网页上的所有元素、属性和文本都能通过使用DOM（文档对象模型）的脚本来获得，这些内容将在第20章中涉及。我们也可以写下脚本来对用户的输入做出反馈，或者变更网页内容、CSS样式或浏览器的行为。

---

**注意：**1996年欧洲电脑制造商协会（ECMA）将JavaScript标准化，这就是有时候也将其称作ECMAScript的原因。

---

如果你已经尝试注册网站、登记用户名，那么你会立即获得反馈，即你登记的用户名已被使用（如图19-1所示）。文本输入的红色边框以及“对不起，该用户名已被使用”的信息是JavaScript变更网页内容的表现；阻止表单提交是JavaScript改变浏览器错误行为的表现。

#### Whoops! Some errors occurred.

- That username is already in use.
- Email confirmation doesn't match

Username   
Must be at least 4 characters

Email

Confirm Email

Password

Confirm Password

图19-1：JavaScript发现用户名不可用，然后插入了一个信息，改变样式使问题明显化

总之，JavaScript允许在没有载入新页面时创建一个高度自适应的界面，这个界面有助于提升用户体验，同时提供动态功能。例如，可以使用JavaScript来完成以下事情：

- 提示用户输入搜索框的完整术语。你可以在谷歌网站看到（如图19-2所示）。

what can javascript do

what can javascript do

what can javascript be used for

what can javascript do for a website

what can javascript programs do

图19-2：Google使用JavaScript来自动完成搜索术语

- 不需要重新载入整个网页，只需来自服务器的内容和信息，同时根据需要将其载入当前文档，这称为Ajax。
- 基于用户单击链接和标题，展示或隐藏内容，来创建一个“可折叠”的内容区域（如图19-3所示）。
- 对浏览器的特征以及性能进行测试。例如，对触摸事件的存在进行测试，要求用户通过移动设备的浏览器与网页交互，以及添加更多方便触摸的样式和交互方法。

Section 1

Section 2

Collapsible content

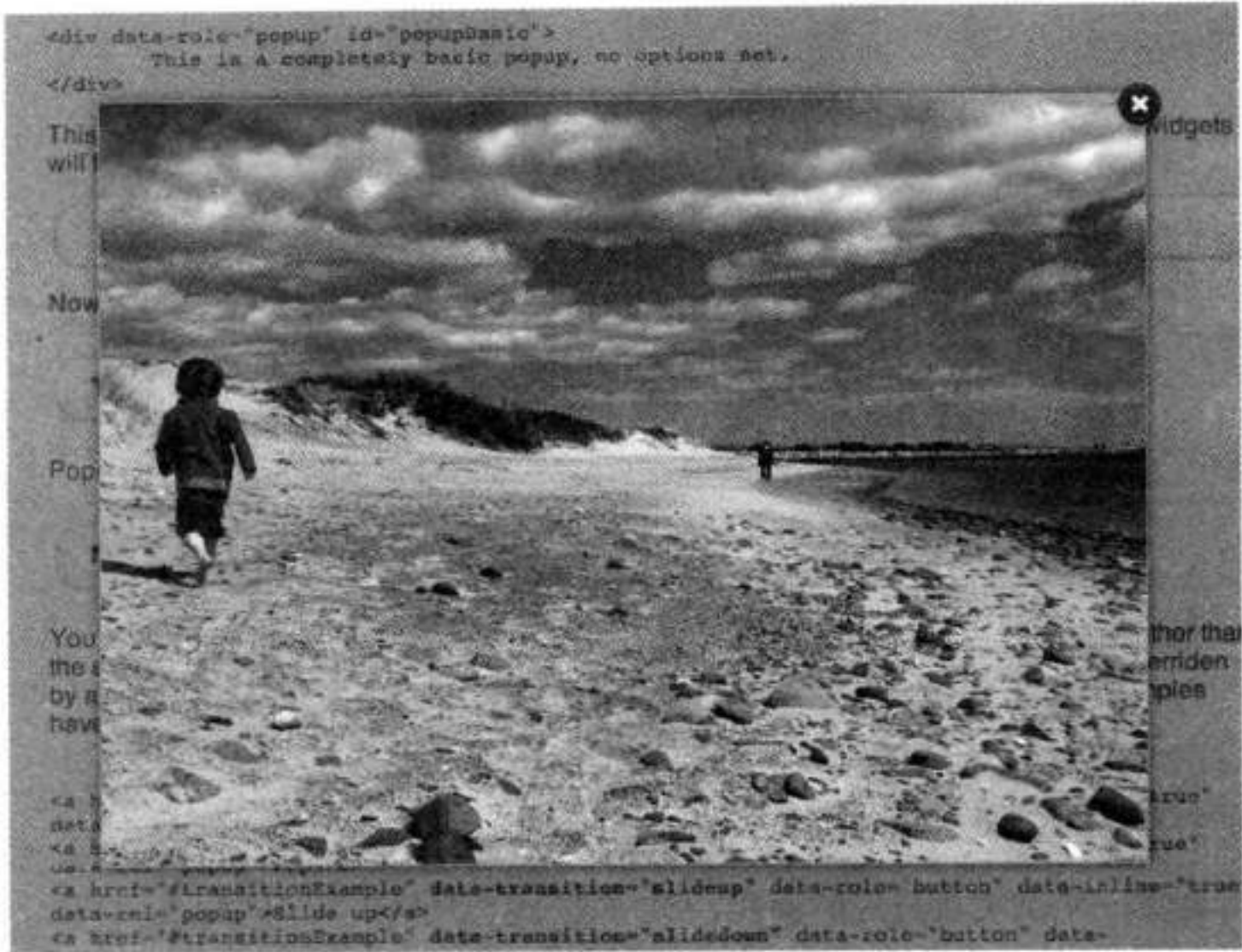
Section 3

图19-3：JavaScript可用来显示或隐藏部分文本



- 弥补浏览器嵌入功能缺失的不足，或者在新浏览器中添加一些新的功能。这些种类的脚本语言称为shims或polyfills。
- 在用户单击一个小图标之后，在自定义样式后的lightbox中加载图片或内容，lightbox使用CSS，位于界面上（如图19-4所示）。

图19-4：JavaScript能用于加载图片



注意：在HTML4.01中，script标签必须包括type属性，以便操作有效：

```
<script type="text/javascript">...</script>
```

对于XHTML文档而言，你必须把脚本元素的内容作为CDATA，就像下面一样：

```
<script type="text/javascript">
 // <![CDATA[
 ...JavaScript code goes here
 //]]>
</script>
```

外部脚本

其他方法是使用src属性，通过URL指向脚本文件（以.js结尾）。在以下例子中，脚本元素没有内容。

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

外部脚本的优势是你可以将相同的脚本用于多个网页。缺点是，每个外部脚本都需要一个额外的服务器HTTP请求，而这又会减慢运行速度。

脚本位置

script元素可置于文档的任何位置，但最普遍的位置是文档的head和body的末尾。需要记住的是，不要通过文档来设置，因为这样难以寻找和继续。

对大多数脚本而言，</body>标签之前的文档结尾，是最合适的位置，因为浏览器会用于解析文档，及其DOM结构。因此，使用脚本时，信息既是可用的，又运行迅速。此外，脚本下载以及运行阻碍了网页的显示，

这个目录并不是太详细。

给网页添加JavaScript

如同CSS，你可以在文档中嵌入一个脚本，或者在外部文件中嵌入一个脚本，并把它链接到网页。这两种方法都使用script元素。

嵌入的脚本

在网页中嵌入脚本，只需添加代码作为script元素的内容。

```
<script>
... JavaScript code goes here
</script>
```

所以将脚本置于底部有助于改善性能。然而，大多数情况下，你可能在主体完全下载之前，就想让脚本做一些事情，因此，将其置于head性能会更好。

## 脚本剖析

以下是David Flanagan的*JavaScript: The Definitive Guide*这本书长达1100页的原因。关于JavaScript有许多内容可以讲述。在这里，我们将用少许篇幅给你打下一个基础，以便当你看见脚本语言时，能有所理解。许多开发者通过找到存在的脚本，并且将其为自己所用。在多次练习之后，他们就能编写自己的代码了。你可能也希望学会自己编写JavaScript，同时完成网页设计技能的训练。识别脚本是第一步，我们将从这里开始。

最初，JavaScript的功能仅限于与用户交互。我们能使用JavaScript的一些内置功能（图19-5）给用户反馈，比如，`alert()`向用户提供一个告示，`confirm()`要求用户赞成或拒绝一项行为。为了达到要求用户输入的目的，我们或多或少受制于内置的`prompt()`函数。虽然这些方法仍在使用，但它们是不合适的，至少与用户交流起来并不方便。随着JavaScript的不断改进，已经提供了许多对网页添加行为，以及为用户提供满意体验的好方法。

为了充分利用这些方法，必须首先理解脚本的基础逻辑。这些逻辑与所有编程语言都相同，即便语法类型多样。即使各语言间的语法不同，但为了在编程语言以及词汇之间寻求平衡，许多的语法模式也是共享的。

最后，你将了解有关变量、数组、比较操作，以及if/else声明、循环、函数等内容，准备好了吗？

### 基础

以下是一些JavaScript普遍的语法规则。

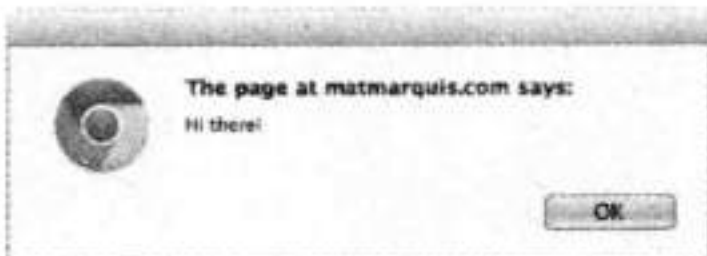
要知道，JavaScript是区分大小写的。不同的命名，`myVariable`、`myvariable`和`MYVariable`被认为是三种不同的对象。同时如制表符、空格等空白是被忽略的，除非它们是一串文本和引号中的一部分。

### 声明

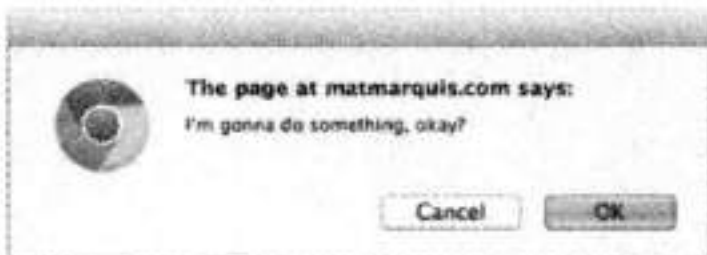
一个脚本是由一系列声明构成的。一个声明就是一个指令，告知用户做什么。这里有一个简单的声明，使得浏览器显示带有“Thank you”短语的警示。

```
alert("Thank you.");
```

```
alert("Hi there");
```



```
confirm("I'm gonna do something, okay?");
```



```
prompt("What should I do?");
```

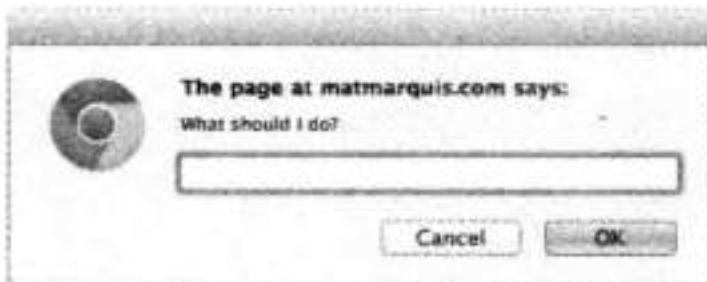


图19-5：内置的JavaScript功能：`alert()`（上图）、`confirm()`（中图）和`prompt()`（下图）

JavaScript是区分大小写的。



声明最后的分号意味着这是指令的末尾，如同句号结束一句话一样。根据JavaScript标准，一个换行将会引发一个指令的结束，但最好用分号结束每个声明。

## 注释

JavaScript允许进行注释，在运行脚本时它是被忽略的，因而你可以在代码中进行备注。如果此编码有可能在将来被其他的开发者编辑，那么注释便相当有帮助。

以下有两种方法进行注释。对于单行注释，在开头使用两个反斜线（//）。你可以将单行注释与一个声明置于同一行，只要放在声明之后即可。注释并不需要关闭，换行就可以有效地结束注释。

```
// This is a single-line comment.
```

多行注解使用CSS中同样的语法。所有置于/\* \*/中的内容都是被浏览器所忽略的。在解决问题时，你可以使用这个方法将提示语句和大块脚本都进行注释。

```
/* This is a multi-line comment.
Anything between these sets of characters will be
completely ignored when the script is executed.
This form of comment needs to be closed. */
```

我会使用单行注释对示例编码进行简短的备注，同时也会充分利用alert()函数（如图19-5所示），以便迅速查看工作的结果。

## 变量

如果像我一样，那么“变量”术语就会引发如同八年级数学课的那段令人不愉快的时光。虽然此刻你的老师并不是很糟糕，但前提是一样的。

一个变量就像一个信息容器。你可以给它命名并且赋值，这个值可以是一个数字、一个字符串、DOM中的一个元素又或者是一个函数。这有助于日后通过名称来引用该值。该值可通过脚本逻辑所指定的方式来修改和重新赋值。

以下声明生成了一个名为foo、值为5的变量。

```
var foo = 5;
```

通过关键词var来声明变量。单等号(=)表明赋值。因为语句已经完成，所以以分号结束。声明变量也可以没有关键词var，关键词会影响脚本中能够获取信息的那部分。我们将会在本章后面的“变量范围和var关键字”那部分进行深入讨论。

---

一个变量就像一个信息容器。

---

你可以使用任何你喜欢的内容来给变量命名，但确保命名是有意义的。你应该不会以“数据”来给一个变量命名吧，命名应该描述它所包含的信息。在上述具体的例子中，“数字5”会比“foo”更实用。以下是一些命名规则：

- 必须以字母或下划线开始。
- 在任何组合中必须包含字母、数字和下划线。
- 不包含字符空格。作为替代，可在空格处或靠近空格处使用下划线，或使用突出的大小写来替代。（例如，`my_variable`或`myVariable`）。
- 不包含特殊字符（“!”、“.”、“,”、“/”、“\”、“+”、“\*”、“=”等）。

在脚本中，可通过重新声明来随时改变变量值。记住：JavaScript是区分大小写的，变量命名也是。

## 数据类型

变量的赋值属于不同的数据类型。

### 不明确的

数据类型中最简单的是“不明确的”。如果声明了一个变量，但是没有赋值，那么该变量就包含一个“不明确的”值。

```
var foo;
alert(foo); // This will open a dialog containing "undefined".
```

你可能不会立刻发现这些用处，但为了解决在JavaScript中面临的差错，知晓这些是值得的。如果一个变量拥有一个本不该有的“不确定的”值，你也许应该复核这个变量是否声明正确或者变量名称有没有打印错误。

### NULL

与以上类似，对变量赋予一个“null”（记住，区别大小写哦）。简单来说，“定义该变量，但没有给予固定值”。

```
var foo = null;
alert(foo); // This will open a dialog containing "null".
```

### 数字

你可以给变量赋予数字值。

```
var foo = 5;
alert(foo); // This will open a dialog containing "5".
```

就JavaScript而言，单词“foo”现在意味着与数字5同义。因为



JavaScript是松散类型，所以不需要在脚本中将变量foo视作数字5。

变量与数字本身的表现是一样的，所以你可以采用对数字所采用的古典数学符号的方式，诸如“+”、“-”、“\*”和“/”来分别表示加、减、乘和除。在下例中使用加号(+)来将两个foo加到一起(foo + foo)。

```
var foo = 5;
alert(foo + foo); // This will alert "10".
```

### 字符串

变量的另一种类型是字符串，其基本上是一连串文本。在一组单引号或双引号中的字符表明它是一个字符串，如下所示：

```
var foo = "five";
alert(foo); // This will alert "five".
```

变量foo现在被视作单词five。这适用于任何字符组合：字母、数字、空白符等。如果给一个值附上引号，那么可以把它视作文本字符串。如果对数字5附上引号，然后赋值，那么变量就不会表现为数值，而是成为包含字符5的文本字符串。

以前我们见到的加号(+)用于数字相加。当加号用于字符时，可将字符串连接为一个长字符串。如下例所示：

```
var foo = "bye"
alert (foo + foo); // This will alert "byebye".
```

在下例中需要注意的是，当用引号来定义5时，意味着将其视作一个字符串而不是数字。

```
var foo = "5";
alert(foo + foo); // This will alert "55".
```

如果将一个字符串和一个数字连接，那么JavaScript会假定该数字应该视作一个字符串，因为数学中不会发生该情况。

```
var foo = "five";
var bar = 5;
alert(foo + bar); // This will alert "five5".
```

### 布尔运算

我们可以对变量赋值“true”或者“false”，这称为布尔运算值，同时也是各种高级逻辑的关键。布尔运算值将“true”和“false”关键词融入JavaScript，因而引号是不需要的。

```
var foo = true; // The variable "foo" is now true.
```

如同数字一样，如果在数值上加上引号，那么将保存单词“true”而不是固有值true（也就是非false）。

某种意义上而言，JavaScript的所有内容要么是固有值“true”要么是“false”。无效的、不明确的、0以及空字符串（“”）都是固有值“false”，而其他值是固有值“true”。这些值尽管与布尔值“true”和“false”不完全相同，但是通常称为“truthy”和“falsy”。我可以保证，真的是这样。

## 数组

一个数组是一群数值的集合，这些数值能分配给单独的变量。数组中的数值被索引，意味着你可以依据它们在列表中的顺序号来引用它们。第一个数字索引号是0，第二个索引号是1，以此类推，这就是为什么总是从零开始，因为这就是JavaScript以及其他编程语言计数的方式。记住这一点可以避免将来的许多麻烦。

我们的脚本需要之前所定义的所有变量。我们可以定义它们三次并且用foo1、foo2等命名，或者将其置于一个用方括号（[]）标注的数组。

```
var foo = [5, "five", "5"];
```

当你需要使用这些值时，可以通过引用他们的索引号来从单一的foo数组中找到它们。

```
alert(foo[0]); // Alerts "5"
alert(foo[1]); // Alerts "five"
alert(foo[2]); // Also alerts "5"
```

## 比较运算符

既然知晓了如何将值赋予变量和数组，下一步就是了解如何比较这些值。以下有一些称为比较运算符的特殊符号，它们以不同的方式评估和比较赋值。

==	等于
!=	不等于
===	等同于
!==	不等同于
>	大于
>=	大于等于
<	小于
<=	小于等于

以下是将这些定义视为声明部分的原因。在比较值中，我们强调，目标



包含着固有值true或是固有值false的结果。当比较两个赋值时，JavaScript会对表述进行评价同时会给予一个基于该语句是正确还是错误的布尔值。

```
alert(5 == 5); // This will alert "true"
alert(5 != 6); // This will alert "true"
alert(5 < 1); // This will alert "false"
```

## 等于与等同于

比较难理解的部分就是“equal to”（==）与“identical to”（===）的区别。我们知道所有这些值都属于某个数据类型。举例而言，一个字符串5与数字5是类似的，但并不等同。

```
alert("5" == 5); // This will alert "true". They're both "5".
alert("5" === 5); // This will alert "false". They're both "5", but
they're not the same data type.
alert("5" !== 5); // This will alert "true", since They're not the
same data type
```

---

**警告：** 注意不要使用一个单等号，否则你会将第一个变量的值赋予第二个变量。

---

即使你不得不多次阅读，但理解之前的句子意味着你已经开始满足成为一个程序员所需的特殊要求。欢迎加入。

## 数学运算符

运算符的另一种类型就是数学运算符，它对数值进行数学运算。我们用简单的数学运算符，诸如加（+）、减（-）、乘（\*）和除（/）。以下是一些你应该知道的有用的快捷符：

+=	把值加到自己
++	增大数字的值（或者是包含数值的变量），每次增加1
--	减小数字的值（或者是包含数值的变量），每次减小1

## if/else语句

if/else语句就是JavaScript用来获取问题真假的方式。它们或多或少为那些JavaScript中的高级逻辑提供基础，它们的难易程度等同于编程。事实上，它们几乎都是用浅显的语言来描述的。条件语言的结构如下所示：

```
if(true) {
 // Do something.
}
```

它向浏览器发出指令，“如果条件满足，就执行花括号（{ }）中的命

令”。记住，代码中的空格对JavaScript没有影响，因此（true）两边的空格纯粹是为了方便代码阅读。

这里是之前所阐述的使用数列的例子。

```
var foo = [5, "five", "5"];

if(foo[1] === "five") {
 alert("This is the word five, written in plain English.");
}
```

由于我们正在进行比较，JavaScript将向我们提供“true”或是“false”的值。高亮显示的编码行打破了有关“true or false”的说法，索引号为1的foo变量的值与单词“five”是相同的。

在这种情况下，索引号为1的foo变量与“five”是等同的，因此alert会执行。

可以通过使用比较运算符不等于（!=），来核查是否存在错误。

```
if(1 != 2) {
 alert("If you're not seeing this, we have bigger problems than
 JavaScript.");
 // 1 is never equal to 2, so we should always see this alert.
}
```

我数学并不是很好，但据我所知，1永远不会等于2。JavaScript会这样解释：1不等于2是正确的描述，因此将运行该代码。

如果该描述没有评为“true”，那么花括号内的编码将被完全略过：

```
if(1 == 2) {
 alert("If you're seeing this, we have bigger problems than
 JavaScript.");
 //1 is not equal to 2, so this code will never run.
}
```

它包含了“if”，但“else”呢？

最后，当某些事正确时，我们做一些工作，但是这件事不正确时，我们需要做另外的工作，不正确的情况下该怎么做呢？可以写两个条件语句，但这稍微有些笨。可以使用“else, do something...else”。

```
var test = "testing";
if(test == "testing") {
 alert("You haven't changed anything.");
} else {
 alert("You've changed something!");
}
```

将该testing变量的值变为除了testing以外的内容，这将引出“You've

### 常用的JavaScript

在JavaScript中为了编写代码而创建了一个样式指南。文档“Principles of Writing Consistent, Idiomatic JavaScript”描述了以下内容：所有编码基础上的代码不管实际情况如何都应该显示个人操作。为了达到以上目标，一群开发者已经写成了一本常用样式说明，描述了空白符、换行、引号、函数、变量以及其他许多内容应该怎样编写才能达到“美丽编码”的目标。更多信息详见[github.com/rwldrn/idiomatic.js/](https://github.com/rwldrn/idiomatic.js/)。



## 练习19-1

## 将英语转化为JavaScript语言

在这个快速练习中，通过将英语陈述转化为JavaScript代码，你可以对变量、数组以及其他的一些内容有进一步的了解。你可以在附录A中找到答案。

1. 创建一个“friends”变量，并使其成为四个朋友的名字数组。
2. 显示“friends”目录中第三个名字的对话框。
3. 创建一个变量“name”同时以你的名字进行赋值。
4. 如果“name”的赋值与“Jennifer”的赋值一样，那么就会显示同名的对话框。
5. 创建一个名为“myVariable”的变量，以1~10之间的数值对其赋值。如果值大于5，就会显示“upper”的对话框。如果不是，就会显示“lower”的对话框。

changed something!”的警示。

## 循环

在某些情况下，我们会完成数组中的每一项并且进行一些处理，但是我们并不会写出整个项目列表并且重复十几次或更多。接下来你会学习称为循环的强大技术。

我知道，也许循环不太能吸引人眼球，但它们却是很有用的。用之前我们所涉猎的内容就能够很好地处理一些单独的变量，但仅限于此。循环能使我们处理大量的数据更容易。

现在有一个所有字段都留空的表单。如果使用DOM来获取页面上的所有文本输入，那么DOM会产生一个有所有文本输入元素的数组。（在第20章中将进行更多的讲解），当然，也可以在同一个时间对数组中的每一个值进行检查，但是那将是一个有许多代码，而且维持起来非常可怕的工作。如果使用循环来核对每一个值，那么就不需要改变脚本。不管页面中添加或删除了多少字段，不管数组的大小如何，循环总能使我们数组的每一项进行处理。

编写循环指令有很多方法，但for方式是最受欢迎的。以下是一个for循环的基本结构：

```
for(initialize the variable; test the condition; alert the value;)
{
 // do something
}
```

以下是for循环的具体例子：

```
for(var i = 0; i <= 2; i++) {
 alert(i); // This loop will trigger three alerts, reading "0", "1",
 // and "2" respectively.
}
```

这有一点复杂，让我们一部分一部分来：

## for()

首先，我们需要for语句，它构成了JavaScript。它表述为：如果条件每次都是对的，那就继续做下去。接下来会详细介绍这个语句。

```
var i = 0;
```

这个创建了一个值为零的新变量i。你可以通过单等号来描述。通常你不会见到用字母i来作为变量名称的代码，但是记住你可以使用任何变量名称。这是一个常见的惯例而已，并不是什么规定。

可以将其初始值设为零，因为习惯于从零开始计数，JavaScript也是

这么做的。

```
1 <=2;
```

只要`i<=2`，我们就可以继续循环。因为从零开始计数，这就意味着循环会持续三次。

```
i++
```

最后，`i++`是对`i`值加1的简便写法（`++`是之前介绍过的一种数学快捷操作符）。如果没有这步，那么`i`始终等于零，同时循环会无止境地继续下去。幸运的是，现代浏览器足够智能，不会发生此类事情。如果这三部分有一部分缺失，那么循环就不能进行。

```
{ script }
```

每循环一次，花括号中的内容就会运行一次，这种情况下循环会进行3次。如同在下面介绍的，`i`变量在循环代码操作中是可用的。

回到“在数组中核查每个项”的例子，应该怎样编写循环项呢？

```
var items = ["foo", "bar", "baz"]; // First we create an array.
for(var i = 0; i <= items.length; i++) {
 alert(items[i]); // This will alert each item in the array.
}
```

这个例子有两个地方与第一个循环不同。

`Items.length`

为了避免使用数字来限制循环的次数，可以用JavaScript中的属性来确定数组的“长度”，即它所包含的项数量。`.length`仅仅是标准属性以及JavaScript中数组方法的一种。

`items[i]`

记住我所提到的，我们可以使用循环中的`i`变量。可以通过它来表示数组的索引。从零开始是正确的，如果将值设为1，那么数组中的第一项就会被遗漏。

现在不管数组是多大，循环执行的次数与数组中的项的个数相同，同时会对数组中的每一项进行引用。

有几十个编写循环的方式，但这是更为普遍的一种方式。开发者使用循环来完成许多任务，例如：

- 通过页面的一系列元素来实现循环，并且核对每个变量值，对每个变量进行样式处理，或者添加/去掉/改变变量的属性。例如，我们会通过每一个元素来实现循环，并且确保用户在继续之前已经得出变量的有效值。
- 创建一个新数组，与有确定值的原有数组包含类型相同。在循环中



核查原有数组中每一项的值，如果该值与我们所寻找的值相匹配，那么我们就将这些项置于新数组中。这勉强称为循环过滤器。

## 函数

我已经潜移默化地介绍了一些函数。以下是一个你也许认识的函数例子：

```
alert("I've been a function all along!");
```

一个函数是一些除非被引用否则不会运行的代码。`alert()`是浏览器中内置的一个函数。这是只有我们明确发出指令时才会运行的代码块。某种程度上说，可以将函数视作包含逻辑的变量，在逻辑中，变量会运行逻辑所包含的所有代码。

所有的变量都使用同一种模式（图19-6）。函数名后面总是跟着一组括号（没有空格）然后是包含相关代码的花括号。这些括号有时会包含附加信息，这些函数所使用的信息称作参数。参数是能够影响函数作用方式的数据。例如，我们所熟知的`alter`函数将一串文本视作一个参数，同时使用该信息来形成结果对话框。

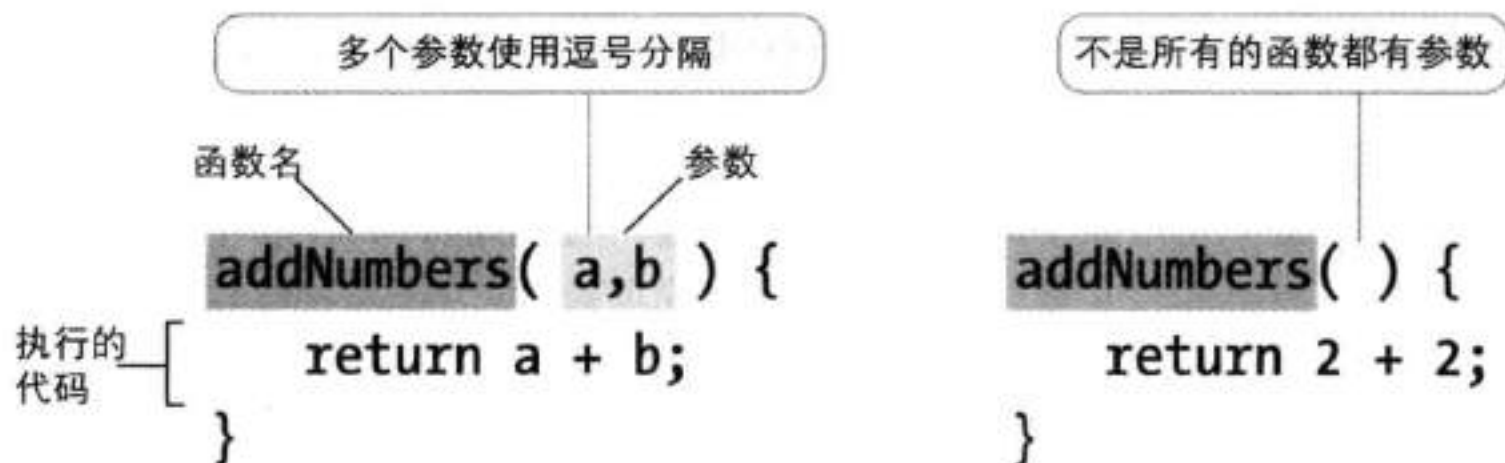


图19-6：函数的结构

这里有两种类型的函数：JavaScript的固有（原生）函数和自定义函数。让我们来具体看下这两种函数。

### 固有（原生）函数

JavaScript中有数百个已经定义的函数，包括：

`alert()`, `confirm()`, and `prompt()`

这些函数会产生浏览器级别对话框。

`Date()`

返回当前的日期和时间。

`parseInt("123")`

这个函数把一个包含数字的字符串数据类型变为一个数字数据类型

型。字符串作为参数传递给函数。

```
setTimeout(functionName, 5000)
```

它将在以后运行一个函数。在第一个参数中指定函数，在第二个参数中以毫秒为单位指定时间。（5000毫秒等于5秒。）

固有函数远不止这些。

## 自定义函数

创建一个自定义函数，需要写出函数关键词，这些关键词之后有函数名称、开闭括号以及开闭花括号。

```
function name() {
 // Our function code goes here.
}
```

如同变量和数组，函数的名称可以随意命名，但必须使用相同的语法规则。

如果要创建一个警示文本的函数（我知道，这个有些麻烦），那么其形式如下所示：

```
function foo() {
 alert("Our function just ran!");
 //This code won't run until we call the function 'fool()'
}
```

通过编写以下代码，可以在脚本中调用函数，并运行函数中的代码。

```
foo(); // Alerts "Our function just ran!"
```

可以通过代码来数次调用函数，这节约了很多时间，也省得重复编写代码。

## 参数

拥有一个通过脚本来运行相同代码的函数并不可能总是有用的。可以“传递参数”——向原生的以及自定义的函数提供数据，以在不同的时间对不同的数据运用函数逻辑。

---

参数是函数运行时所使用的一个值或一个数。

---

为了给参数留有一席之地，可以在定义函数时在括号中添加一个或多个以逗号分隔的变量。然后，当调用这个函数时，括号中所包含的任何东西将作为运行变量传递给函数。这听起来有点混乱，但在具体操作中并没有这么糟糕。

例如，想要创建一个非常简单的函数，该函数涉及数组中所包含的项。那么可使用`.length`来获取数组的长度，所以只需要将数组传递给函数。



可将数组作为一个参数。为了达到这个目的，在自定义函数时在括号中指定变量名称。函数中的变量是有效的，同时会包含调用函数时所传递的参数。

```
function alertArraySize(arr) {
 alert(arr.length);
}
```

现在当调用函数时，括号中的数组将被传递到参数名为`arr`的函数中。我们所要做的就是获取其长度。

```
var test = [1,2,3,4,5];
alertArraySize(test); // Alerts "5"
```

## 返回值

这个部分特别乱，但非常的有用。

这是很常见的使用函数来计算，然后得出一个可以在脚本中使用的值的方法。可以通过多种巧妙的方式来应用该方法，但这里有更为简单的方法。

函数中的`return`关键词能有效地将该函数转化为一个有动态值的变量。这是个更适合展示的方法，所以当我们考虑下列例子时，请耐心等待。

```
function addNumbers(a,b) {
 return a + b;
}
```

现在有一个含有两个参数并且将它们加总的函数。如果结果一直存在于函数中，那么这就没有多大用处，因为该结果不能在脚本中使用。这里使用`return`关键词将结果从函数中取出。现在任何有关函数的引用都能得出函数的结果——如同变量那样。

```
alert(addNumbers(2,5)); // Alerts "7"
```

某种意义上，`addNumbers`函数是一个包含动态值的变量：该值是我们计算的。如果没有在函数中得到值，那么先前的脚本就会警示“`undefined`”，如同没有赋值的变量一样。

`return`关键词有一个特点。一旦JavaScript得到返回值，那么函数也就结束了。看看下列内容：

```
function bar() {
 return 3;
 alert("We'll never see this alert.");
}
```

当你调用使用`bar()`的函数时，第二行的警示就永远不会运行。一旦到了返回值，那么函数就会结束。

## 变量范围和var关键词

有时，你会想要在一个函数中定义变量，该变量通过脚本在任何地方都是可用的。有时你也许会想限定它，使它只有在其存在的函数中才是可用的。变量的可用性称为变量的范围。页面中能被任何脚本使用的变量称为全局变量，那些只有在本函数中使用的变量称为局部变量。

JavaScript变量使用函数来界定其范围。如果一个变量定义于函数之外，那么它就是全局变量并且对所有的脚本都是可用的。当你在函数中定义变量，并且希望它是局部变量时，你可以通过在变量名称前添加`var`关键词来达到将其标记为局部变量的目的。

```
var foo = "value";
```

为了将局部变量转化为全局变量，需要忽略`var`关键词，并且简单定义该变量：

```
foo = "value";
```

你需要注意定义局部变量的方式，否则会得出意料之外的结果。看看以下JavaScript代码片段：

```
function double(num){
 total = num + num;
 return total;
}
var total = 10;
var number = double(20);
alert(total); // Alerts 40.
```

因为你明确地对变量`total`赋值10，所以希望你脚本末尾的`alert(total)`函数能得出值10。但是由于我们并没有说明函数中`total`变量为`var`，它将是全局变量。因此，尽管变量`total`的值设为10，但之后的语句将运行该函数并且运用所定义的`total`变量值。如果没有`var`关键词，变量就会“被漏出”。

正如你所见的，全局变量的问题是，在页面中可以在所有的脚本中共享。全局变量中变量越多，那就越有可能陷入一种“碰撞”，即全局变量名与你所设定的相同。这可能会导致变量无意中被重新赋值，而这又会导致脚本出错。

记住，不可能总是控制页面中运行的所有代码。页面中包含的代码由第三方编写是很常见的现象，例如：

### 保持变量的局部性

如果想确保所有变量都是局部变量，那么可将JavaScript的所有内容置于以下形式中：

```
<script>
(function() {
 //All your code here!
})();
</script>
```

这个小的封闭方案称为IIFE（独立调用的函数表达式），同时将该方法归功于Ben Alman。



- 脚本呈现广告内容
- 用户追踪和分析脚本
- 社会媒体的“共享”按钮

最好不要产生变量“碰撞”，所以当开始编写脚本时，要随时将变量界定为局部变量（参见侧栏）。

这就是关于JavaScript的简短介绍。当然，要学的还有很多，但以上内容应该能给你的自我学习提供一个良好的基础，并且当你碰到它们时能阐明脚本的含义。在看一些例子之前，我们需要处理一些与JavaScript相关的属性。

## 浏览器对象

除了能在网页中控制元素外，JavaScript也可以实现对浏览器窗口本身的访问和操作。例如,你可能想要获得或替换浏览器地址栏中的URL，又或者想要打开或关闭浏览器窗口。

在JavaScript中，浏览器是window对象。窗口对象有若干可以用来与之交互的属性和方法。事实上，`alter()`是一个标准的浏览器对象方法。表19-1列明了一些属性和方法，它们能用来与window对象一起向你展示什么是可能的。

表19-1：浏览器属性和方法

属性/方法	描述
event	代表事件的状态
history	用户已经在浏览器窗口中浏览的URL
location	对地址栏中的URL提供读/写的权限
status	设置或返回窗口状态栏的文本
alert()	显示一个有详细信息的警示窗口和一个OK按钮
close()	关闭当前窗口
confirm()	显示一个有详细信息的对话框、OK按钮以及一个取消按钮
focus()	在当前窗口设置焦点

## 事件

JavaScript能在页面以及浏览器窗口存取对象，但是你知道它也在“侦听”某些事件的发生吗？一个事件是能够用JavaScript检测的功能，比如，当文档加载时或用户单击一个元素时，或是仅仅移动了鼠标。HTML 4.0使得脚本与页面中的事件绑定，这些事件可以是由用户、浏览器本身或者其他脚本引起的。这个称作事件绑定。

**注意：**事件句柄会“侦听”某个文档，浏览器或是用户操作，并且将脚本绑定到这些行为上。

在脚本中，事件是由一个事件句柄来识别的。例如，当一个文档加载时，`onload`事件句柄就会触发一个脚本，同时当用户单击或用鼠标移动一个元素时，这个`onclick`和`onmouseover`句柄也会触发一个脚本。表19-2列出了一些最为常用的事件句柄。

表19-2：常用事件

事件句柄	项描述
<code>Onblur</code>	失去焦点的元素
<code>Onchange</code>	表单字段的内容被更改
<code>OnClick</code>	鼠标单击一个对象
<code>Onerror</code>	当文档或图像加载时产生的错误
<code>Onfocus</code>	元素获得焦点
<code>Onkeydown</code>	按下键盘上的一个键
<code>Onkeypress</code>	按下或按着键
<code>Onkeyup</code>	松开按键
<code>Onload</code>	图像或页面完成加载
<code>Onmousedown</code>	按鼠标按钮
<code>Onmousemove</code>	移动鼠标
<code>Onmouseout</code>	从元素上移开鼠标
<code>Onmouseover</code>	鼠标经过元素
<code>Onmouseup</code>	放开鼠标按钮
<code>Onsubmit</code>	在表格中单击提交按钮

以下有三种常用的运用事件句柄的方法：

- 作为一个HTML属性
- 作为一个附加给元素的方法
- 使用`addEventListener`

在后两种方法的实例中，将使用`window`对象。所有附加到`window`的事件都应用于整个文档。也将使用`onclick`事件。

## 作为一个HTML属性

你可以设定函数在标记的属性中运行，如下例所示：

```
<body onclick="myFunction();"> /* myFunction will now run when the user
clicks anything within 'body' */
```



虽然依旧是函数形式的，但这却是一个在页面中将事件附加到元素的老方法。与避免在标记中使用`style`属性来对单个元素应用样式的原因相同，应该避免使用该方法。如果使用该方法，那么页面中语义层和行为层间的界限会变得模糊，可能导致难以维护。

## 作为一个方法

这是另一个附加事件的有些过时的方法，虽然它在脚本中运行很严格。我们还可以使用JavaScript内置的助手来附加函数。

```
window.onclick = myFunction; /* myFunction will run when the users
click anything within the browser window*/
```

可以使用一个匿名函数而不是预定义函数：

```
window.onclick = function() {
 /* Any code placed here will run when the users clicks anything
 within the browsers window */
};
```

这个方法不仅简单而且易于维护，但也有一个相当大的缺点：每次只能绑定一个事件。

```
window.onclick = myFunction;
window.onclick = myOtherFunction;
```

在所展示的例子中，第二个绑定覆盖第一个，所以当用户在浏览器窗口单击时，只有`myOtherFunction`运行。有关`myFunction`的引用被淘汰了。

## addEventListener

虽然第一眼看上去有点复杂，但这个方法可以保持脚本中的逻辑，并且可以运行单一对象的多个绑定。这个语法有点烦琐。从调用目标对象的`addEventListener`方法开始，然后在疑问中设定事件以及作为两个参数运行的函数。

```
window.addEventListener("click", myFunction);
```

注意，在这个语法中，忽略了事件句柄中的“on”。

如同之前的方法，`addEventListener`也能在匿名函数下使用。

```
window.addEventListener("click", function(e) {
});
```

---

注意：更多`addEventListener`内容，详见Mozilla Developer Network的“`element.addEventListener`”页面（[developer.mozilla.org/en/DOM/element.addEventListener](http://developer.mozilla.org/en/DOM/element.addEventListener)）。

---

## 综合应用

现在你已经对JavaScript的重要模块构建有了一些了解。你已经了解了变量、数据类型以及数组。你也知晓了if/else语句、循环以及函数。你可以从事件句柄了解浏览器。总之就是许多零零碎碎的东西。下面通过一些简答的脚本例子来了解它们的组合方式。

### 例一：有关两个参数

这里有一个包含两个参数，并且可以得出两个值中较大值的简单函数。

```
greatestOfTwo(first, second) {
 if(first > second) {
 return first;
 } else {
 return second;
 }
}
```

首先命名函数为“greatestOfTwo”。将其设为两参数的函数，这两个参数称为“第一个参数”和“第二个参数”。这个函数包含一个if/else语句，即当第一个参数大于第二个参数时显示“第一个参数”，否则显示“第二个参数”。

### 例二：最长的词

以下是一个含有单参数的字符串数组，并且得出数组中最长字符串的函数。

```
longestWord(strings) {
 var longest = strings[0];

 for(i = 1; i < strings.length; i++) {
 if (strings[i].length > longest.length) {
 longest = strings[i];
 }
 }
 return longest;
}
```

首先，命名函数并且让其含有一个参数。然后将longest变量值设为数组中第一项的值：strings[0]。循环从1而不是0开始，因为已经在数组中得到了第一个值。每一次遍历循环，都会将数组中当前项的长度与longest变量值的长度进行比较。如果数组中当前项包含更多的字符，那么就会将longest变量的长度值改变为当前项的长度。如果不是这样，那么就什么都不做。当循环结束时，就能得出longest变量的值，该变量包含数组中的最长字符串。



## 练习19-2 你来试试看

在此练习中你将编写脚本,该脚本用“new messages”来更新浏览器窗口中的页面标题。你可能偶尔遇到过这类脚本。对于该练习,我们假定将来某一天,这个练习将成为一个更大网页应用程序的一部分,我们只负责使用“unread messages”来更新页面标题的内容。

我已经创建了一个文档 (*title.html*), 可以从 *learningwebdesign.com* 上本章的 *materials* 文件夹中获取。

1. 首先在浏览器中打开 *title.html*, 你会看到一个标题标记已被填写的空白页。如果你查看浏览器窗口的顶端, 你会注意到它写着“Million Dollar WebApp”。
2. 现在在文本编辑器中打开文档。你会发现在封闭 `</body>` 标签之前, 有一个包含注释的 `script` 元素。你可以随时删除注释。
3. 如果要改变页面标题, 那么首先应保存原始的标题。创建一个名为 `originalTitle` 的变量。对于变量值, 可以通过使用 `document.title` DOM 方法来使浏览器获得文档标题。在页面加载时, 我们有页面标题的保存变量。该变量应该是全局变量, 所以不属于任何函数。

```
var originalTitle = document.title;
```

4. 接下来将定义一个函数, 这样就可以在以后需要的时候使用该脚本。给函数起一个容易记住的名称, 以便在之后的代码中遇到时, 一眼就能知道它的含义。我称它为“`showUnreadCount`”, 当然你也可以随便起个名字。

```
var originalTitle = document.title;
```

```
function showUnreadCount() {
}
```

5. 我们需要思考函数需要什么, 才能使函数有用。对一些未读信息, 这个函数完成了一些工作, 所以它的参数是一个在本例中称为“unread”的数字。

```
var originalTitle = document.title;
```

```
function showUnreadCount(unread) {
}
```

6. 现在添加一些函数所需的代码。我们需要页面文档的标题中加上未读信息的个数。听起来像

是个增加的工作 (+!), 这里将 `document.title` 设为 (=) 已经保存的 `originalTitle`, 再加上 `showUnreadCount` 中的数字。正如之前所学的那样, JavaScript 将一个字符串与一个数字连接起来, 虽然它们并不全是字符串。

```
var originalTitle = document.title;
```

```
function showUnreadCount(unread) {
 document.title = originalTitle + unread;
}
```

7. 在继续之前, 试用脚本。在定义的函数以及 `originalTitle` 值下面, 键入 `showUnreadCount(3)`, 现在保存页面, 并且在你的浏览器中查看 (图 19-7)。

```
var originalTitle = document.title;
```

```
function showUnreadCount(unread) {
 document.title = originalTitle + unread;
}
showUnreadCount(3);
```



图19-7: 标题标签已经改变, 虽然并不是很快

8. 我们的脚本正在运行, 但并不是很容易浏览。幸运的是, 对于我们连接的字符串并没有数量的限制。现在添加额外的字符串, 该字符串包括计数值以及括号中的“new messages!”字样 (图 19-8)。

```
var originalTitle = document.title;
```

```
function showUnreadCount(unread) {
 document.title = originalTitle + " (" + unread
 + " new messages!) ";
}
showUnreadCount(3);
```



图19-8: 看起来更好一些

## 自我测验

本章中涵盖了许多新内容。这里有一个测试，看看你理解了多少。

1. 说明链接外部.js文件的一个好处和一个坏处。

2. 有下列数组：

```
var myArray = [1, "two", 3, "4"]
```

对于下面这些例子，写出每个例子会提示什么信息：

- a. `alert( myArray[0] );`
- b. `alert( myArray[0] + myArray[1] );`
- c. `alert( myArray[2] + myArray[3] );`
- d. `alert( myArray[2] - myArray[0] );`

3. 以下警示信息将提示什么？

- a. 

```
var foo = 5;
foo += 5;
alert(foo);
```
- b. 

```
var foo = 5;
alert(foo++);
```
- c. 

```
var foo = 2;
alert(foo + " " + "remaining");
```
- d. 

```
var foo = "Mat";
var bar = "Jennifer";
if(foo.length > bar.length) {
 alert(foo + " is longer.");
} else {
 alert(bar + " is longer.");
}
```
- e. `alert( 10 === "10" );`

4. 描述以下操作：

```
for(var i = 0; i <= items.length; i++) { }
```

5. 全局变量存在的问题是什么？

### 进一步阅读

这个激起了你更强的求知欲吗？你看到通过使用JavaScript所引起的变化了吗？当然，网上有关JavaScript的入门教程并不少。我也要推荐几本书（所有这些都是O'Reilly Media出版的）：

- Shelley Powers的《Learning JavaScript》
- David Sawyer McFarland的《JavaScript & jQuery: The Missing Manual》
- Douglas Crockford的《JavaScript: The Good Parts》



6. 将事件句柄与所发生的事情进行匹配。
- |                |                   |
|----------------|-------------------|
| a. onload      | 1. 用户完成表格并且单击提交按钮 |
| b. onchange    | 2. 页面完成加载         |
| c. onfocus     | 3. 指针悬停在一个链接上     |
| d. onmouseover | 4. 选择文本字段并且输入     |
| e. onsubmit    | 5. 用户在表单字段修改名字    |

# 第20章 使用JavaScript

现在，你对JavaScript语言已经有一些感觉，让我们来看一些可以把它应用于现代网页设计中的方法。首先，我们将探索DOM脚本，这使得我们能够操纵这些元素、属性和网页上的文字。我将向你介绍一些现成的JavaScript和DOM脚本资源，所以你不必去管它。你还将了解一些关于“polyfills”的东西，它可以给旧版本的浏览器提供具有现代特征和标准化的功能。我还将介绍一些JavaScript库，它们可以与polyfill和快捷方式一起完成常见的任务，从而使开发人员的生活变得更方便。

## 遇见DOM

这本书你已经多次看到文档对象模型（简称DOM），但现在我们应该重视它了。DOM为我们提供了一种访问和操作一个文件的内容的方法。我们通常使用HTML，但是DOM也可以使用任何XML语言来完成。虽然我们更加关注JavaScript的关系，但有一点是值得注意的，DOM可以被其他语言访问，如PHP、Ruby、Python、C++、Java、Perl以及其他更多语言。虽然DOM Level1是在1998年W3C发布的，DOM脚本开始流行却是在将近5年之后。

DOM是一种HTML和XML页面的编程接口（API）。它提供了一种文档的结构化地图，还有一组方法，以便与所含元素交互。实际上，它是把我们的标记方式转换成JavaScript（和其他语言）可以理解的格式。我知道这听起来很枯燥，但基本意思是DOM就像页面上所有元素的一个地图。我们可以使用它通过名字和属性来找到元素，然后添加、修改或删除元素及其内容。

没有DOM，JavaScript将无法理解文件的内容，我指的是整个文件的内容。从页面的doctype到文本中的每个文字，都可以通过DOM来访问并通过JavaScript操作。

### 本章内容

使用DOM访问并改变元素、  
属性和内容  
使用polyfills实现浏览器版本兼容  
使用JavaScript库  
简单介绍Ajax

---

DOM为我们提供了一种访问和操作一个文件的内容的方法。

---



快速浏览

DOM是一个节点的集合：

- 元素节点
- 属性节点
- 文本节点

节点树

可以简单地把DOM看做文档树（如图20-1所示）。当你学习CSS选择器时，你看到的文件都是以这种方式绘图的。

```
<html>
<head>
 <title>Document title</title>
 <meta charset="utf-8">
</head>
<body>
 <div>
 <h2>Subhead</h2>
 <p>Paragraph text with a link here.</p>
 </div>
 <div>
 <p>More text here.</p>
 </div>
</body>
</html>
```

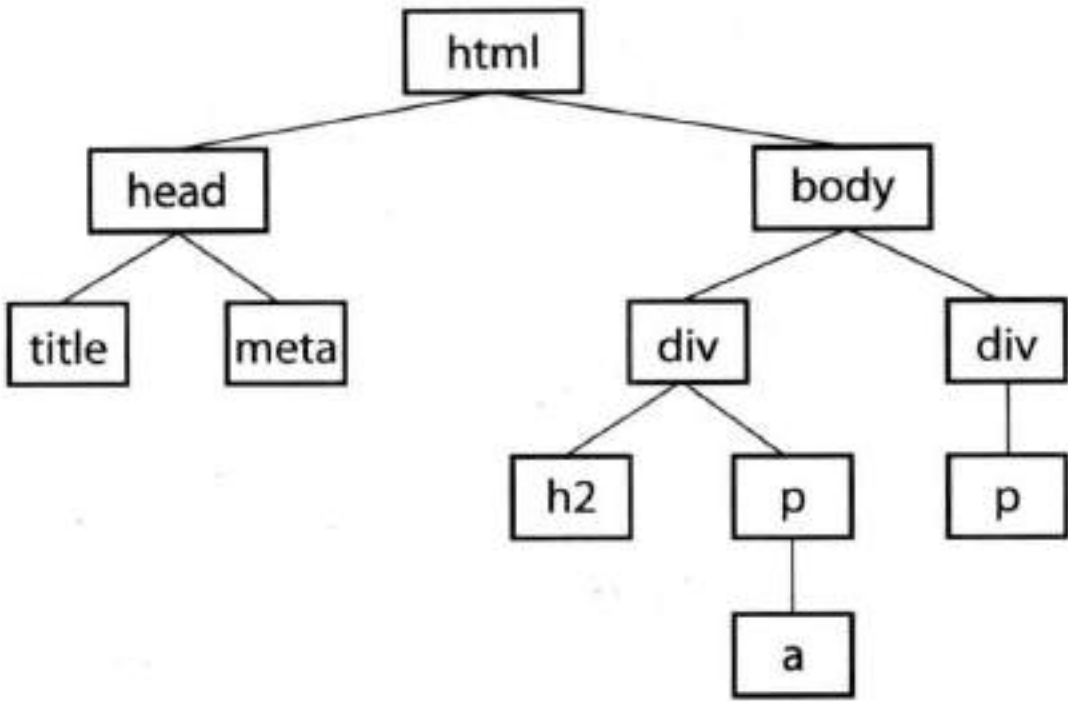
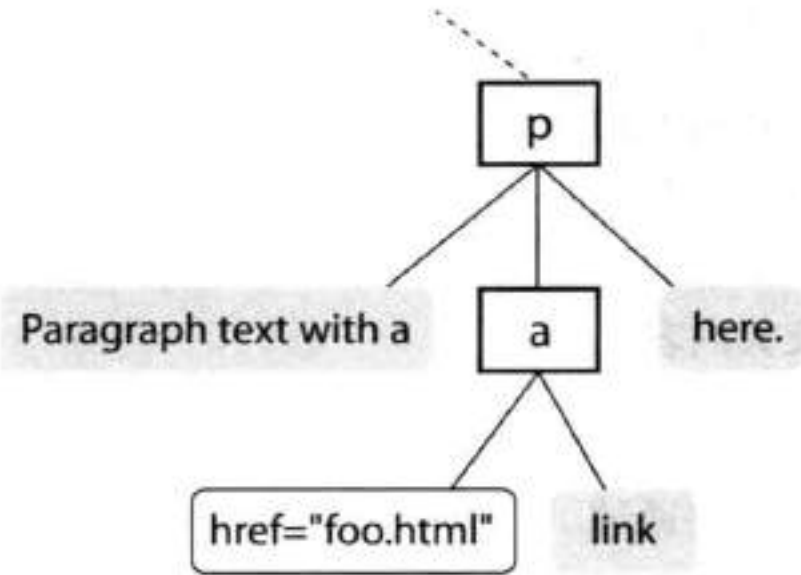


图20-2：在样例文档中，第一个P元素的节点

图20-1：简单的文档树

```
<p>Paragraph text with a link here.</p>
```



页面内的每个元素都称为一个节点。如果你把DOM当做一棵树，每个节点都有一个分支，并且每个分支又含有各自的小分支。但DOM允许深入访问的内容比CSS多，因为它把实际的内容也作为一个节点。图20-2显示结构中的第一个元素。元素、属性和内容是DOM节点树的所有节点。

它还提供了一套标准化的方法和功能，可以在网页中与元素互动。大多数DOM脚本包括读取文档和写入文档。

有几种方法可以帮你在文档中找到你需要的东西。让我们看一些具体的方法，我们可以用

其来访问被DOM定义的对象（JS开发者称为“遍历DOM”或“抓取DOM”），以及一些操纵这些元素的方法。

## 访问DOM节点

DOM中的document对象标识网页本身，而且往往会作为DOM抓取的起点。document对象有一些标准的属性和方法来访问元素集合。这让人想起在第19章学到的length属性。就像length是所有数组的标准属性一样，document对象具有包含文档信息的一系列内置属性。通过使用这些属性连同方法（由句点分隔），就可以找到我们需要的元素，从而形成一种遍历文件的路径。

为了让你了解我的大致意思，下面例子中的声明是要在网页（document）中找到id值为“beginner”的元素，找到元素（innerHTML）中的HTML内容，并把这些内容保存到一个变量（foo）中。

```
var foo = document.getElementById("beginner").innerHTML;
```

因为属性方法链往往很长，所以每个属性或方法往往都换行，使它看起来更容易。记住，空白符在JavaScript中容易被忽略，但这不会影响语句的解析。

```
var foo = document
 .getElementById("beginner")
 .innerHTML;
```

有几种访问文档中的节点的方法。

## 通过元素名称

### getElementsByTagName ()

我们可以通过使用document.getElementsByTagName()和标签来访问单个元素。此方法可以检索参数指定的任何元素或多个元素。

例如，document.getElementsByTagName("p")返回页面的每一个段落，将其包含在所谓的集合collection或节点列表nodeList中，这些节点按从上到下的顺序排列。nodeList的行为就像是数组。为了访问nodeList中特定的段落，我们可以使用索引来引用，就像一个数组。

```
var paragraphs = document.getElementsByTagName("p");
```

基于此变量声明语句，paragraphs[0]是文档中第一个段落的引用，paragraphs[1]指第二个段落，以此类推。

---

**注意：**nodeLists生活收藏。如果你在nodeList循环中操纵文档（例如，循环遍历所有段落，并追加新的段落），你可以在一个无限循环中结束。

---



如果我们要分别访问nodeList中每个元素，一次一个……嗯，这就可以用到我们之前学过的查询数组方法了。循环可以很好地处理nodeList。

```
var paragraphs = document.getElementsByTagName("p");
for(var i = 0; i < paragraphs.length; i++) {
 //do something
}
```

现在你可以在循环中使用paragraphs[i]来访问页面中的每个段落，就像操作一个数组一样，只是数组的内容是元素而不是数值。

### 通过id属性值

#### getElementById()

此方法根据元素ID（id属性的值）来返回一个单独的元素，我们把ID作为参数传递给该方法。例如，为了访问特定的图像：

```

```

可以把id值作为参数传给getElementById()方法：

```
var photo = document.getElementById("lead-photo");
```

### 通过类属性值

#### getElementsByClassName()

这个方法就像它的名字一样，可以让你以class属性的值为基础来访问文档中的节点。下面的语句将class值为“column-a”的元素指派给变量firstColumn,这样它可以很容易在脚本中访问。

```
Var first Column = document.getElementsByClassName("column-a");
```

就像getElementsByTagName一样，会返回一个节点列表，我们可以通过循环一个一个地访问列表内容。

### 通过选择器

#### querySelectorAll()

querySelectorAll允许你访问CSS-style选择器代表的DOM节点。以下例子中的语法应该是我们比较熟悉的。它就像访问一个特定元素的子元素一样简单：

```
var sidebarPara = document.querySelectorAll(".sidebar p");
```

或像根据属性来选择一个元素一样复杂：

**警告：** 这是一个比较新的访问节点方法。虽然当前的现代浏览器支持getElementsByClassName(), 但它在IE 8或之前版本中将无法工作。

**警告：** 因为它是一种较新的方法，querySelectorAll()可在当前版本的浏览器下工作，但不可在IE 7或之前版本中工作。

```
var textInput = document.querySelectorAll("input[type='text']");
```

与`getElementsByTagName`、`getElementsByClassName`和`querySelectorAll`相同，它也可以返回一个`nodeList`（即使选择器只匹配一个元素）。

## 访问属性值

### `getAttribute()`

正如我前面提到的，元素不是唯一的可以通过DOM获得的东西。为了获得附加到一个元素节点的属性值，我们可以调用带有一个参数-属性名-的`getAttribute()`。假设有一个图像`source.jpg`，标记如下：

```

```

在下面的例子中，会访问特定图像（`getElementById`），并且在一个变量（`bigImage`）中保存对它的引用。在这一点上，可以在`getAttribute`方法中指定一个参数，就可以访问元素的任何属性（`alt`、`src`或`id`）。在这个例子中，得到`src`属性的值，并且将它作为`alert`的内容。（我不知道在真实设计中什么时候需要这样做，但在这里，它可以演示这种方法。）

```
var bigImage = document.getElementById("lead-image");
alert(bigImage.getAttribute("src")); // Alerts "stratocaster.jpg".
```

## 控制节点

一旦我们使用之前讨论的方法来访问节点，DOM会提供几个内置的方法来操纵这些元素，即它们的属性和它们的内容。

### `setAttribute()`

继续前面的例子，看看如何获取属性值，但如果我们想要设定`src`属性的值为一个新的路径，那么就使用`setAttribute()`！这种方法需要两个参数：属性和新的属性值。

在这个例子中，我们使用JavaScript，通过改变`src`属性的值来改变图像。

```
var bigImage = document.getElementById("lead-image");
bigImage.setAttribute("src", "lespaul.jpg");
```

想想看通过改变属性值，可以对一个文档做什么呢？在这里，我们更换了图像，但是这个方法同样可以用来更改整个文档：

- 根据用户在页面上的交互，更新复选框和单选按钮的`checked`属性



- 找到.css文件的link元素，并将href值转向一个不同的样式表，改变所有页面的样式
- 将title属性更新为元素状态的信息（例如，“这个元素是目前选定的”）

### innerHTML

innerHTML提供了一个简单地访问和改变文本和标记元素的方法。它的表现不同于目前已经介绍的方法。我们需要一个快速的方式，使用intro类来给页面的第一个元素增加一段文字：

```
var introDiv = document.getElementsByClassName("intro");
introDiv.innerHTML = "<p>This is our intro text</p>";
```

因为innerHTML告诉JavaScript将字符串“<p>”和“</p>”作为标记，所以第二个语句会将字符串的内容添加给introDiv（class值为“intro”的元素）。

### style

DOM还可以让你使用style属性来添加、修改或删除一个元素的样式。它就像使用内联style属性一样。单独的CSS属性可作为style属性的值。学习了CSS和DOM技术，你就能理解这些语句的意思：

```
document.getElementById("intro").style.color = "#fff";
document.getElementById("intro").style.backgroundColor = "#f58220";
//orange
```

在JavaScript和DOM中，CSS中有连字符的属性名（如background-color和background-top-width）都改为首字母大写的方式（比如backgroundColor和borderTopWidth），所以“-”号就不会当做操作符。

在这个你刚刚看到的例子中，style属性是用来设置节点的样式。它也可以用来获得一个样式值，可以用于脚本的其他地方。这一声明获取#intro元素的背景颜色，并将其指派给变量brandColor：

```
var brandColor = document.getElementById("intro").style.backgroundColor;
```

## 添加和移除元素

到目前为止介绍了在现有的文档中获取和设置节点的例子。DOM还允许开发者通过添加和删除节点来更改文档结构本身。开始时我们将创建一个新的节点，这是相当简单的，然后你将看到我们如何添加节点并创建

页面。在这里显示的方法比使用innerHTML添加内容更精彩。添加完成后，会删除节点。

### createElement()

使用createElement()的方法来创建一个新的元素。这个函数接受一个单一的参数：需要创建的元素。使用这种方法是有点违背常理的，因为新的元素一开始并没有出现在网页中。一旦我们以这种方式创建一个元素，除非我们把它添加到文档中，否则它只会停留在JavaScript中。把它当做一个新元素（也就是我们想象中的新元素）的引用，我们可以在JavaScript中操作这个元素，一旦我们准备好，就可将其添加到页面。

```
var newDiv = document.createElement("div");
```

### createTextNode()

如果要为已经创建的元素或者页面上的元素输入文字，那么可以调用createTextNode()方法。使用它时，需要提供一个文本字符串作为参数，该方法会创建一个支持DOM的文本，并且准备加入到页面中。就像createElement，这创造了一个新的文本节点的引用，我们可以在一个变量中存储这个节点，并在必要的时候将其添加到页面。

```
var ourText = document.createTextNode("This is our text.");
```

### appendChild()

在创造了一个新的元素和一个新的文本字符串后，该如何才能让它们成为文档的一部分呢？可以使用appendChild方法。这种方法需要一个单一的参数：需要添加到DOM的节点。你可以使用现有的元素来调用它，这个元素会成为文档结构中新元素的父节点。看看下面的例子。

在这里，在页面上有一个id为“our-div”的简单div。

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

假设要将包含文字“Hello,world”文本的段落添加到一个#our-div中。可以先创建p元素（document.createElement()）和包含内容的文本节点（createTextNode()），这个文本节点会放在p元素中。

```
var ourDiv = document.getElementById("our-div");
var newParagraph = document.createElement("p");
var copy = document.createTextNode("Hello, world!");
```

现在有了元素和一些文本，我们可以使用appendChild()将它们结合起来。



```
newParagraph.appendChild(copy);
ourDiv.appendChild(newParagraph);
```

第一个语句将copy（“Hello, World”文本节点）添加到我们创造的新段落（newParagraph），所以现在这个元素包含了内容。第二个语句追加新段落到原来的div中（ourdiv）。现在ourdiv不是DOM中空占其位却毫无内容的东西，它将在网页上显示内容“Hello, World”。

你应该理解它的工作方法了。接下来再看看其他的方法。

### insertBefore()

该insertBefore()方法，你应该能猜到，它会在一个元素之前插入另一个元素。它需要两个参数：第一个是需要插入的节点，第二是需要在哪一个元素前面插。你还需要知道被添加元素的父元素。

例如，在段落标记前插入一个新标题：

```
<div id="our-div">
 <p id="our-paragraph">Our paragraph text</p>
</div>
```

我们通过把变量名指派给div和它所包含的p元素，然后创建h1元素及其文本节点，并把它们放在一起，就像我们看到的最后一个例子。

```
var ourDiv = document.getElementById("our-div");
var para = document.getElementById("our-paragraph");

var newHeading = document.createElement("h1");
var headingText = document.createTextNode("A new heading");
newHeading.appendChild(headingText);
// Add our new text node to the new heading
```

在例子中最后一个语句中，insertBefore()方法将para元素之前的newHeading h1元素放在ourdiv中。

```
ourDiv.insertBefore(newHeading, para);
```

### replaceChild()

replaceChild()方法用一个节点取代另一个节点，它带着两个参数。第一个参数是新的子节点（即你要替换成的节点）。第二个是被第一个节点所取代的节点。与insertBefore()一样，你还需要确定取代发生地方的父元素。为了简单，用以下标记开始：

```
<div id="our-div">
 <div id="swap-me"></div>
</div>
```

我们想把id为“swap-me”的div替换为一个图像。首先，创建一个新的

img元素，并且把src属性设置为图像文件的路径。在最后的语句中，使用replaceChild()来用newImg元素来替换swapMe。

```
var ourDiv = document.getElementById("our-div");
var swapMe = document.getElementById("swap-me");
var newImg = document.createElement("img");
//Create a new image element

newImg.setAttribute("src", "path/to/image.jpg");
// Give the new image a "src" attribute
ourDiv.replaceChild(newImg, swapMe);
```

## removeChild()

我向我的母亲解释道：“我们可以将元素带进这个世界，同样可以把它移出去。”你可以使用removeChild()方法将一个节点或整个分支从文件中移出。但是这个方法需要一个参数，就是你想要移除出去的那个节点。请记住DOM考虑的是节点，而不仅是元素，因此元素的“孩子”可能是元素所包含的文本，而不是其他元素。

如同appendChild()一样，removeChild方法总是让所要移除元素的母元素来调用（因此才有“移除‘孩子’”的说法），也就是说同时需要母节点和要移除的节点的引用。让我们看看以下的标记模式。

```
<div id="parent">
 <div id="remove-me">
 <p>Pssh,I never liked it here anyway ,</p>
 </div>
</div>
```

脚本看起来就如下所示：

```
var parentDiv= document.getElementById("parent");
var removeMe= document.getElementById("remove-me");

parentDiv.removeChild(removeMe);
// Removes the div with the id "remove-me" from the page.
```

## 进一步阅读

以上应该使你了解了DOM脚本语言。当然，我只是对DOM的相关应用做了浅要的叙述，如果你想要深入学习，请认真查阅书籍：DOM脚本语言：用JavaScript和文档对象模型设计网站，第二版，作者Jeremy Keith和Jeffrey Sambells（Ed的朋友）。

## polyfills

目前你已经熟悉了这本书中的大量新技术：新的HTML 5元素、新的CSS3的处理方式、使用JavaScript操作DOM等。在一个完美的世界里，所有的

### 浏览器大战

JavaScript语言产生于一个黑暗的、毫无法律秩序的时期，网络标准还没有运行，浏览器世界的主要使用者为了更好的网络时代正在进一步完善它。由于一个令人差异的大事件：在最好的浏览器将占据市场的理念下，网景和微软实行了两个根本不同的DOM版本。

我将省略那些JavaScript语言之争所产生的腥风血雨的斗争细节，但是除非你掌握两种不同的编码技术，或者在你的网站上做一个“IE最佳视图/网景最佳视图”的警示标签，这两种竞争性的版本将因为它们的巨大差异而没有用处。

终于开始网络标准化运动了。在这样一个不公正的时期，W3C完成了我们逐步了解并喜爱的现代标准化的DOM的基础。幸运的是，Netscape和微软也与标准化运动在同一条船上。标准化的DOM同时支持IE 5和Netscape Navigator 6。

遗憾的是，IE在这个领域中停滞了很长一段时间直至IE 6的出现。结果，更老版本的IE与现代DOM有很多重大的不同。幸运的是，随着IE 9，以及不久后的IE 10的出现，它们正在赶上来。

麻烦的是，你的项目可能仍需要那些老版本IE使用者的支持。这很头疼，但是我们正在不断完善它。在处理过程中，我们拥有惊人数量的工具，例如，polyfills和拥有援助功能的JavaScript库，它们将使我们远离各个浏览器遇到的离奇的诡异事件，从而使工作简单化。



浏览器都将有条不紊，追随最新的技术，并且立即支持最新的技术（请看侧栏的“浏览器大战”）。在那样的完美世界里，那些赶不上变化的浏览器（我所想到的就是IE 6）将完全消失。可悲的是，那不是我们所生活的世界，不良浏览器仍然是每个研发者身上的肉中刺。

我承认我喜欢轮子的发明。一方面，这很值得学习。另一方面，这归咎于我们的汽车不能使用圆形石头和树节开动。但是当要处理每个浏览器的怪癖时，将不需要重新摸索。成千上万比我聪明的人都已经思考过处理它们的方法，并且已经找到聪明的方法来解决不符合JavaScript和DOM的浏览器的问题。可以使用JavaScript去修补JavaScript。

Polyfill是Remy Sharp在描述将不同浏览器行为标准化的JavaScript时创造的术语。

“一种模仿未来API并为旧浏览器提供后备功能的衬垫。”

——Paul Irish

这句名言所描述的未来还有很长的路要走，但是他表达的基本意思是：我们正在使浏览器创造他原本不能提供的新功能——就像检测用户物理位置的崭新功能，或者修理浏览器正常运行过程中的错误。

大量的polyfills都有特定的工作任务，比如使旧版本的浏览器识别新的HTML 5元素和CSS3选择器，以及一旦问题出现就立即出现的新功能。在这本书中，我将告诉你现代开发者手册中最常用的polyfills。你在网络设计奋战过程中会发现这些新东西非常实用。

## HTML 5 shiv（或者叫shim）

你可能记得在第5章看过相关内容，但是请注意现在你手上已经有JavaScript语言工具了。

HTML 5 shiv/shim是用来使IE 8及其之前的版本识别和使用像article、section和nav等HTML 5新元素的。

### 如何运作

HTML 5 shiv/shim上有几个变量，但是他们以同样的方式工作，运行DOM以寻找IE不能识别的元素，然后立即使用相同的可识别元素将其替换，从而使他们在DOM中对IE可见。现在我们针对这些元素写的任何样式都将如同预期一样正常工作。

### 谁制造了它

Sjoerd Visscher最初发现了这样的技术，并且现在还存在很多这样的脚本。现在Remy Sharp的版本可能是应用最广泛的。

## 如何使用它

脚本上的每一个变量都有一个要求：在文档的head部分进行编码，从而使IE能够在完成页面显示前分辨出这些元素。

```
<!--[if it IE9]>
 <script src="html5shim.js" ></script>
<![endif]-->
```

## 潜在缺点

主要的警告就是不能使用JavaScript的更老版本的IE接收没有样式的元素。

## 哪里可以获得并学到更多知识

- 维基百科HTML shiv入门 ([en.wikipedia.org/wiki/html5\\_Shiv](http://en.wikipedia.org/wiki/html5_Shiv))
- Remy Sharp原帖 ([remysharp.com/2009/01/07/html5\\_enabling-script](http://remysharp.com/2009/01/07/html5_enabling-script))

## Modernizr

Modernizr不是polyfill，也没有包括在polyfill中，而是相当于一个测试集，它能用来探测浏览器的特征，以及装载所需的polyfill。Modernizr小组还为大量polyfill库协管多种特征（请看先前的注意）。

---

注意：modernizr小组维护的polyfill存档可在[github.com/Modernizr/Modernizr/wiki/html5-Cross-Browser-Polyfills](https://github.com/Modernizr/Modernizr/wiki/html5-Cross-Browser-Polyfills)获取。

---

## 如何运作

Modernizr搜寻更新的HTML 5的JavaScript API所用的方法和函数，以及新的CSS3特性，并使用这些信息来决定浏览器是支持这些网页功能，还是必须接受polyfill。例如，如果浏览器包含了内置的用来与HTML 5的canvas元素互动的方法，我们就可以设定这个浏览器支持canvas。这就是著名的“特征检测”，而且在与更古老的UA（用户代理，或者浏览器）检测的比较下，它显得更好。Modernizr还包括类似于先前详细说过的HTML 5 shim。

## 谁制造了它

Modernizr由Faruk Ates创造，并被Paul Irish、Alex Sexton、Ryan Seddon和Alexander Farkas等进行了有效的完善。

## 如何使用它

Modernizr.com拥有一个生成器工具，可以将文本筛选成仅与你的项目相关，还有个包含整个测试库的“开发者”版本。如果你已经下载了一个定制版本，那么它仅包括你所需的外部脚本。

哪里可以获得并学到更多知识

- Modernizr网站 ([modernizr.com](http://modernizr.com))



## Selectivizr

Selectivizr可以使旧版本的IE正确处理复杂的CSS3选择器，如:nth-child和::first-letter。

### 如何运作

Selectivizr可以使用JavaScript提取和分析样式表的内容，并修补浏览器原生的CSS解析器的缺陷所产生的漏洞。

### 谁制造了它

Selectivizr由Keith Clark创造和维护。

### 如何使用它

Selectivizr必须和JavaScript库一起应用（接下来将详细阐述）。指向库.js文档的链接后面，指向脚本的链接将成为IE的条件注释，如下所示：

```
<script type="text/javascript" src="[JS library]"></script>
<!--[if (gte IE 6)&(lte IE 8)]>
 <script type="text/javascript" src="selectivizr.js"></script>
 <noscript><link rel="stylesheet" href="[fallback css]" /></noscript>
<![endif]-->
```

### 潜在缺点

因为撇开了原生的CSS解析器，所以在可用的浏览器中性能会有一些损失。

### 哪里可以获得并学到更多知识

- Selectivizr网站 ([selectivizr.com](http://selectivizr.com))

## Respond.js

Respond.js是一个快速的、轻量级的polyfill，它可以使得旧版本的浏览器（最普遍的就是IE 8及其以下版本）理解min-width和max-width的媒体查询，这些媒体查询广泛应用于自适应网页设计。

### 如何运作

如同Selectivizr一样，Respond.js仔细检查独立于浏览器内置解析器的样式表，分辨出min-width或max-width媒体查询，并根据浏览器窗口的宽度，使用JavaScript对页面中的元素应用这些样式。

### 谁制造了它

Respond.js是由我同事的Filament Group和jQuery Mobile 小组成员Scott Jehl创造的。它最初用于BostonGlobe.com网站的自适应设计，随后作为一个开源项目发行。

### 如何使用它

毫无疑问，你只需要下载Respond.js，并在文档head中的script标签

里引用它。

### 潜在缺点

同样，像Selectivizr一样，在使用这个脚本的浏览器中，可能会有一个小小的性能损失。

### 在哪里能够获取它并学到更多

- Scott Jehl的Respond page on github ([github.com/scottjehl/Respond](https://github.com/scottjehl/Respond))

## JavaScript库

现在继续讨论主题“你不必从头编写所有的程序”，现在应该介绍JavaScript库了。一个JavaScript库是一组预先写好的函数和方法，你可以在你的脚本中使用它们来完成常见任务，或简化任务的复杂性。

其实已经有许多JavaScript库了，有些是大的框架，包括最常见的polyfills、快捷方式和窗体小部件widget（以前你需要自己构建的Ajax Web应用程序）（见侧栏“Ajax是什么”）。有些库是针对特定的任务，比如处理表单、动画、图表或数学函数。对于经验丰富的JavaScript程序员来说，它能够节省大量的时间；而那些像你这样刚刚开始的人，一个库可以处理的任务，可能超出了你现在的技能水平。

### Ajax是什么

Ajax（有时写为AJAX）代表异步JavaScript和XML（Asynchronous JavaScript And XML）。“XML”部分不是重要的，你使用Ajax并不一定要使用XML（稍后有更多介绍），“异步”部分是重要的。

通常，当用户打开一个Web页面时，数据需要从服务器发送，一切不得不停下来等待数据传送，并且整个页面需要重新加载才可用。这不是很好的用户体验。

但随着Ajax的出现，因为页面可以在后台从服务器获取数据，你可以在更新页面的同时和用户进行顺利实时的交互。这使得Web应用程序感觉更像“真正的”应用程序。

你可以在许多现代网站看到这种处理，虽然有时候它微妙而不易察觉。在Twitter上，例如，滚动到下一个页面，装载一组新的tweets。这些并不需要编写页面标

记：它们是根据需要来动态加载的。谷歌的图片搜索使用类似的方法。当你达到当前页面的底部时，你可以看到一个按钮，允许你加载更多，但你从来没有离开当前页面。

术语“Ajax”由Jesse James Garrett在一次文章题为“Ajax：一个Web应用程序的新方法”中率先提出了。Ajax不是一个单一的技术，而是结合HTML、CSS、DOM和JavaScript，包括XMLHttpRequest对象，允许数据以异步方式传输。Ajax可以为数据使用XML，但它更常使用JSON（JavaScript对象符号），一个基于JavaScript的和人类可读的格式，来进行数据交换。

使用Ajax编写Web应用程序，不是一件你很快就能学会的事，但本章讨论的许多JavaScript库有内置的Ajax助手和方法，可以让你更简单地开始使用。



函数库的缺点是：一般而言，在一个大的.js文件中含有所有的函数，因此必须要求用户下载许多用不到的代码。但是许多的函数库作者都知道这个情况，从而把他们的函数库模块化，并且不断地优化他们的代码。在某些情况下，还可以定制脚本，并使用你需要的部分。

## 你必须知道的一些函数库

以下是一些最常用的JS函数库：

- jQuery ([jquery.com](http://jquery.com))。John Resig在2005年所写的jQuery到目前仍是最通用的JavaScript函数库，10000个最受欢迎的网站中有超过一半的网站在使用它。它是免费的、开放的资源，并且如果你已经能操作CSS、JavaScript和DOM，那么还有更方便的使用语法。你能够使用jQuery UI 函数库来补充jQuery，该函数库增加很多很酷的界面元素，诸如日程小窗口，拖放函数，扩展折叠列表，以及简单的动画效果。我在前边提到过，我也在研究jQuery Mobile。另一个基于jQuery的库，提供了UI元素和旨在完善各种移动浏览器和各种怪癖的polyfills。
- Dojo ([dojotoolkit.org](http://dojotoolkit.org))。Dojo是一个开放资源，模块化的工具箱，尤其有助于使用Ajax开发Web应用程序。
- Prototype ([prototypejs.org](http://prototypejs.org))。Sam Stephenson所著的Prototype JavaScript Framework，主要用于对Ruby添加Ajax支持。
- MooTools ([mootools.net](http://mootools.net))。MooTools（代表我的对象关联工作）是另一个由Valerio Proietti所编写的开源模块库。
- YUI ([yuilibrary.com](http://yuilibrary.com))。Yahoo! 用户界面库是另一个为构建富网络应用程序的免费开源库。这是Yahoo! YUI函数库项目的一部分，由Thomas Sha所创建。

对于处理特定函数的较小的JS库而言，因为他们一直处于创建和淘汰的状态，我建议对“支持\_\_\_\_\_的JavaScript库”进行网络搜索，看看什么是可用的。以下是一些函数库目录：

- 表单
- 动画
- 游戏程序
- 信息图表
- 图像以及canvas 3D效果
- 字符串以及数学函数

**注意：**对于二十多个JavaScript函数库以及它们的尺寸、特性的比较，参看维基百科上的“JavaScript框架的比较”：[en.wikipedia.org/wiki/Comparison\\_of\\_JavaScript\\_frameworks](http://en.wikipedia.org/wiki/Comparison_of_JavaScript_frameworks)。

Google开发网站上也包含许多更为流行的JavaScript函数库开放资源：[developers.google.com/speed/libraries/](http://developers.google.com/speed/libraries/)。

- 数据库处理

## 如何使用JS库 (jQuery)

运行我刚刚列出的函数库是很简单的。你所要做的就是下载JavaScript (.js) 文件, 将其装在服务器上并且在Script标签上标明, 然后就可以用了。这就是.js文件, 它含有所有预先编写的函数和语法快捷键。一旦拥有它, 你就可以利用框架中的特性来编写自己的脚本。当然, 使用它所完成的工作会很有意思 (遗憾的是, 这些超出了本章的范围)。

作为jQuery Mobile小组的一员, 我对其会有明显的偏爱, 因此我们需要在接下来的例子中继续练习。不仅由于它是最常用的函数库, 而且因为他们会在我提到jQuery时, 每次给我一美元。

### 下载jQuery.js文件

从jQuery开始, 浏览jQuery.com网站, 单击最大的下载按钮, 获取你自己的jquery.js复制。你可以选择“产品”版本, 它移除了所有额外的空格, 以保持较小的尺寸; 也可以选择“开发”版, 它更容易阅读但文件是“产品”版的八倍大。如果你自己不编辑则前者会更适合。

复制代码, 将其粘贴到一个新的纯文本文档, 然后用你在浏览器窗口的地址栏中所见的文件名进行保存。在本书写作时, jQuery的最新版本是1.7.2并且“产品”版本的文件名为jquery-1.7.2.min.js。将该文件与其他文件一起置于目录中。考虑到组织性, 一些开发者将他们的脚本置于一个.js目录中或者根目录中。不管你决定将它置于何处, 注意文件路径名, 因为在标记时会用到。

### 将其添加到文档

就像包含其他脚本的方式一样, 用一个script元素包含jQuery脚本。

```
<script src="pathtoyourjs/jquery-1.7.2.min.js"></script>
```

这种方法很好。然而, 值得一提的是还有另一种方法。如果你不想自己托管文件, 你可以指定一个公共的托管版本并且使用它。jQuery下载页面列出了一些选项, 包括Google服务器上的代码的链接。简单复制代码, 将其粘贴到文档head或</body>标签之前, 你就可以使用jQuery了!

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"></script>
```

### 准备就绪

你不想在文件和DOM准备就绪前使用脚本, 对吗? 嗯, jQuery有作为就



绪事件（ready event）的一种声明，它可以检查文件并等待，直到可以操作。不是所有的脚本都需要这样（例如，如果一个浏览器警报脚本），但如果你正运用DOM，在你的自定义script或者.js文件中包含这个函数，判断脚本的准备情况，其实是个不错的想法：

```
<script src="pathtoyourjs/jquery-1.7.2.min.js"></script>

<script>
$(document).ready(function(){
 // Your code here
});
</script>
```

## 使用jQuery编写脚本

一旦你设置完成，就可以开始使用插件来编写自己的脚本了。jQuery提供的快捷键代码可以分成两大类：

- 内置的特性检测脚本和polyfills集合
- 更短的、更直观的语法来选择元素（jQuery的选择器引擎）

在最后一节自己尝试后，你应该对什么是polyfills有一个更深刻的感知，所以让我们看看选择器引擎能够为你做些什么。

其一是jQuery的简化总是围绕DOM进行，因为你可以使用学习过的CSS选择器语法。下面是一个通过没有库的id值得到一个元素的例子：

```
var paragraph = document.getElementById("status");
```

语句发现带有ID为“status”的元素，并且把这个元素的引用保存在一个变量（paragraph）中。这个简单任务有很多特点。你可以想象当你访问页面上大量的元素时，代码会变得多么冗长。现在有了jQuery，但是，可以利用下面的快捷方式。

```
var paragraph = $("#status");
```

这是对的——这就是CSS中你了解和喜欢的id选择器。它不只是停在那里。任何你在CSS中使用的选择器都会在那个特殊的辅助函数中起作用。

你想用一个“header” class来找到所有内容吗？使用\$(".header")；

通过元素的名称呢？当然：\$("div")；

在你侧栏中的每个小标题呢？简单：\$("#sidebar.sub")；

你甚至可以使用基于属性的值来获取元素：\$("[href='http://google.com']")；

但这不能用选择器停止。可以用jQuery和库中的大量辅助函数来遍历DOM，就像一个蜘蛛人一样，编织这些函数和脚本。

jQuery也允许以CSS都做不到的方式（如获取一个元素的父元素）来将对象连在一起。假设有一个段落，要对其父元素添加一个class。我们没必要知道父元素是什么，所以不能直接指出父元素。在jQuery中可以用parent()对象来得到它。

```
$("#p.error").parent().addClass("error-dialog");
```

另一个主要的好处是，这是非常方便理解的：“找到类为‘error’的任何一段，并给它们的父元素添加类‘error-dialog’。”

## 但如果我不知道如何写脚本怎么办

学习JavaScript是需要时间的，在你可以自己写脚本之前需要一段时间。但是不要担心。如果你在网上搜索你需要的（例如，“jQuery image carousel”或“jQuery accordion list”），你会发现很多人已经创造并共享了很多脚本，并且已经形成了使用说明文档。因为jQuery使用一个选择器的语法与CSS非常相似，这使得它更容易使用你自己的标记来定制jQuery脚本。

## 小结

这两章中的全部内容，我们已经学习了变量的基础、操纵DOM，到利用JavaScript库。即使学完我们这里的内容，我们其实才刚刚开始了解JavaScript的功能。

下一次当你寻找一个有很酷的网站时，可以在你的浏览器中查看源代码，再看看周围的JavaScript。你可以从阅读中学到很多，甚至分解别人的代码。

记住，没有什么可以用JavaScript打破，这些就像不能用删除键实现撤销功能一样。

更好的是，JavaScript有来自整个社会的热情的开发者，他们渴望学习就像渴望教学一样。寻找志同道合的开发者，并分享你在这条路上学到的东西。如果你困在一个棘手的问题中，不要犹豫，去寻求帮助和问问题。你不太可能会遇到别人没有遇到的问题，并且开源的开发者社区总是乐意分享他们学到的东西。事实上，这就是你不得不跟我学习这两章的原因。



## 自我测验

几个非常简单的问题。

1. Ajax是什么技术的组合?

2. 这是在做什么?

```
document.getElementById("main")
```

3. 这是在做什么?

```
document.getElementById("main").getElementsByTagName("section");
```

4. 这是在做什么?

```
document.body.style.backgroundColor = "papayawhip"
```

5. 这是做什么的? 这是一个有点棘手的问题, 因为它嵌套着函数, 但你应该能够把它们理清楚。

```
document
 .getElementById("main")
 .appendChild(
 document.createElement("p")
 .appendChild(
 document.createTextNode("Hey, I'm walking here!")
)
);
```

6. 用右边的任务匹配polyfill

- |                |                                 |
|----------------|---------------------------------|
| a. HTML 5 Shim | 1. 添加支持::first-letter           |
| b. Respond.js  | 2. 对min-width和max-width媒体查询添加支持 |
| c. Modernizr   | 3. 为nav和aside添加支持               |
| d. Selectivizr | 4. 检查浏览器对canvas的支持              |

7. 使用如jQuery一样的JavaScript 库的好处是什么?

- a. 访问polyfills的包
- b. 可能更短的语法
- c. 简化Ajax支持
- d. 上述所有

# 创建Web图像

## 第五部分

---

### 本部分内容

- 第21章  
Web图像基础
- 第22章  
精简Web图像





# 第21章 Web图像基础

除非你计划发布纯文本的网站，否则你就需要了解如何创建Web图像。很多人都想亲自使用图像编辑程序，并获得一些基础的图像制作技能。如果你是一个熟悉打印、经验丰富的设计师，那么你需要修改样式，以使图像更适合Web传播。

本章涵盖Web图像制作的基础，从查找和创建图像的一些选项开始。然后，本章介绍可用的网页图像文件格式，并帮助你决定使用哪种格式。你还将学习图像分辨率、缩放和透明度的基础知识。

跟前面的章节一样，本章提供了循序渐进的练习。然而，我要指出，我写的内容基于一个假设：即你稍微熟悉图像编辑程序。在例子和练习中，我使用Adobe Photoshop（专业标准版），但是你可以使用本章列出的其他工具，跟着做大多数步骤。如果你是从零开始的，那么我建议你花些时间阅读关于图像软件的帮助文档或其他书籍。

## 图像来源

你必须要有图像，才能保存为新图像。所以在进入学习图像格式本质之前，首先来看获取图像的一些方法：通过扫描、拍摄或自己绘制，以及使用库存照片和剪贴画，或者雇人为你创建图像。

## 创建自己的图像

大多数情况下，给网站创建图像最划算的方法是从一开始就自己制作。你知道，额外的好处是你拥有使用这个图像的全部权利（接下来将再次讨论版权问题）。设计师可以使用扫描仪、数码相机或画图程序重置图像。

### 本章内容

图像来源

GIF、JPEG和PNG格式概述

图像大小和分辨率

在Photoshop中调整图像大小

二进制和Alpha透明度

SVG简介



## 专业工具

以下是对在专业图形设计师圈里使用的最流行的图形工具的一个简要介绍。除此之外还有许多其他的工具能制造出图形文件，如果你找到一个适合你的，这很好。

### Adobe Photoshop

毫无疑问，创建图形的行业标准是Photoshop。它包含了许多功能，专门用于创建Web图形。可以在[adobe.com](http://adobe.com)下载试用版和所有的Adobe软件。

### Adobe Fireworks

应用位居第二，Fireworks是第一个从基础到解决Web图形特殊要求的图形程序。它具有用于创建矢量（基于行的工具）和光栅（基于像素的）图像的工具，这一方面是独一无二的。

### Adobe Illustrator

Illustrator是在印刷和网页设计等行业标准的矢量绘图程序。它使Photoshop更加完整。

### Corel Paint Shop Pro

如果你使用Windows且预算有限，Paint Shop ProPhoto以较低的价格提供与Photoshop类似的功能。你可以在[corel.com](http://corel.com)下载试用版。

### GIMP

GIMP是一个免费的、开源的图像编辑工具，功能非常类似于Photoshop。它适用于Linux、Windows XP、Vista和Mac OS X。在[www.gimp.org](http://www.gimp.org)获得更多信息和免费下载。

## 数码相机

你可以使用数码相机拍摄你周围的世界，并将它导入图像编辑程序。根据图像的类型，你可以使用标准的数码相机，甚至手机摄像头，来获得你需要的图像质量。

## 电子绘图

如果你有绘画技能，那么可以在图画或照片编辑程序里绘制自己的图像。侧栏“专业工具”介绍了一些可用的现在最流行的图像程序。每个设计师都有自己最喜欢的工具和技术。对于Logo和插画，我建议从矢量画图程序开始，比如Adobe Illustrator或者Fireworks，创建Web可用的版本。你会发现，为打印和其他高分辨率应用提供高质量、与分辨率无关的版本是非常有用的。对于照片、纹理和其他位图图像类型，Adobe Photoshop是可选的专业工具。必须再次强调，创建高分辨率的图像很重要，你可以在需要的时候再制作小一点的复制品。

## 扫描

扫描是采集源材料的好方法。你可以扫描几乎所有东西，从平面画到三维对象。在搜索并使用已找到的图像时要小心。记住你找到的大多数图像可能有版权保护，需要许可才能使用，即使你做了相当大的修改也是如此。参考侧栏“扫描提示”，看一些指导信息。

## 库存照片和插画

如果对自己的设计技能不自信，或想从使用一些最新的图片开始，有大量成品的照片、插画、按钮、动画和纹理（texture）包可供购买或免费使用。库存照片和插画总体上可分为两大类：版权受控类和免版税类。

版权受控意味着版权所有人（或者他们的代理公司）管理再次使用图像的人。为了使用版权受控的图像，你在使用照片时必须获得许可。许可图像的一个好处是，你具有图像在某个具体媒介（如Web）或具体行业（如保健业或银行业）的排他性权利。不好的一面是，版权相当昂贵。根据许可的宽度和长度，价格标签可能是一幅图几千美元。如果你不需要排他性权利，只想用于Web，根据图像的来源，花费更可能是几百美元。如果听起来还是太不合理，可以考虑使用免版税的图像，这种图像不需要你付许可费。免版税的图像一次付费无限次使用，但是你不能控制别人使用这幅图像。免版税图像可以从顶尖的专业图库获得，比如，Getty Images的一幅图只需30美元，其他网站则更便宜（甚至免费）。

另一个获取免费图像的办法是查找创作共用许可（Creative Commons license）中艺术家发布的照片和绘图。因为有几种不同的创作共用许可，所以一定要查看条款。一些艺术家允许你自由使用，而另一些艺术家要



求你必须让他们署名；还有一些只允许图像用于非商业用途。

下列是一些我最喜欢的，可以寻找高质量的库存图片和插画资源，但并不详尽。上网搜索可以找到更多出售图像的网站。

#### Flickr Creative Commons ([www.flickr.com/creativecommons/](http://www.flickr.com/creativecommons/))

照片共享服务器Flickr是我在Creative Commons许可下寻找发布的照片的第一站。质量参差不齐，但我通常可以找到在成本内所需要的照片（如在第10章中的红色小熊猫）。根据“兴趣”来尝试使用Flickr的搜索工具Compfight ([compfight.com](http://compfight.com)) 寻找图像。

#### ISTockPhoto ([www.istockphoto.com](http://www.istockphoto.com))

如果你预算很紧（或者根本就没有），那没有比在ISTockPhoto寻找图像更好的方案了。价格一幅三美元起，这是我本人最喜欢的图像资源。

#### Getty Images ([www.gettyimages.com](http://www.gettyimages.com))

Getty是最大的图库，近些年来它战胜了竞争对手。它提供版权受控和免版税的照片和插画，有各种价格范围的。

#### Veer ([www.veer.com](http://www.veer.com))

我喜欢Veer，因为它往往比竞争者更新潮。它提供版权受控和免版税的照片、插画、字体和库存视频。

## 剪贴画

剪贴画涉及免版税的插画、动画、按钮和其他小摆设的集合，可以在大范围应用中复制和粘贴这些图像。有很多在线资源，并有一个好消息：一些网站免费提供图像，虽然你必须忍受弹窗广告的侵扰。其他网站收会员费，每年10~200美元。缺点是，它们中很多图像质量很低，还有些是人造的假图（但只是旁观者眼中的“假图”）。下面是一些带你入门的网站。

#### Clipart.com ([www.clipart.com](http://www.clipart.com))

这项服务收取会员费，但其组织良好，而且往往比免费的网站提供更高品质的作品。

#### #1 Free Clip Art ([www.1clipart.com](http://www.1clipart.com))

另一种无装饰的免费剪贴画网站。

很容易找到免费的或者价格低廉的网页版的图标，（简单地搜索“免费图标”就可以了）。这里有两个你可以开始的资源。

**注意：** 了解更多关于创作共同许可的信息请访问：[creativecommons.org/licenses/](http://creativecommons.org/licenses/)。

#### 扫描提示

如果你扫描图像用于Web，那么下面的提示将帮助你创建更高质量的图像。

- 因为缩小比放大更容易保留图像质量，所以扫描图像最好比实际需要的大一点。这样在后面缩放的时候你会有更多弹性，但不要过火，因为如果你把图像缩得太小，那么图像将变得模糊。图片尺寸的问题将在本章后面详细讨论。
- 在灰度（8位）模式下扫描黑白图像，而不要在黑白（1位或点阵图）模式下。一旦你设置了最终的尺寸和分辨率，这样还可以让你在其他区域作调整。如果你真的只想要黑白图像，那么可以在最后一步将图像转换成黑白的。
- 如果你扫描打印出来的图像，那么你需要消除打印过程中产生的点状图案。最好的解决方案是应用微小的模糊到图像（在Photoshop中，使用高斯模糊滤镜），稍微缩小图像，再使用锐化滤镜。这样可以消除这些讨厌的点。当然也要确保你有权使用这个打印的图像。



### The Noun Project ([thenounproject.com](http://thenounproject.com))

Noun Project从世界各地收集并组织单一颜色的经典图标，并免费提供使用。这该有多么酷呀！

### Icon Finder ([www.iconfinder.com](http://www.iconfinder.com))

这是一个巨大的资源库，有各种风格的免费全彩色图标。请务必核对Creative Commons的许可条款，这些条款因不同的图标集而不同。

## 雇设计师

寻找和创建图像要花很多时间，而且需要有特殊才能的人。如果你的钱多过你的时间和特殊才能，可以考虑雇图像设计师为你制作网站的图像。如果你开始就有一套原始照片和插画，你还可以利用本书所学技能来制作这些图像的Web版本。

## 格式简介

你一旦亲自处理图像，就需要给它们设置网页可用的格式。此外，世界上还有几十种图像文件格式。例如，如果你使用Windows，那你应该熟悉BMP图像，或者如果你是印刷设计师，你可能通常使用TIFF或EPS格式的图像。在Web领域，位图需要保存为三种格式：GIF（读作“jiff”或“giff”）、JPEG（读作“jay-peg”）和PNG（读作“ping”或“P-en-gee”）。

我还想让你了解第四种格式，SVG（可伸缩矢量图像），它是一个XML文本文件生成的矢量画图格式，这有些奇怪，所以我会在本章结束的时候来讲解。现在，我们要关注普遍使用的GIF、JPEG和PNG。

这对你来说可能是字母大杂烩，但是不必担心。在本节结束之前，你将能区分GIF图像和JPEG图像，并知道如何使用。下面是一个快捷提纲：

GIF图像最适合用于平色（flat color）和硬边（hard edge），或者当需要透明度或动画时。

JPEG图像在照片或平滑混色的图像中效果最好。

PNG文件可以包含任何图像类型，它们可以高效地保存平色图像。PNG是唯一一种允许多级别透明度的格式。

本节将解决术语，并深入挖掘每种格式的特性和功能。了解技术细节会帮助你以最小尺寸做出最高品质的网页图像。

## 随处可见的GIF

GIF (Graphic Interchange Format, 图像交换格式) 文件是常用的受网页欢迎的格式。它虽然不是专为Web设计的, 但是由于它的多功能性、小文件尺寸、跨平台兼容性而成为第一个被快速接受的格式。GIF还提供透明度和包含动画的能力。经过20年之后, 它依然是最受欢迎的网页图像格式。

GIF压缩方案在压缩平色图像方面表现出色, 因此它成为用于logo、艺术线条、图标(icon)等的最好的文件格式(如图21-1所示)。你也可以将照片或纹理图像存为GIF格式, 但保存效率不高, 会导致文件尺寸较大。GIF也比较适合由少量照片和平色大区域组成的图像。

为了制作真正好的GIF图像, 熟悉它们的底层工作机制和它们的工作范围非常重要。

### 8位索引颜色

在科技术语中, GIF文件是包含8位(8-bit)颜色信息的索引颜色图像(它们也可以另存为更低位深(bit depth)的图像)。接下来, 我们会依次解释这些术语。8位意味着GIF图像最多包含256种颜色——8位信息能定义的最大数( $2^8=256$ )。低位深导致更少的颜色种类, 还可以减小尺寸。

索引颜色意味着图像中的颜色集, 即色板, 存储在一个颜色表中(也称作颜色映射)。图像中的每个像素都包含一个数值引用(或“索引”), 索引确定颜色在颜色表中的位置。用一个简单的示例可以清楚地说明。图21-2显示了2位(4色)索引颜色图像如何引用它的颜色表。8位图像的颜色表中有256个位置。

当你在Photoshop中打开已有的GIF图像时, 你可以查看(甚至编辑)它的颜色表, 依次选择图像(Image)→模式(Mode)→颜色表(Color Table)(如图21-3所示)。当你使用本章稍后会讲到的Photoshop中“保存为Web格式”功能将图像导出为GIF格式时, 你会看到图像的颜色表预览。在FireWorks中, 颜色表显示在优化面板中。

### 合理命名文件

确保给图像文件使用正确的文件扩展名。GIF文件的名字必须使用.gif后缀。JPEG文件必须以.jpg(或者.jpeg, 但是不常用)为后缀。PNG文件必须以.png结尾。浏览器会查看后缀来决定如何处理各种媒体类型, 所以最好使用图像文件格式的标准后缀。

图21-1: 对于主要由平色和硬边组成的图像, GIF格式非常合适





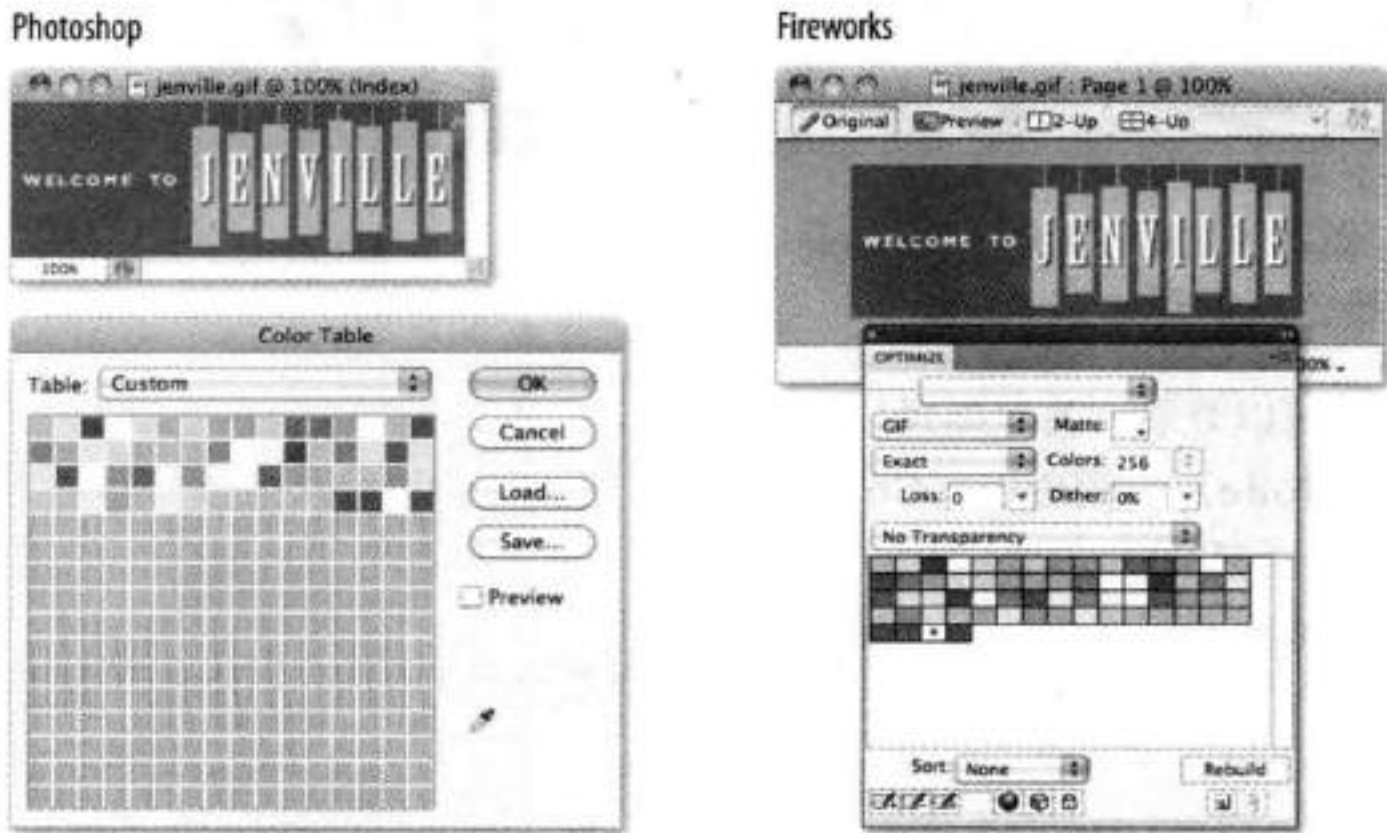
图21-2：一个2位图像和它的颜色表



多数源图像（扫描图、插画、照片等）开始是RGB格式，所以它们需要转换为索引颜色，这样才能保存为GIF图像。当图像从RGB模式变为索引模式时，图像中的颜色减少为256（或更少）色的色板。在Photoshop和Fireworks中，当你保存或导出为GIF的时候转换发生。其他的图像编辑程序可能需要你首先把图像转换到索引模式，第二步导出GIF。

在其他情况下，你需要为索引颜色图像选择一个色板。侧栏“普通颜色色板”，概括了大多数流行图像工具中各种可用的色板。推荐在Photoshop中使用可选择（Selective）色板和可感知（Perceptual）色板，在Fireworks中使用最合适（Adaptive）色板，在Paint Shop Pro中使用最优平衡杂质（Optimized Median Cut）色板，这样对所有的图像类型都会产生最好的效果。

图21-3：Photoshop和Fireworks中的颜色表，显示图像使用的64像素色彩



### 普通颜色色板

所有的8位索引颜色图像（包括GIF和PNG）都使用色板来定义图像中的颜色，有多个标准色板可以选择。一些是基于图像中的颜色制作的自定义色板。其他的是将先前已有的色板应用到图像。

**精确（Exact）**：如果图像包含少于256种颜色，那就创建图像中实际颜色的自定义色板。

**适用（Adaptive）**：使用图像中最常用的像素颜色创建自定义色板。当要保留图像的原始字符时，允许减少色深（color-depth）。

**可感知（Perceptual，仅用于Photoshop）**：通过把肉眼更敏感的颜色赋予更高的优先级，来创建自定义色板。不像适用色板，这基于算法，而不仅是像素个数。它产生的图像效果比适用色板更具有真实感。

**可选择（Selective，仅用于Photoshop）**：类似于可感知色板，但它考虑了广阔的颜色。

**Web适用（Adaptive）、受限（Restrictive）或Web216色**：创建216色色板之外的色板，在8位显示器上不抖动，8位显示器是古董，所以web安全色板不再适用了，所以这个不推荐使用。

**自定义（Custom）**：允许你载入以前保存的色板，并应用于当前图像。否则，它把当前颜色保存在色板中。

**系统（System，Windows 或 Macintosh）**：使用指定系统的默认色板中的颜色。

**最优平衡杂质（Optimized Median Cut，仅用于Paint Shop ProPhoto）**：使用类似于适用色板的方法把图像减少到很少的颜色。

**最优八叉树（Optimized Octree，仅用于Paint Shop ProPhoto）**：如果原图颜色很少，而且你想精确保留这些颜色，那就使用这个色板。

### GIF压缩

GIF压缩是“无损的”，这意味着为了压缩索引图像不会牺牲图像信息（虽然当RGB转换为有限颜色色板时，一些图像信息可能会丢失）。第二，它使用了利用数据循环的压缩方案（LempeZiv-Welch称为“LZW”）。当它遇到同样颜色的像素的字符串时，它可以压缩到一个数据描述中。这就是大面积平色的图像比带纹理的图像压缩效果好的原因。

用一个极端简化的例子，当压缩方案遇到一行14个同样的蓝色像素时，它会生成一个快捷符号来代表“14个蓝色像素”。下次再遇到14个蓝色像素时，它只需要使用代码快捷方式（如图21-4所示）。与之对比的是，当它遇到从蓝到浅绿的渐变色的一行时，它需要在整个过程中给每个像素存储一个描述，需要更多数据。当然，在专业术语中，实际发生的过程更复杂，但这个例子是一个很好的启发，在设计最大压缩率的GIF图像时要记住它。

### 透明度

你可以让GIF图像的一部分透明，这样背景图像或背景色可以显露出来。虽然所有位图本来是矩形的，但你可以让你的图像有更有趣的形状（如图21-5所示）。透明度将在本章后面详细讨论。

GIF压缩会给各个像素都保存为一条描述。

  
"14 blue"

在一个色彩渐变的图像中，它不得不为每个像素保存信息。信息描述越长，意味着文件尺寸越大。


  
"1 blue, 1 aqua, 2 light aqua..." (and so on)

图21-4：GIF图像使用的LZW压缩的简化示例





图21-5：透明可以让背景条纹透过图像显示出来



图21-6：交错GIF以一系列路径显示，每个路径都比上次清楚

**警告：**在写本书时，GIF动画在Android手机和平板电脑上是有问题的。虽然Android 2.2增加了对GIF动画的支持，但许多手机制造商将其默认为关闭，需要用户在设置中把它们打开（甚至都无法打开）。Opera Mobile 10+支持，但Opera Mini不支持。GIF动画在iPhone上可以正常工作。

## 交错

交错是你应用于GIF的效果，而让图像以一系列路径（pass）显示。每个路径都比前面的清晰，直到图像完全显示在浏览器窗口（如图21-6所示）。如果没有交错，那么一些浏览器会一直等到整个图像全部下载才显示图像。其他的浏览器可能每次显示几行，从上到下，直到整个图片完成。

在一个连接速度快的网络中，这些效果（交错或图像延迟）可能察觉不到。然而，在非常慢的网络中（例如，调制解调器或移动网络），可以使用交错来提供一个提示：图像正在下载。是否交错是你自己的设计决定的。但是我从不这样做，如果有一个特别大的图像，而用户中上网速度慢的占很大比例时，那么交错是值得的。

## 动画

GIF文件格式另一个内建的特征是能够显示简单的动画（如图21-7所示）。你见过的许多旋转的、闪烁的、渐入的甚至是其他的广告横幅（banner）都是GIF动画（虽然Flash影片用于Web广告越来越普及）。

动画GIF包含很多动画帧，这些帧是分散的图片，一起被快速查看，给出了动作或随时间变化的假象。所有的帧图像都存储在一个GIF文件中，附带描述它们如何播放的设置。设置中包括：这个序列是否重复、重复多少次、每帧保持可视多长时间（帧延迟）、一帧取代另一帧的方式（清除方法）、图像是否透明、是否交错。

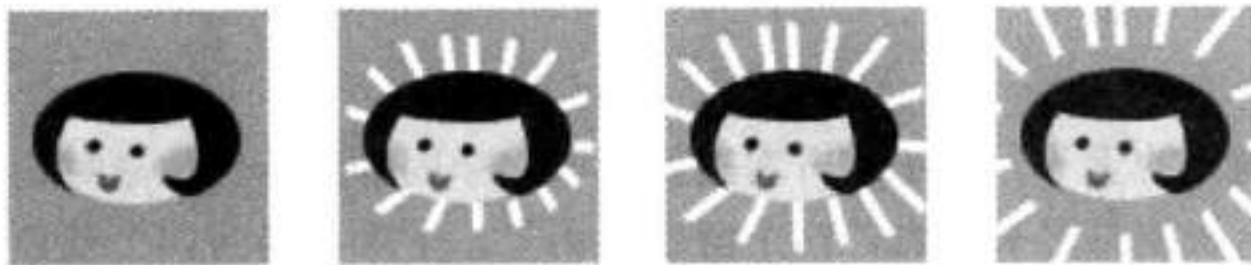


图21-7：简单动画的所有帧都包含在一个GIF文件中

Adobe Fireworks和Photoshop中有用于创建GIF动画的接口。在Photoshop CS5及更早版本中，使用“动画”窗口。CS6在时间轴窗口中选择“创建帧动画”。在网络可以搜索到许多专用的GIF动画工具，其中许多是免费的。

进一步阅读

## GIF动画

如果你想学习如何制作GIF动画，你可以从本书以前版本的动画GIF章节下载一个PDF，网址为[www.learningwebdesign.com](http://www.learningwebdesign.com)。那一章中包括动画设置和一步步创建动画指导的详细解释。

## 上镜的JPEG

Web上另一个流行的格式是JPEG，它代表联合图像专家组（Joint Photographic Experts Group），也就是创建它的标准组织。

不像GIF图像，JPEG图像使用喜欢渐变和混色的压缩方案，但在平色和硬边方面表现平平。JPEG的全色（full-color）和压缩方案使之成为照片图像的最佳选择（如图21-8所示）。

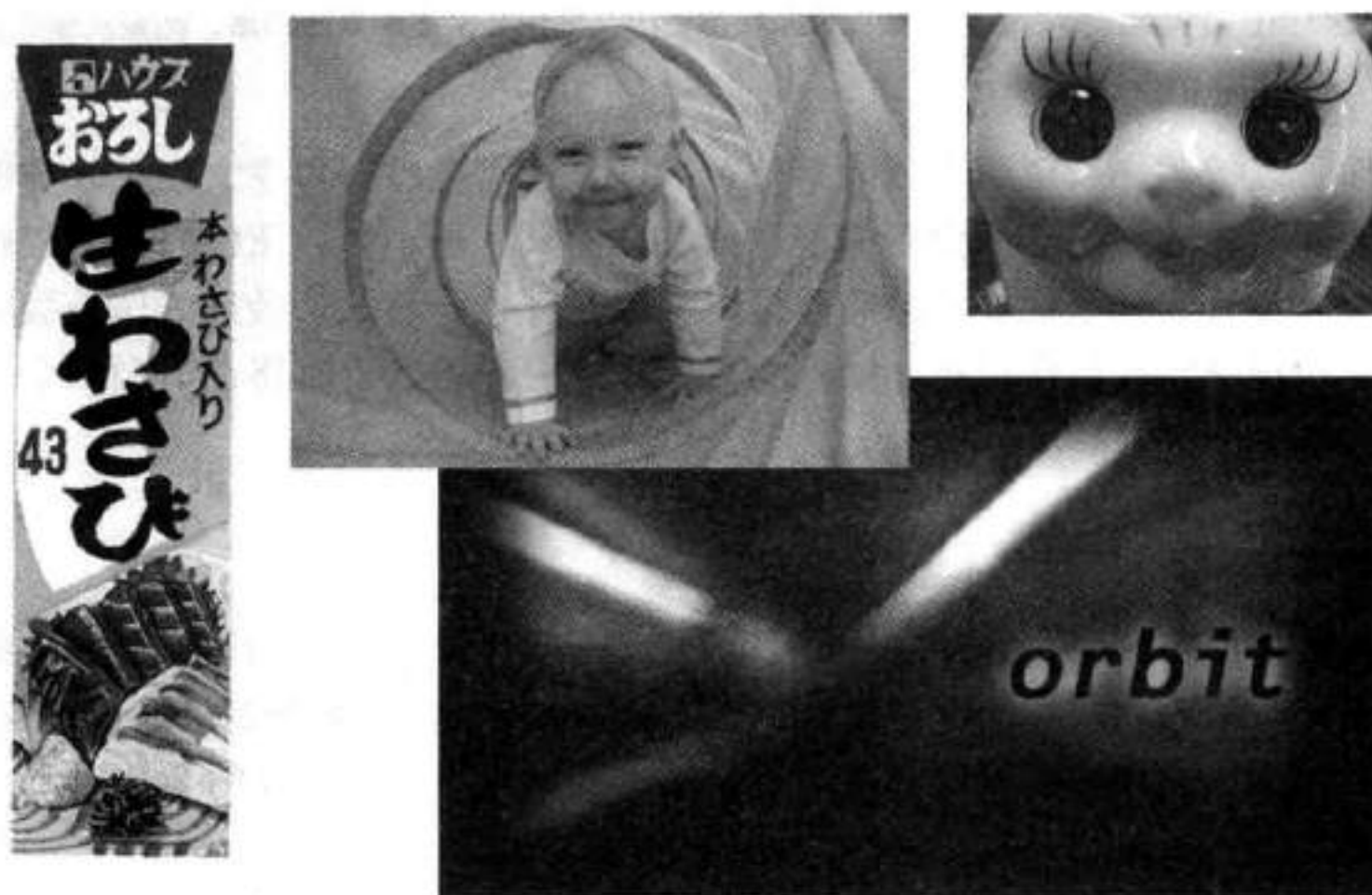


图21-8：JPEG格式最适合照片（彩色照片和灰度照片），或其他色彩微小渐变的图像

## 24位真色彩图像

JPEG图像不像GIF图像那样使用色板。相反，它们是24位的图像，能够显示RGB颜色空间（也称为真色彩空间，见“注意”）中的上百万种颜色。这是使JPEG最适合照片的一个方面——它们有你需要的所有颜色。使用JPEG，你不必担心像GIF一样限制在256种颜色内。JPEG图像更直观。

注意：在第14章中解释了RGB颜色。



## 警告

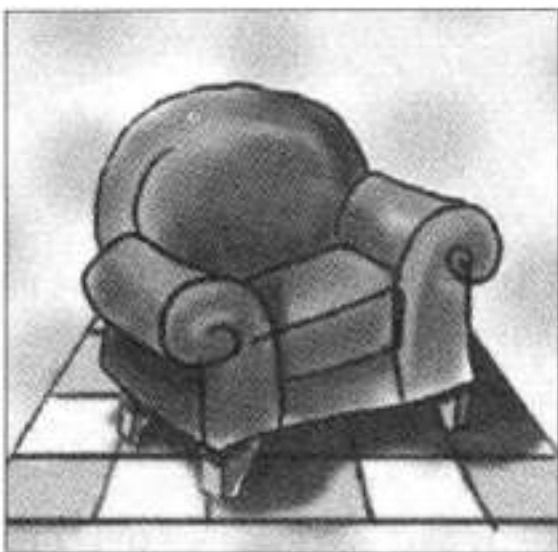
## 累积图像损失

图像质量一旦在JPEG压缩中损失，就再也不能恢复了。因此，你应该避免将JPEG再存储为JPEG。你每次都会损失图像质量。

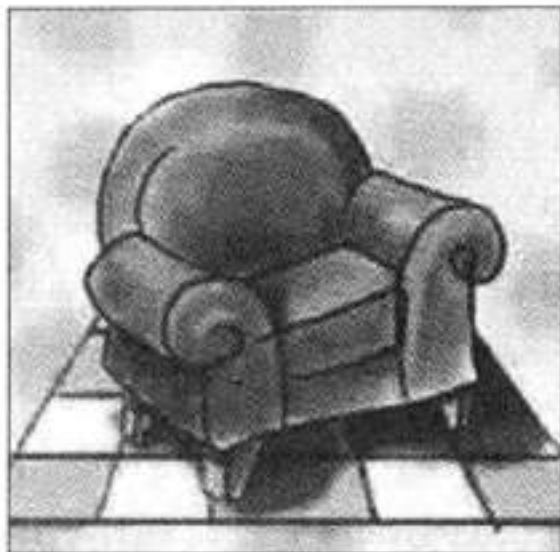
最好保留原始图像，需要时制作JPEG复本。这样，如果你需要改变JPEG版本，可以回到原图，保存并导出最新的图像。幸运的是，Photoshop的“存储为Web格式”的功能并不完全如此。Fireworks也保留原图，并让你保存且导出复本。

## 有损压缩

JPEG压缩方案有损失，这意味着在压缩过程中有些图像信息会丢失。幸运的是，大多数图像的大多数压缩级别的损失是不易察觉的。当图像使用高压压缩率压缩时，你会看到，压缩方案从图像取样的方法导致大斑点和色块（通常称为人工膺像）（如图21-9所示）。



原图



高压压缩图

图21-9：JPEG压缩会丢弃图像细节来获得更小文件尺寸。使用高压压缩率，图像质量会受损，如右图所示

你可以控制你希望图像压缩到什么程度。这需要综合考虑文件尺寸与图像质量间的平衡。图像压缩率越高（文件尺寸越小），图像质量受损越大。相反，如果你要求最高质量，文件尺寸也会最大。最好的压缩级别是根据图像的特殊用途和网站的目标来定的。压缩策略将在第22章中详细讨论。

## 渐进式JPEG图像

渐进式（Progressive）JPEG图像用一系列的路径显示（同交错GIF图像），从低分辨率的版本开始，每个路径都更清晰，如图21-10所示。在一些图像程序中，你可以指定用于填充到最终图像中路径的个数（3、4或5）。

图21-10：渐进式JPEG图像以一系列路径渲染



使用渐进式JPEG图像的好处是浏览者可以在完全下载之前熟悉图像。而且，这样改进通常可以稍微减小文件尺寸。缺点是消耗更多处理能力，并延迟了最终显示。

## 解压

在显示之前，JPEG图像需要解压；因此，浏览器在解码并汇编一个JPEG图像

上花的时间，比同等尺寸的GIF多。这些差别通常不易察觉，但是，这并不是不使用JPEG格式的理由。这只需要知道就行了。

## 令人惊异的PNG

加入网页图像大军的最后一个文件格式是万能的PNG（Portable Network Graphic，便携网络图形）。虽然开头很不顺利，但是当前使用的浏览器全部支持PNG图像，并且成为网页图像格式中很多开发者的首选。

PNG图像提供一系列不寻常的特征：

- 能够包含8位索引、24位RGB、16位灰度甚至48位彩色图像
- 无损压缩方案
- 简单开/关透明度（同GIF）或多级透明度
- 渐进式显示（与GIF交错类似）
- Gamma（亮度）调整信息
- 嵌入文本用于附加作者、版权等信息等

本节会详细讲述每个特征，并帮助你决定PNG是否是图像最好的选择。

## 多图像格式

PNG格式是为在线用途上取代GIF，在存储和印刷方面取代TIFF设计的。PNG可以用于保存很多文件类型：8位索引颜色、24位或48位RGB图像和16位灰度。

### 8位索引颜色图像

就像GIF图像一样，PNG图像可以使用最多256种颜色存储8位索引图像。它们也可以保存为1位、2位、4位色深。索引颜色PNG图像通常称为PNG-8。

### RGB/真色彩（24位或48位）

在PNG图像中，每个通道（红、绿、蓝）都可以定义为8位或16位信息，所以是24位或48位RGB图像。在图像程序中，24位RGB的PNG图像标识为PNG-24。需要注意，48位图像在Web上应用无效，甚至24位图像也要小心使用。由于它是无损的，24位PNG会比同一个图像的有损JPEG大很多。

### 灰度

PNG图像也支持16位灰度图像——总共有65 536种灰色（ $2^{16}$ ），在存储黑白照片和插画时可以保留每个精微的细节，虽然不适用于Web。

### 对于Retina显示屏使用渐进式JPEG

在一般情况下，没有必要逐行保存JPEG。但有一个例外情况，就是当你为Retina显示屏iOS设备创建双倍大小的JPEG文件时，此规则是例外（如撰写本文时）的。在这种情况下，为了解决移动Safari浏览器的一个字节限制问题，你应该保存为渐进式格式。随着Retina显示屏更普及，此替代方法在未来版本的Safari浏览器可能不再是必要的。其他特殊考虑将在本章后面侧栏“处理高密度显示器”中讨论。



**警告：** Windows Internet Explorer 6和更早版本不支持多级透明度。有关详细信息，请参阅本章透明度部分侧栏中的“Internet Explorer 6和Alpha透明度”。

## 透明度

PNG图像也能包含透明区域，使背景图像或背景色显露。然而，这个杀手级特征方面，PNG已经胜过GIF，它能够包含多级别的透明度，通常称作Alpha通道（或alphah）透明度。

图21-11显示了两个不同背景图像的相同的PNG形式。橙色圆圈是完全不透明的，但阴影中包含多个级别的透明度，范围从几乎不透明到完全透明的。存储在PNG中的多级别透明度让阴影与任何背景无缝结合。PNG透明度的来龙去脉将在接下来的“使用透明度”部分得到解决。

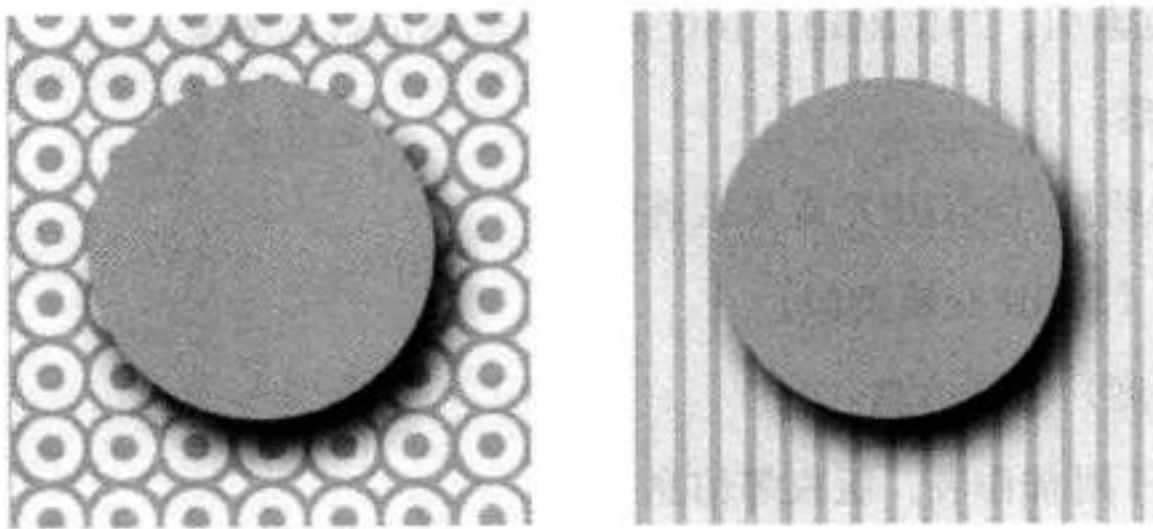


图21-11：Alpha通道透明度允许多级透明度，就如图中橙色圆圈PNG显示的下阴影

## 渐进式显示（交错）

PNG图像也可以编码为交错显示。当这个选项被选中时，图像将以七个路径的系列显示。不同于填充水平行的交错GIF，PNG图像从水平和垂直两个方向填充。交错增加文件尺寸，通常没有必要，所以为了保持文件尽可能小，可以关闭交错显示。

## Gamma修正

Gamma与显示器亮度设置有关。因为Gamma设置因平台而异，所以你创建的图像给终端用户显示的方法可能不如你所愿。无论创建它们环境的Gamma设置是什么，PNG能够加上信息标签。有软件可以显示PNG图像，并进行适当的Gamma补偿，这可以解释上面的问题。当应用到创建者和用户端时，PNG保持预期的亮度和色彩强度。

遗憾的是，在写本书时，这个特性并不能在实际中使用。IE（所有版本）不能正确地显示Gamma，而PNG最后会比想象得黑。Photoshop最后会在PNG中嵌入CS3开始的Gamma信息。注意，在一些浏览器上，可能很难让一个PNG图像匹配背景色，甚至RGB值相同都不可以匹配。解决的方案是使PNG图像的边缘透明，这样背景就能透过来了，或者你也可以使用GIF。

嵌入文本

PNG图像还能存储字符串。这对永久附加文本到图像（比如，版权信息或图像的描述）是很有帮助的。能将文本注释嵌入PNG图像的工具仅有Corel Paint Shop ProPhoto和GIMP。理想情况下，PNG中的元信息可以通过右击浏览器中的图像来访问，但这个功能还没有在当前浏览器中实现。

何时使用PNG图像

PNG图像包含很多强大的选项，但网页图像格式之间的竞争几乎可以归结为文件尺寸的比较。

对于一般保存为GIF的图像，8位PNG是一个好的选择。你可以发现图像的PNG版本比相同图像的GIF版本文件尺寸小，但这跟你的图像程序处理PNG压缩的效率有关。

虽然PNG不支持24位颜色图像，它的无损压缩方案，几乎总是导致文件尺寸比应用于同一图像的JPEG压缩大。用于Web，JPEG仍是照片和渐进式混色图像的最佳选择。

如果你想利用透明的多级别，那么“最小文件取胜”规则就不适用了。这种情况下，PNG是你唯一的选择，也许值得用尺寸有点大的文件了。

接下来将从更宽的角度来寻找最好的图像格式。

选择最优格式

制作高质量的图像，同时保证质量和快速下载，有一个诀窍是选择正确的格式。表21-1提供了一个好的起点。

表21-1：选择最好的文件格式

如果你的图像	使用	因为
是平色绘画	GIF或8位PNG	GIF和PNG在压缩平色方面出色
是照片或包含渐变颜色	JPEG	JPEG压缩在混色图像上工作良好。由于有损，它可以得到比24位PNG图像更小的文件尺寸
由平色和照片结合而成	GIF或8位PNG	在保存和压缩平色区域方面，索引颜色格式最好。减少为一个色板，导致在照片区域显示的抖动通常没什么问题
需要透明度	GIF或PNG	GIF和PNG都允许开/关图像透明度

注意：如果你致力于PNG格式并且绝对要求质量，你可以使用像PNGcrush一样的实用程序从PNG中删除Gamma (gAMA) 信息，正如Trevor Morris在这篇文章中详细介绍的一样 ([morris-photographics.com/photoshop/articles/png-gamma.html](http://morris-photographics.com/photoshop/articles/png-gamma.html))。



使用RGB模式

无论文件的最终格式如何，你应该一直在RGB模式下进行图像编辑（非彩色图像也可用灰度模式）。要在Photoshop中检查图像的颜色模式，可以选择图像（Image）→模式（Mode），确认RGB附近有选中的标记。

JPEG和PNG-24文件直接压缩RGB颜色图像。如果文件要保存为GIF或PNG-8，那么RGB颜色必须转换到索引颜色模式，可以手动进行，或者在“保存为Web格式”和“导出”过程中进行。

如果需要编辑已有的GIF或PNG-8图像，应该在进行任何编辑之前转换图像到RGB。这样使你能够在调整图像时，编辑工具使用全部RGB频谱中的颜色。如果你要缩放原始的索引颜色图像，将得到恶心的结果，因为新图像颜色局限于颜色表中的颜色。

如果你曾创建过印刷用图，你可能习惯在CMYK模式下工作（印刷颜色由青色（Cyan）、洋红色（Magenta）、黄色（Yellow）和黑色（black）组成）。CMYK对于网页图像是不相关且不适合的，所以在图像编辑过程中，一开始就要转换到RGB模式。

表21-1：选择最好的文件格式（续）

如果你的图像	使用	因为
需要透明度的多级别	PNG	PNG是唯一支持Alpha通道透明度的格式
需要动画效果	GIF	GIF是唯一能包含动画帧的格式

以你选中的格式保存图像

实际上，每个最新图像程序都允许你以GIF、JPEG和 PNG格式保存图像，但有一些程序有更多的选项。如果你使用Photoshop、Fireworks 或 Corel Paint Shop Pro，确保利用特殊的网页图像功能。

以可利用的最高质量的RGB图像开始——你永远不知道在其他情况下是否需要使用它。当你调整完图像（裁剪和色彩校正等），保存完整大小的图像，这样你就能够有一个好的原始图像。然后，你可以调整图像的大小以使其适用于网页。事实上，近来许多图像定位到不同的设备尺是很常见的，这就更有理由要保存一个整洁的、高品质的原图像。当你调整完（在本章的后面，我会展示调整大小的技术），可以按照这些指令将其保存为GIF、JPEG和PNG。

Adobe Photoshop

打开Photoshop的“存储为Web”对话框（文件（File）→存储为Web（Save for Web））（如图21-12所示），从弹出菜单中选择文件类型。在选择格式时，面板显示与格式相应的设置。保存为Web格式窗口还显示效果图预览和文件尺寸。你甚至可以同时比较不同的设置；例如，相同图像的GIF和PNG-8版本。一旦选中了文件类型并完成设置，单击“存储（Save）”按钮，并命名。

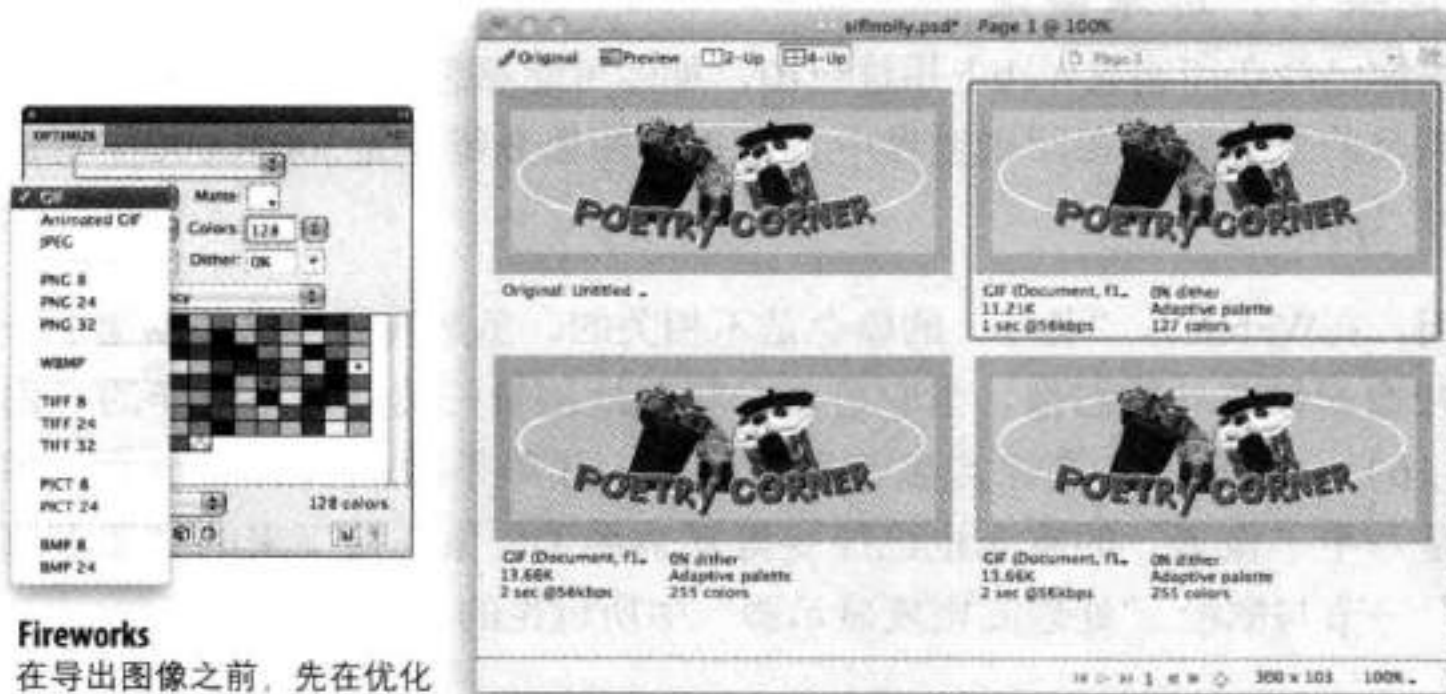
本章后面缩放图像和使用透明度时，将再次看到“存储为Web”对话框。第22章讨论与优化相关的各种设置时，它又会出现。

Fireworks

打开图像，切换到预览选项卡（Preview tab），然后就可以在优化（Optimize）面板中选择文件类型（如图21-13所示）。完成设置后，在文件（File）菜单选中导出（Export），给图像命名。



图21-12: 在Photoshop的“存储为Web”对话框中选择一个文件类型



#### Fireworks

在导出图像之前，先在优化面板中选择一个文件类型。

图21-13: 在Fireworks优化面板中选择文件类型

#### Paint Shop Pro

通过文件（File）菜单的导出（Export）选项，可以访问GIF优化器（GIF Optimizer）、JPEG优化器（JPEG Optimizer）和PNG优化器（PNG Optimizer）。每个都打开一个多面板的对话框，其中包含各自文件类型



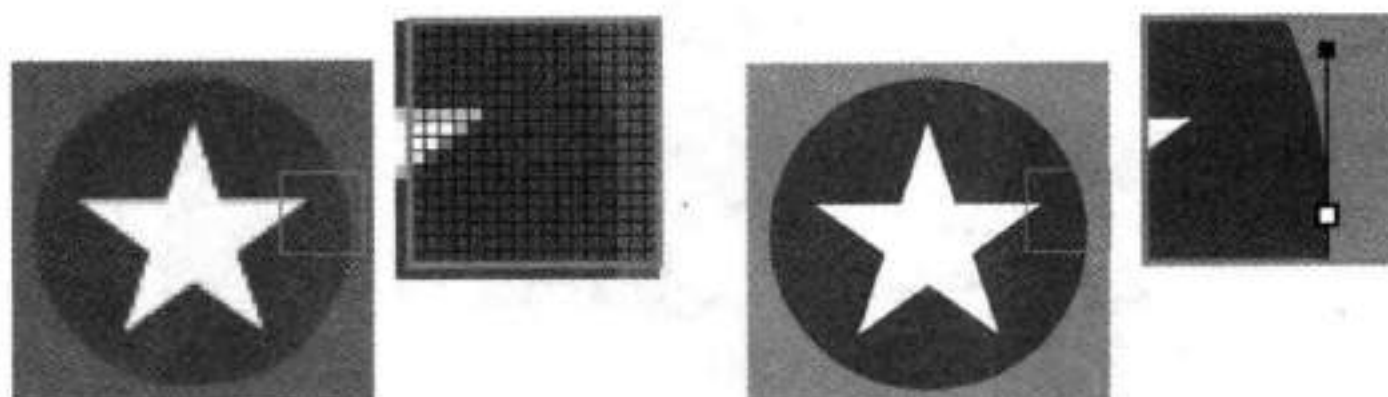


图21-14: Corel Paint Shop Pro中的Web优化选项

的设置和压缩图像的部分预览效果。GIF优化器的颜色面板如图21-14所示。当你完成所有设置后，单击“OK”按钮。注意，在能够访问设置项之前，你需要选择文件类型，这里无法比较不同图像类型的预览图，而在Photoshop和Fireworks中是可以做到的。

## 图像尺寸和分辨率

GIF、JPEG和PNG图像的一个共同点是，它们都是位图（也叫做光栅）图像。当你放大位图时，你可以看到它像是由很多像素（小的单色方格）镶嵌而成的。这与矢量图不同，矢量图由光滑的线和填充区域组成并基于数学公式运算。图21-15描述了位图和矢量图的区别。



位图由不同颜色像素的网格组成，就像一个马赛克。

矢量图使用数学方程来确定形状。

图21-15: 位图和矢量图

## 送走英寸，迎来像素

如果你已经在印刷或Web上用过位图，那你可能熟悉术语“分辨率”，即每英尺的像素数。在印刷世界中，常用的图像分辨率是300和600像素每英寸（ppi）。

然而，在Web上，“英寸”的概念是不相关的。虽然我可能已经创建了一个72像素每英寸的图像，当它显示时精确地测量一英寸是不可能的（图21-16）。事实上，高密度显示屏的出现，如苹果的Retina显示器，即使是一个“像素”的概念也已经变得复杂多了，像在接下来的“像素疯狂”一节与侧栏“处理高密度显示器”中所讨论的一样。

如果你抛开英寸，你也要折腾“像素每英寸”。我们唯一可以肯定的是，图21-16的图形有72像素宽，它的宽度将是36像素图形的两倍。Web设计师会测量图像中的像素总数，因此图像的分辨率在技术上是不相关的。

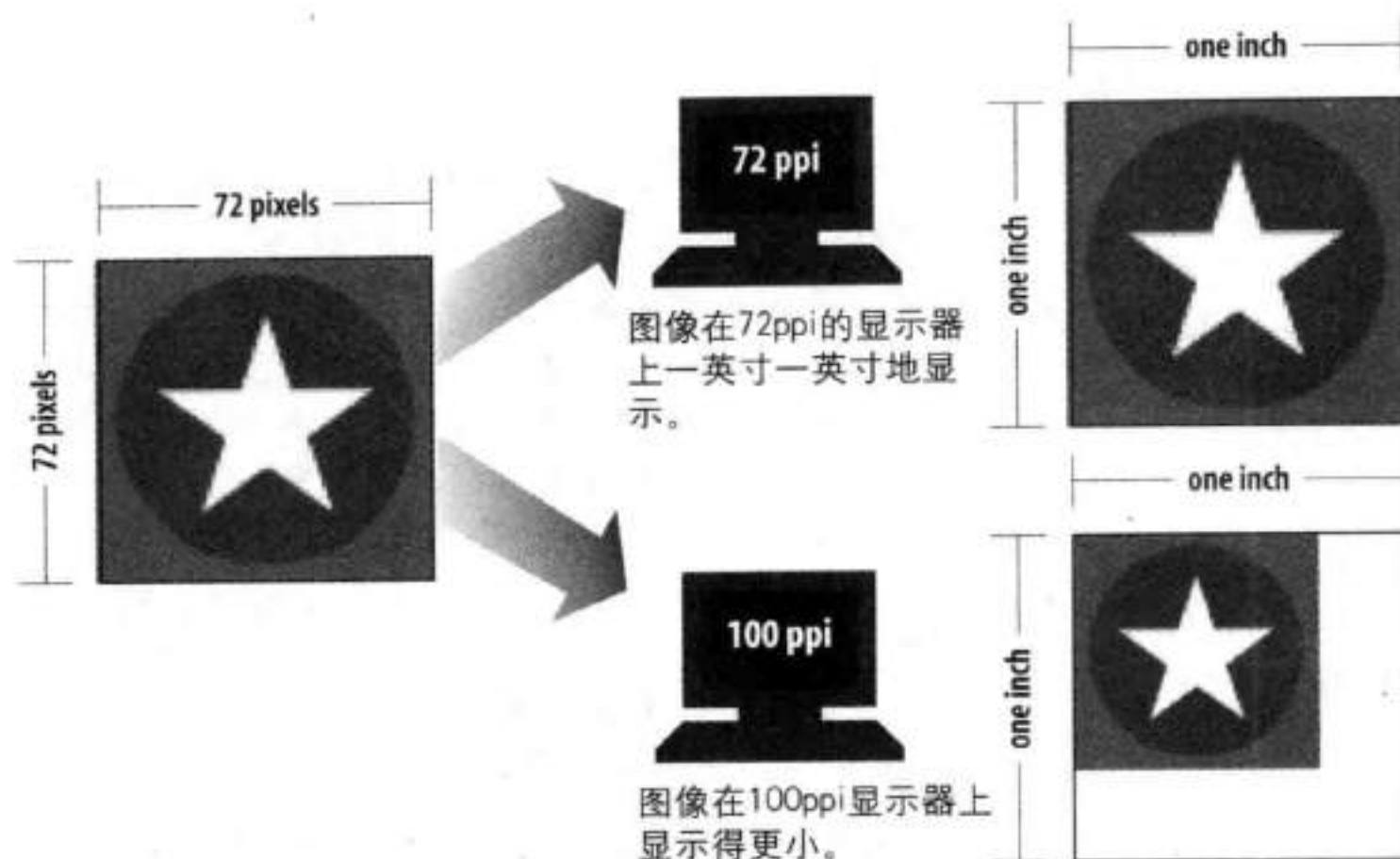


图21-16：英寸以及“每英寸像素”，与数字媒体不相关，在数字媒体中，图像尺寸依赖于显示器分辨率

话虽如此，但我知道，大多数设计师在72ppi下创建自己的图片，只是为了进入活动领域。我发现当我在72ppi下创建所有的图像时，并在Photoshop中以100%的比例查看它们时，它们出现在桌面上的显示器会保持相互的比例，并显示为差不多大小。当旨在创建高密度显示图片（如苹果的Retina显示器）时，72ppi是一个很好的起点分辨率，只需加倍像素尺寸。

## 像素疯狂

就在不久前，我们可以使用桌面显示器的像素硬件数清楚地一对一映射图像的像素。在大多数情况下，这仍然是正确的，但也出现了打破这条规则的技术进展。

首先，现在很多浏览器无论图像规模大小，会自动缩放大图像，以适应浏览器窗口，并且允许用户放大网页，因此1:1的映射会丢失。为了适合小型手持设备，图片会明显缩小。

制造商一直在推动显示器的分辨率，使其越来越高。其结果是，一个实际的硬件像素是如此之小以至于如果图像和文本进行一对一映射，那么它们将会小得难以辨认。为弥补这一缺陷，设备使用参考像素的测量值来衡量图像和文本中的像素，同时CSS规则是被映射的。在新的iPhone、iPad和MacBook Pros笔记本电脑的Retina显示屏上的参考像素宽度相当于两个硬件像素。在一些Android平板电脑中的参考像素为1.5硬件像素。作为网页设计师，我们在工作中加入了一层新的复杂性。（见侧栏中的“处理高密度显示”。）

### 点/英寸

因为Web图像只存在于屏幕上，所以可以用像素/英寸（Pixels Per Inch，ppi）的单位来精确计量分辨率。

然而，当用于印刷时，设备和打印页都用点/英寸（Dots Per Inch，dpi）来计量，点/英寸代表每英寸的图像中印刷点的数量。在图像中，dpi与ppi可能相同，也可能不同。

在你的经历中，你可能听说过术语dpi和ppi通用（虽然这样不正确）。理解它们的区别很重要。

**注意：**要想有更深入的了解，我推荐Scott Kellum的文章“A Pixel Identity Crisis”（[www.alistapart.com/articles/a-pixel-identitycrisis/](http://www.alistapart.com/articles/a-pixel-identitycrisis/)）。



## 处理高密度显示屏

想象一下一个为了充分发挥其潜力而需要巨大图像的设备，但可能要在最慢的3G网络上访问这些图像。这正是我们在2012年春季发布的iPad 3的情况。新iPad使用的Retina显示屏分辨率为 $2048 \times 1536$ ，这是一个高达310万像素的显示屏。随后不久发布了Retina MacBook Pro显示屏，其分辨率为 $2800 \times 1800$ 。我敢肯定，这仅仅是个高密度显示趋势的开始。

这对于消费者来说是令人兴奋的，因为图像可以清晰地显示，并精确地打印。但是，新的Retina显示屏使我们的网络开发人员暂时陷入循环困境中。在Retina显示屏中，常规的网络图像稍微显得模糊和像素化。为了让图像看起来清透，必须加倍图像的尺寸，并且让浏览器使它们的大小符合布局的预期尺寸。图21-17比较了标准的Web图形与特地针对Retina显示屏创建的2倍大小的相同图像。

遗憾的是，当你的网络图像的尺寸加倍时，它的像素数就是原来的4倍了，并且文件大小高达原来的4倍。而我们知道，在Web上，性能就是一切。高密度显示器的设备可能更偏好于显示高清晰度的图像，但是，这并不意味着网络增长速度足以满足这一需要。

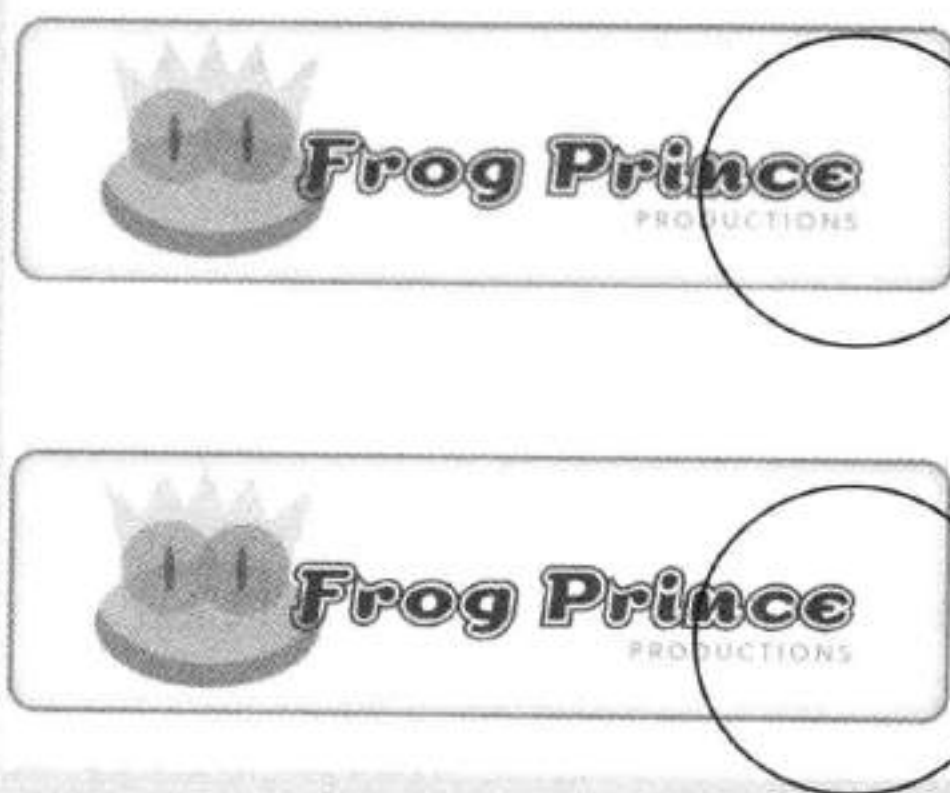
那么，该怎么办呢？说实话，我们仍然在想处理高分辨率设备带来的不可避免的冲击问题的策略。我们的网站要求高分辨率，将成为网络设计师工作的另一方面。

下面是我们了解的一些事情：

- 图像的尺寸增加一倍以使它们在高分辨率下看起来清晰，如图21-17所示。
- 苹果公司的Safari浏览器对JPEG在页面上显示的兆字节数有限制。对于超过两百万像素（210万像素）的图像，它会自动降

级，清晰度也会损失。要解决Safari浏览器对JPEG的限制，请在逐行格式下保存JPEG文件。

- 并不是每一个图像都需要超大尺寸。只为最重要的图像或在页面上的图像（在商务中称为英雄形象）才考虑创建2倍图像。这可能包括一个单一的状态设置图像，你的标志，或产品照片中的细节是很重要的，如指示面料的质地。
- 你可以使用CSS媒体查询以检测设备是否有2倍分辨率，并对这些小型设备提供适当大的图像，避免小型设备获得不必要的大图像。你也可以使用JavaScript用2倍图像替换标准图像。
- 遗憾的是，要知道使用Retina或其他高密度显示屏的人不会告诉你任何有关他的网络速度的信息，所以你可能会有连接速度慢、不能迅速显示图像而发送大图像的风险。发现用户当前连接速度的策略还在发展，也超出了本章的范围。因为这些技术正在迅猛发展，我建议你自己在网上搜索最新的思想。



标准的网页图像在retina显示器上看起来模糊失真。这个PNG是350像素宽，它在一个img元素中，设置为350像素宽。

在retina显示器中，当图像创建为最终布局的两倍时，看起来非常清晰。这个PNG是700像素宽，而img元素设置为350像素宽。

图21-17：仔细查看典型的网页图像，对比其在Retina iPad上的显示。



调整图像大小

由于源图像一般是不适用于Web的，所以图形制作的很大一部分时间都花在调小图像上了。所以图像大小调整是一项必备的基本技能。

在练习21-1中，我将向你展示，使用Photoshop的“存储为Web”功能缩放图像，这个方法很简单。使用这个方法，导出的网页图像被重置了尺寸，但原图保持不变。这使我们很容易把同一个图像保存为多种不同的尺寸，以适应不同的设备。在其他的程序中，或者如果你想更多地控制最终图像质量，可以看看练习后面的侧栏“使用图像尺寸”。

注意：如果你没有Photoshop，你可以在[www.adobe.com/downloads](http://www.adobe.com/downloads)下载一个免费试用版，并跟着练习做。

练习21-1 在Photoshop中缩小图像

本练习中将使用一张高分辨率的图片，并设置它的尺寸使其适合于网页。源图像ninja.tif可以从本章的materials文件夹中获取：[www.learningwebdesign.com/4e/materials/](http://www.learningwebdesign.com/4e/materials/)。

在Photoshop中打开ninja.tif。在图像中选择所有的像素（选择→全部），然后在信息面板中检测像素尺寸（如图21-18(A)所示）。如果信息面板没有打开，选择窗口→信息。如果度量单位是英寸或者其他单位，可以在首选项中改为像素（Photoshop→首选项→单位&规则）。ninja图像是1600×1600像素，对于网页来说太大了。对这个例子来说，假定页面布局的空间为400像素的正方形。

现在，我们将一口气缩放图像并保存为JPEG图像。从

文件（File）菜单选择“存储为Web所用格式”。从格式弹出菜单中选定JPEG。(B)

在设置栏底部，使用图像尺寸设置(C)，输入你希望保存的尺寸，也就是本例的400像素。链接图标选中后，当你输入新的宽度时，高度会自动改变。

接下来，选择品质（Quality）(D)。为了达到最好的效果，我通常使用“两次立方”或“两次立方（更锐利）”。你可以在优化图像视图中看到重置尺寸后的图像（如果还不是，就选择上面的选项卡）。

单击“存储（Save）”(E)，给文件命名，选择保存目录。“存储为Web所用格式”对话框关闭后，你会看到ninja.tif源图并没有改变，这样你就可以以相同方式制作不同大小的图像。保存文件，保存最近导出的设置。

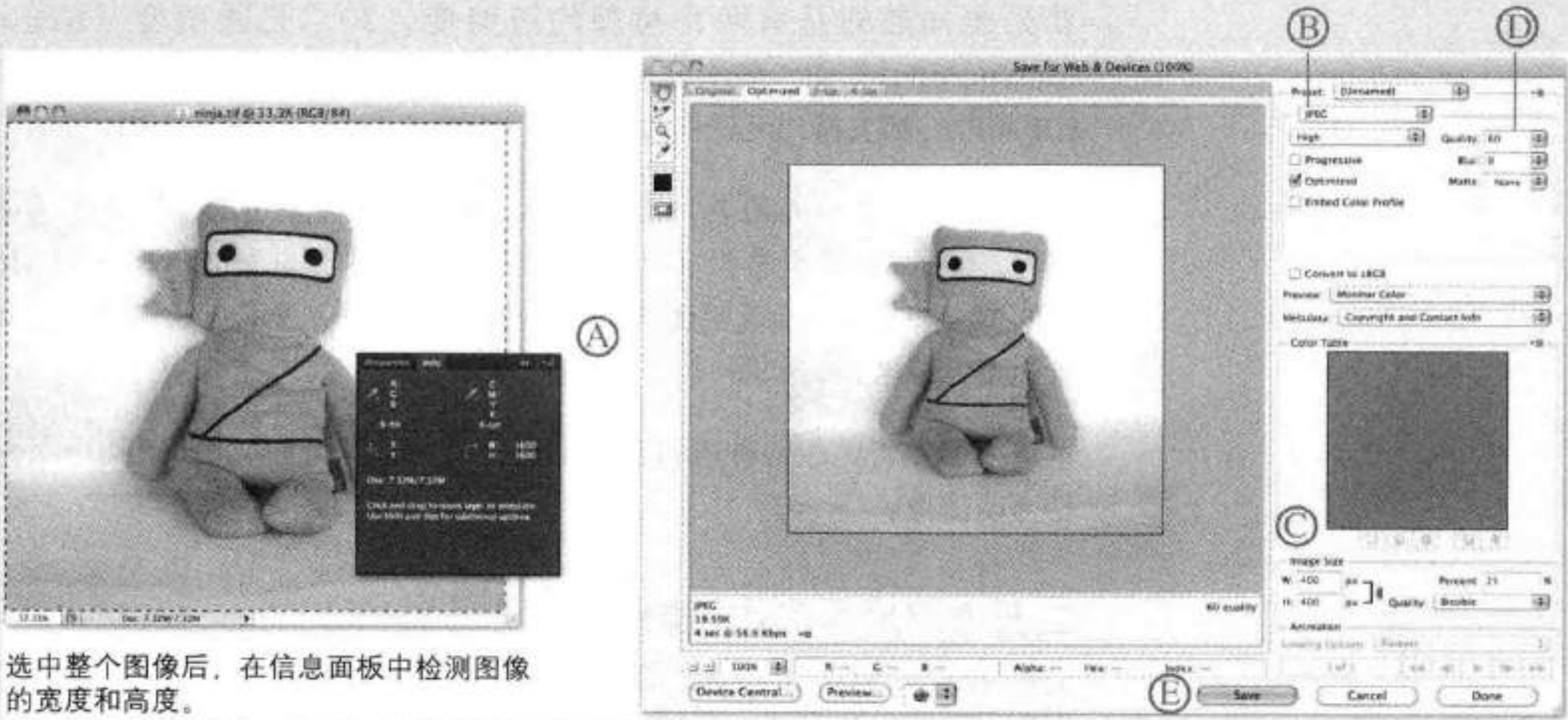


图21-18：使用“存储为Web所用格式”对话框来调整图像



## 使用图像尺寸

练习21-1中方法的缺点是你不能控制图像质量。如果你是一个图像质量控制狂，你可能更喜欢使用图像大小对话框（图21-19）来重置图像尺寸。在Fireworks中，修改（Modify）→画布（Canvas）→图像大小中有相似的选项。

确保底部的重新取样（Resample Image）和约束比例（Constrain Proportions）都被选中，品质（Quality）设置为两次立方（或两次立方（较锐利））。分辨率对网页图像来说并不重要，就像我们前面讨论的。

然后在对话框顶部输入想要的最终像素尺寸，并单击OK。双击放大镜工具（图上没有显示）以100%显示重置尺寸后的图像。

现在可以应用锐化滤镜和其他效果，使用“存储为Web所用格式”，来以Web格式输出图像。

分成两步来减小大文件可以帮助保持质量。首先，设置为中间尺寸并用锐化滤镜来锐化。然后设置为最终尺寸并再次锐化。你不能用“存储为Web所用格式”的方法来实现这个结果。

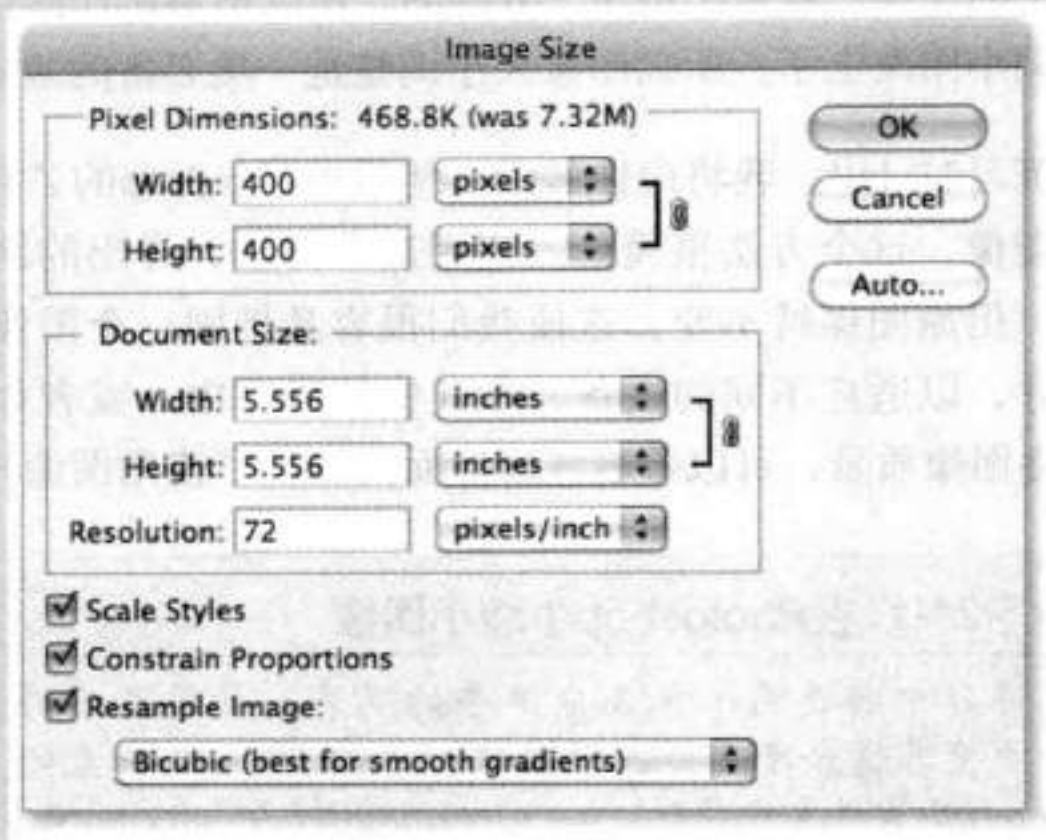


图21-19：Photoshop中的图像尺寸对话框

**警告：**记住，图像大小设置会改变原始图像的尺寸。不要保存，否则你会丢失高质量版本。

## 使用透明度

GIF和PNG都允许图像的一部分透明，允许背景色或背景图像显露出来。本节将进一步了解透明图像，包括制作的提示。

首先要知道的是有两种类型的透明度。在二元透明度（binary transparency）中，像素要么完全透明要么完全不透明，就像一个开关。GIF和PNG都支持二元透明度。

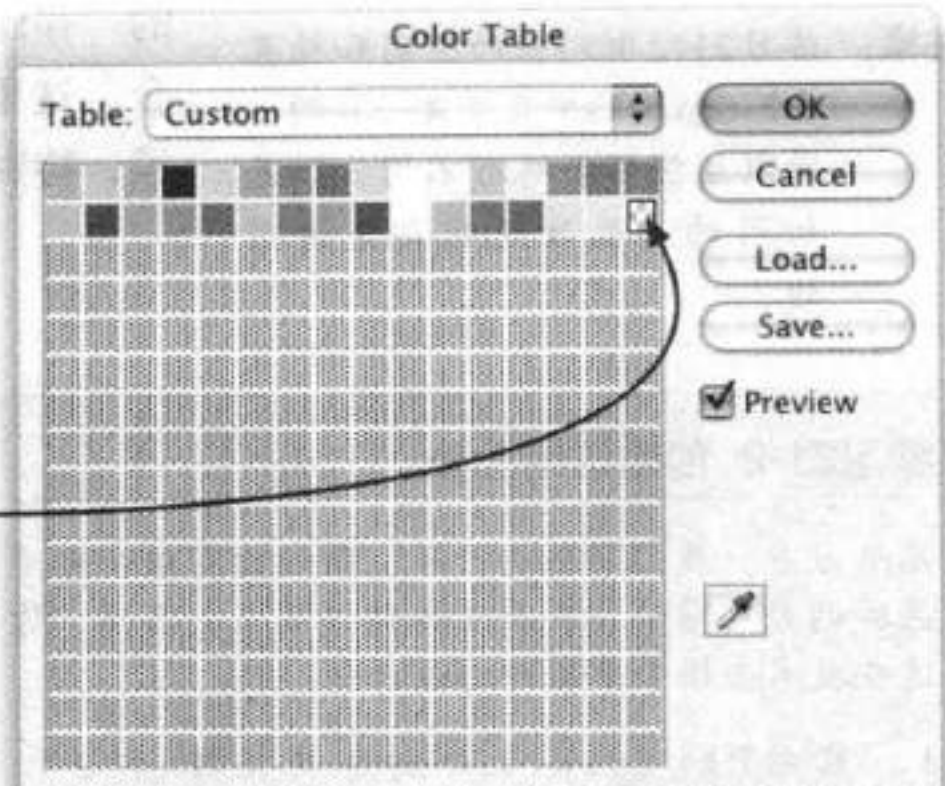
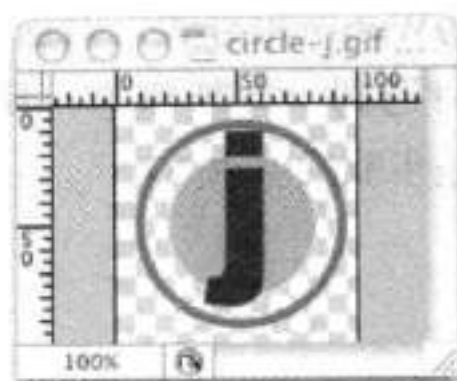
在Alpha（或Alpha通道）透明度中，像素可以完全透明、完全不透明，或者之间的254级不透明（总共256个不透明级别）。只有PNG支持Alpha透明度。使用Alpha透明度的PNG图像的优势在于可以与任何背景色和背景图案无缝融合，如图21-11所示。

本节中，你将熟悉每种透明度的工作机制，并学习如何使用Photoshop制作透明图像。

### 二元透明度的工作机制

记得GIF和PNG-8的图像的像素都储存在索引颜色表中。透明也被简单地当做一种独立的颜色，在颜色表中占一个位置。图21-20显示了Photoshop中简单的透明GIF的颜色表。颜色表中设置为透明的位置用棋（国际象

棋) 盘方格表示。当图像在浏览器上显示时, 那个位置对应的像素将完全透明。注意, 只有一个格子是透明的——其他所有像素颜色都是不透明的。

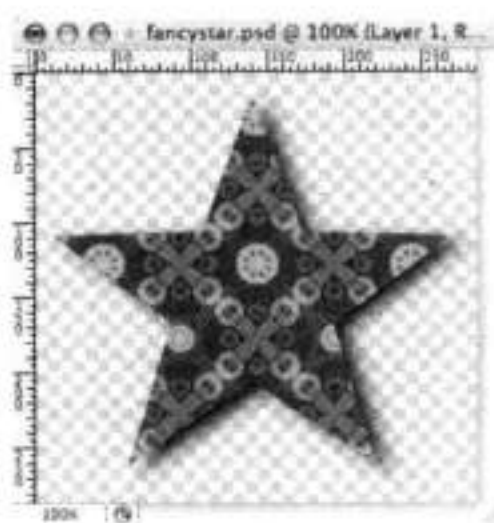


透明像素在索引颜色表中有一个位置。

图21-20: 在索引颜色表中, 透明被当做一种颜色

## Alpha透明度的工作机制

RGB图像 (如JPEG和PNG-24图像) 在各个通道中存储颜色, 一个红色, 一个绿色, 一个蓝色。PNG-24文件添加了另一个通道, 即Alpha通道, 用来存储透明度信息。在那个通道中, 每个像素可以显示256个值之一, 代表图像显示时透明度的256个级别。Alpha通道的黑色区域是透明的, 白色区域是不透明的, 灰色是中间值。我把它看做盖在图像上面的毯子, 告诉下面的每个像素它的透明度 (如图21-21所示)。



原先的透明图像

Alpha通道的黑色区域对应透明图像区域; 白色区域是不透明的; 灰色区域是中间值。

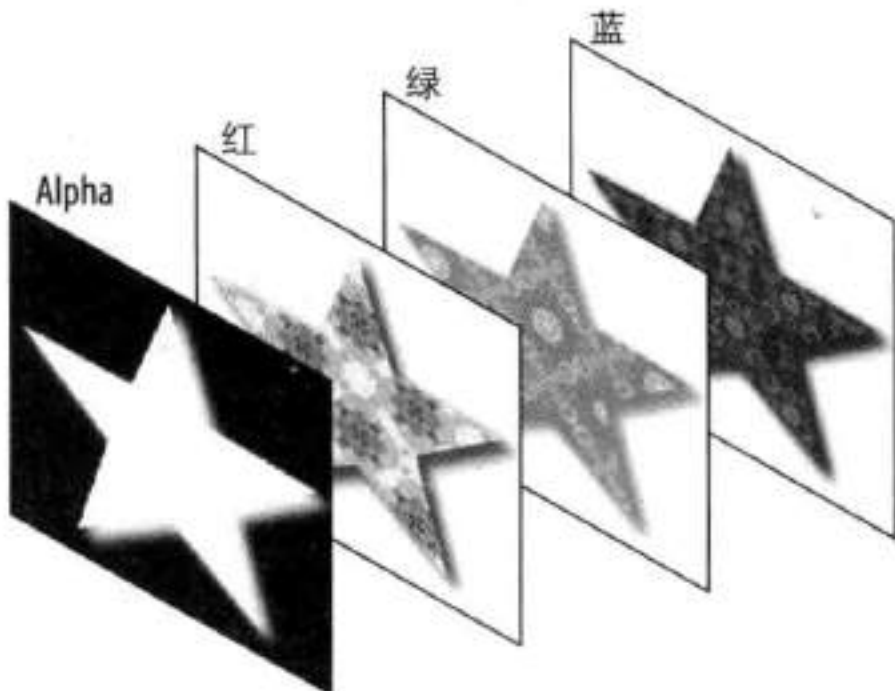


图21-21: 透明信息保存在24位PNG的单独通道中

## 制作透明的GIF和PNG图像

使图像部分透明的最简单的方法是, 当创建GIF或PNG版本的图像时, 从一开始就保存透明区域。Photoshop的“存储为Web所用格式”功能和Fireworks的优化面板正是完成这项任务的绝佳工具。

可以添加透明区域到图层的不透明图像中, 但很难与背景无缝融合。在本节后面将介绍使已有图像部分透明的处理过程。

但首先, 跟着练习21-2的步骤来做, 这个练习演示了如何使用Photoshop

### IE 6和Alpha透明度

值得一提的是, IE 6及其更早版本以完全不透明的alpha透明度来显示PNG图像。如果你需要支持IE 6, 有一个复杂的解决方法是使用微软专有的AlphaImageLoader筛选器, 由Michael Lovitt记录在这里: [www.alistapart.com/articles/pngopacity](http://www.alistapart.com/articles/pngopacity)。



**注意：**练习21-2中列出的原则和设置在Fireworks中几乎是一样的，所以虽然界面略有不同，但是相同的一般性说明都是适用的。

的“存储为Web所用格式”对话框，来保存透明区域并保证与背景完美融合。这里将遇到一些新概念，所以即使你不做这个练习，我也推荐你阅读，特别是步骤5、6和7。

## 练习21-2 创建透明图像

本练习中，我们将从头开始，所以你有机会创建有透明区域的分层图像。我将保持简单明了，当然，你可以将这些技术应用于更美妙的设计中。

1. 启动Photoshop并创建新文件（文件→新建……）。在“新建”对话框（如图21-22所示）有一些设置，可能使你不能正确创建透明的网页图像。

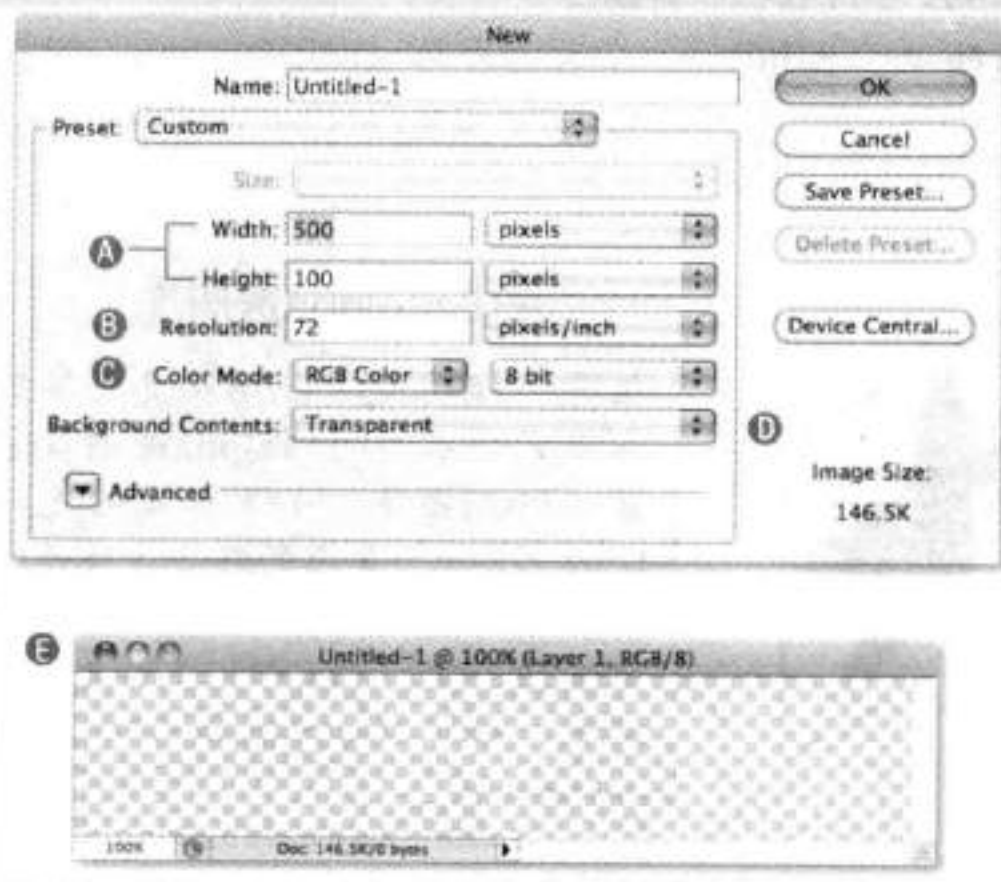


图21-22：使用透明背景创建一个新的图像

- 首先，设置新图像为500像素宽、100像素高，以匹配练习中的例子(A)。
- 设置分辨率为72像素每英寸(B)，这也是我制

作网页图像时使用的尺寸（虽然你也知道，这并不重要）。

- 确保颜色模式为RGB颜色，8位(C)。
- 最后，也是本练习中最重要的，从背景内容的选项中选中“透明”(D)。这个选项创建一个带透明背景的分层Photoshop文件。保存透明区域比在后面添加容易得多。透明区域（本例中为整个区域，因为我们还没有添加更多的图像内容）由灰色的棋盘图案表示(E)。

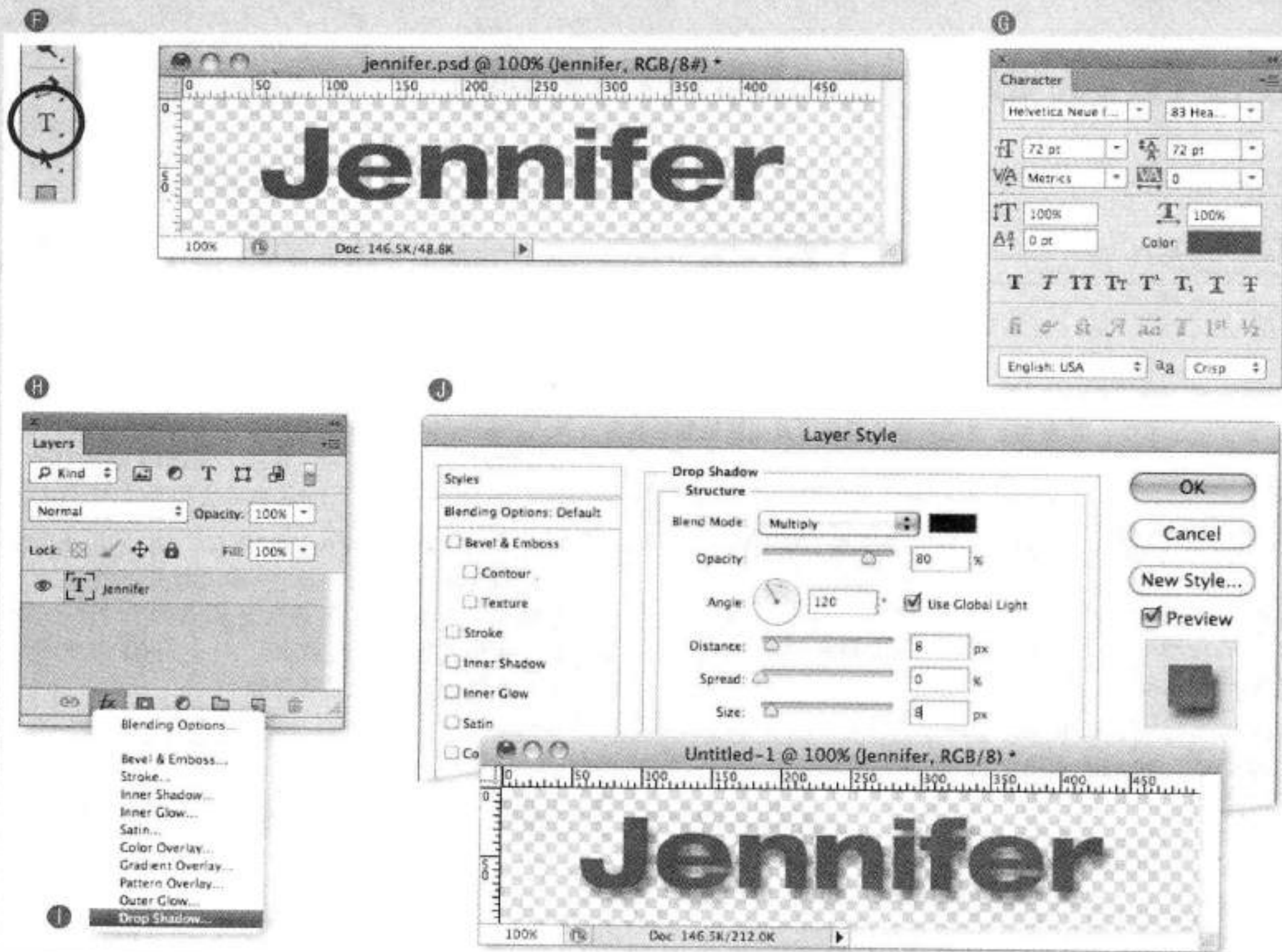
2. 现在添加一些文本，并添加文本投影（图21-23）。

- 使用文字工具(F)并打出你的名字。打开字符窗口(G)（窗口→字符）来改变字体外观。选中文本，选择粗体（有点矮胖的），并设置字体尺寸，大的能填满所有空间，如例子中所示。单击“颜色”旁边的样本颜色，使用拾色器为文本选择颜色，不要太淡，也不要太深。我使用中等的粉红色。
- 然后，给文本添加柔和的投影。如果还没有打开，就打开图层窗口(H)（窗口→图层）。你可以看到图层列表中包含的文本。通过单击图层窗口下方的“图层样式”按钮（看起来像FX），并选中“投影……”(I)，就可以添加投影。在图层样式对话框中(J)，你可以进行这些设置，但我推荐设置距离和大小至少为5，这样能让剩下部分的练习发挥最大的功用。完成后，单击“OK”按钮。



3. 保存图像为Photoshop文件，以此保存以后最易编辑的图层，如果需要编辑。命名我的文件为 *jennifer.psd* (使用 *.psd* 后缀)。漂亮的源图像保存之后，我们准备制作一些Web版本。

图21-23：使用柔和的投影添加文本



4. 使新文件保持打开状态，从文件菜单选择“存储为Web所用格式”。选择对话框上方的“四联 (4-Up)”选项卡，比较源图与其他几个版本的区别 (如图21-24所示)。注意，你的预览将显示在一格而不是一堆中。

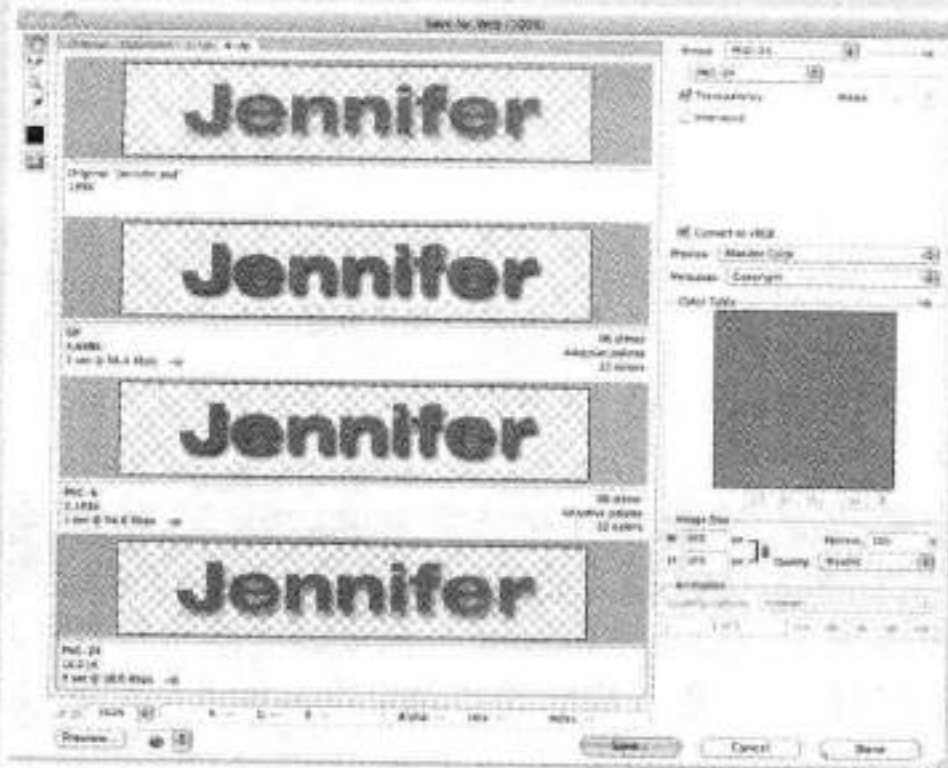
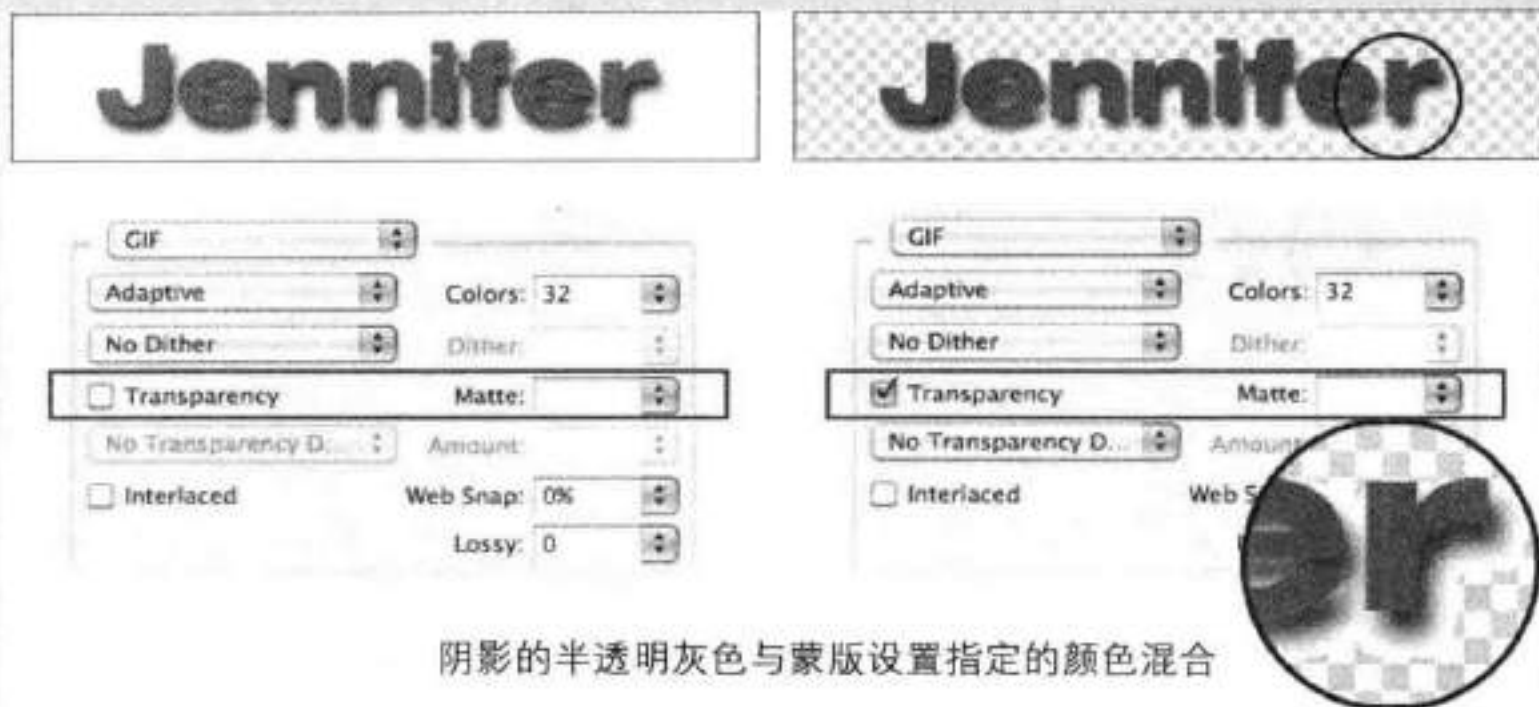


图21-24：“存储为Web所用格式”中的“4-Up”选项卡允许你比较同一图像的4个不同版本



5. 看看有透明和无透明的GIF图像。单击第二个预览来选中它，然后设置文件类型为GIF，设置颜色数为32。现在，关闭和打开“透明度”附近的复选标记（图21-25）。



阴影的半透明灰色与蒙版设置指定的颜色混合

都使用二元透明度。预览如图21-25所示。

7. 现在选择第四个预览，设置为PNG-24的文件类型，并开关“透明度”复选框（如图21-26所示）。

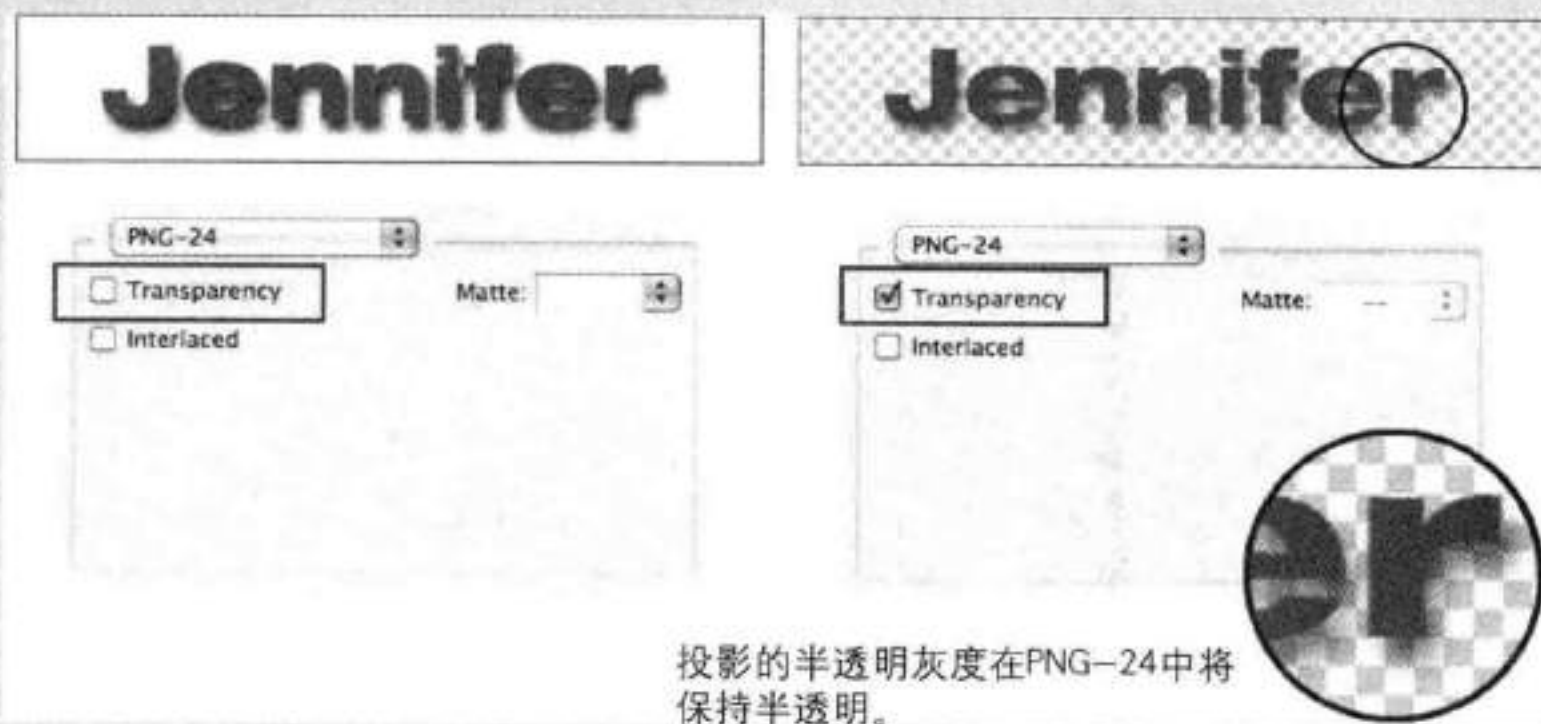
当未选中时（左图），蒙版颜色填充原图的透明区域。但是当透明度被选中（右图），棋盘图案将从投影混合区显露出来。所以，网页的背景也会这样。当透明度被选中时，蒙版工具不再有效，因为不需要指定这页的背景色……，使用Alpha透明度的PNG能与任何事物融合。

花些时间注意透明PNG-24的文件尺寸。我的将近10.6 KB。透明的GIF是5 KB，而透明的PNG-8只有3.3 KB。多出的文件尺寸是你为Alpha透明度功能支付的费用。

图21-25: GIF透明预览：左边为关闭透明，右边为选中透明

- 当透明度处于关闭状态（没有选中，如图左边）时，蒙版颜色将用于填充原图的透明区域。设置蒙版颜色为白色，与本例子匹配。
  - 当透明度处于打开状态（选中，如图右边）时，棋盘图案显示在图像的透明区域，表示网页的背景色和背景图案将显露出来。如果仔细看投影区域，你会看到灰色的影子与白的蒙版颜色混为一体。尝试改变蒙版颜色，并观察投影区域发生的变化。
6. 保持GIF预览不动并选择下一个预览。设置文件类型为PNG-8，尝试开关“透明度”复选框。不出所料，它的行为与GIF一样，因为这两种格式

8. 保存透明度打开的PNG-24图像，并以.png后缀命名（我的是jennifer.png）。再次打开“存储为Web所用格式”对话框，保存为透明度打开的GIF图像版本（确保蒙版设置为白色）。以.gif后缀命名文件。接下来将再次用到这些图像。



投影的半透明灰度在PNG-24中将保持半透明。

图21-26: PNG-24透明预览：左图为关闭透明，右图为选中透明

#### 设计提示

使透明GIF与背景无缝融合的诀窍，是使用网页背景色的RGB值（或者背景图像的主色）作为蒙版颜色。如果你的网页背景是多颜色的图像，或者其他很难匹配的，那么蒙版颜色选择比主背景色稍深的颜色。

### PNG-8的Alpha透明度

从技术上讲，多级别的透明度并不局限于24位PNG图像。PNG-8也能做到。不是使用Alpha通道，它们将不同的透明级别存储到索引颜色表的格子中。最终的文件尺寸比相同图像保存为使用Alpha通道的PNG-24的文件尺寸小一些。

本书写作时，只有Fireworks允许创建使用多级别透明度的PNG-8图像，并且缺乏浏览器支持。大多数浏览器的显示看起来就像在使用简单的二元透明度。而现在，这又是一个很酷的PNG特性，但是由于软件支持滞后，实际上还未使用。

### 避免“光晕”

既然有了一些透明图像，我将在白色背景的最小网页上试验。如果你想跟着做，打开文本编辑器并创建HTML文档，如下所示：

```
<!DOC TYPE html>
<html>
<head>
 <title>Transparency test</title>
 <style
 body {background-color: white;}
 </style>
</head>
<body>
 <p></p>
 <p></p>
</body>
</html>
```

当我在浏览器中打开文件时，在白色背景下，图像看起来差不多一样（图21-27左）。但是，我将网页背景色改为鳊蓝色（background-color: teal;），Alpha透明度和二元透明度的区别显而易见（右图）。

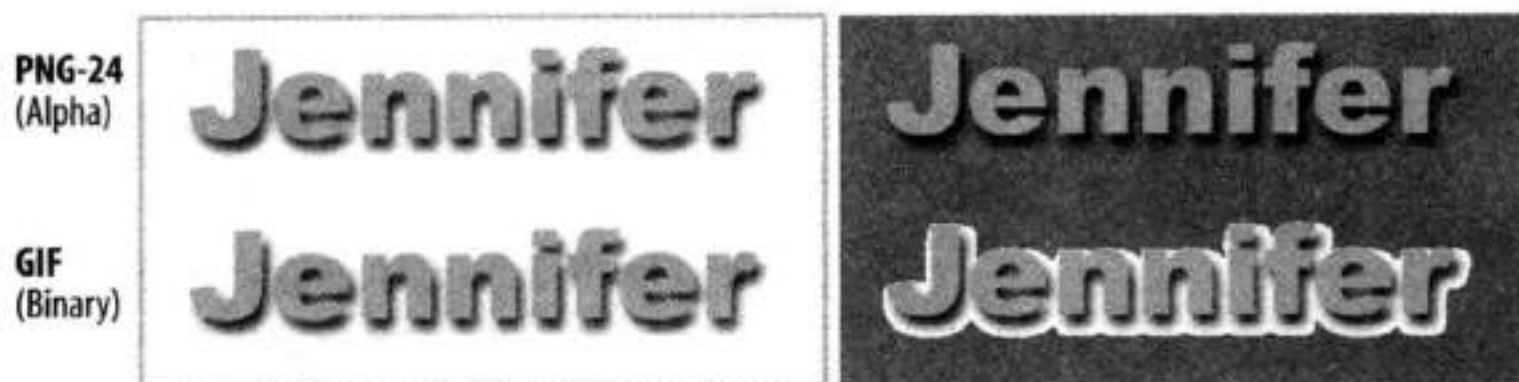


图21-27：当页面的背景色改变时，二元透明度和Alpha透明度的区别非常清楚

当背景色改变时，GIF图像不再与背景相配，导致通常称为“光晕（halo）”的丑陋边缘。光晕是消除锯齿边的结果，锯齿边与网页背景色之外的其他颜色融合。它们是二元透明度的潜在危险，无论是GIF还是PNG-8。

处理二元透明度和光晕时预防为主。如你所见，Photoshop和Fireworks

#### 术语

#### 抗锯齿（anti-aliasing）

抗锯齿是一种模糊处理方式，一般用于位图的圆形边缘，以便使得颜色之间的变化更圆滑。相比之下，锯齿边缘有步进楼梯样的边缘。抗锯齿文本和图像可以使你的图片看起来更专业。



### 蒙版的替代方案

如果你使用的图形工具没有蒙版功能，在该层底部的“栈”中创建一个新层，并填写你网页的背景颜色。当图像由于改变索引颜色而被抹平时，反走样边缘与适当的背景颜色相混合。导出为GIF或PNG格式时只要选择使背景颜色透明，你的图像就应该是免光晕的。

中的蒙版颜色功能使图像的边缘很容易与目标背景色融合。如果背景改变，你可以使用新的蒙版颜色重新导出GIF或PNG-8图像。如果你的工具没有蒙版设置，参考侧栏“蒙版的替代方案”寻找替代方案。

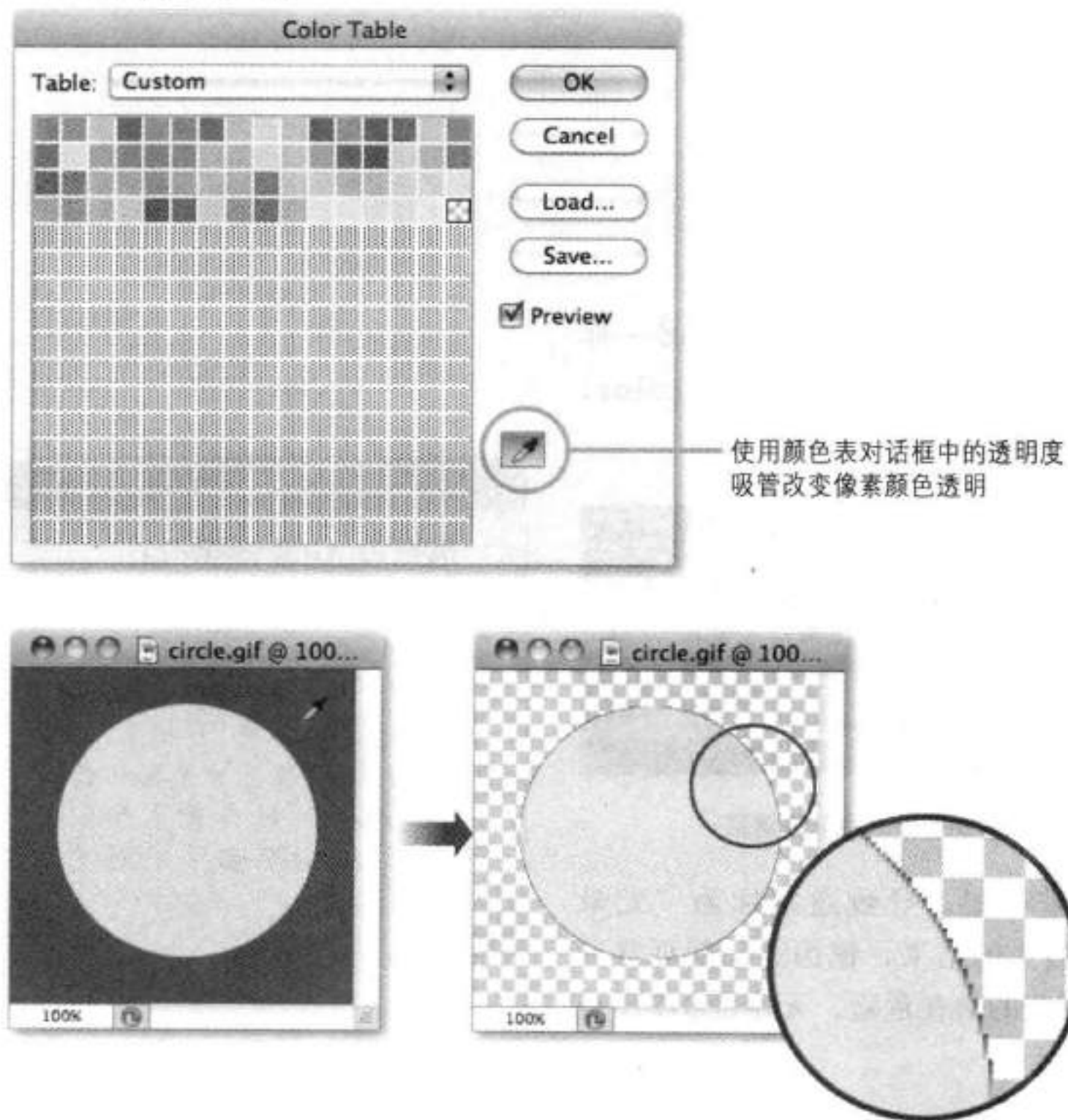
另一个选择是图像保存为使用多级透明的PNG-24。这样，你就不必担心背景色或背景图案，将来改变也不会有问题。当然，代价是需要下载的文件尺寸变大，而且不能在Internet Explorer 6及其更早的版本中使用。

### 添加透明度到图层合并的图像

可以添加透明区域到图像，而这个图像已经图层合并，并保存为GIF或PNG。图21-28中的GIF包含一个在紫色背景上的黄色圆，与纯紫色背景融为一体，但如果背景变成图案，GIF将成为明显的正方形。解决方案是使紫色区域透明以让背景显露。幸运的是，多数图像工具通过选择图像中的像素颜色，可以很容易做到，通常用一个吸管（eyedropper）工具，来处理要透明的图像。

图21-28：在Photoshop中使一个颜色透明

Photoshop (版本6或更高)



在Photoshop中，在颜色表对话框中找到透明度吸管（图像→模式→颜色表）。单击吸管，然后放在图像中的像素颜色上，它奇迹般地变透明（如图21-26所示）。要保存这个新的透明图像，使用前面演示过的“存储为Web所用格式”功能。

如果仔细观察，你可以看到，有一个边缘像素仍然对于紫色是反走样的，这意味着该图形除了紫色背景外，其他都效果良好。在其他背景颜色下，会有一个光晕。不幸的是，修整已经被抹平的图像的光晕的唯一办法，是在光晕处，逐一像素地删除反走样边缘。即使你消除了边缘，你可能会留下毫无吸引力的楼梯台阶边缘。你还可以使用图层蒙版来删除你希望是透明的区域，一定要擦除原始图像中的混合边缘。

如果你关注网站的专业外观，那么与其浪费时间修整光晕，还不如重

新从头开始创建图形，要注意避免它们。这是经常要保存多层文件的另一个原因。

## SVG简介

到目前为止，本章集中于可靠的网络位图格式，但也有我想你要熟悉的另一种有前途的选项。称可伸缩矢量图形（SVG）“有前途”有点误导，因为它自1999年以来就在规范中发展，它在2003年成为一个推荐格式，但幸好浏览器支持的改善，我们才最终能够充分利用它所提供的好处。

正如我在本章开头提到的，SVG是有点古怪。不像其他的网页图像格式，SVG是矢量图像格式，这意味着它包含用于绘制图形的工具，而不是像素网格。这使得SVG对于图标、标志、图表和其他轮廓图是一个不错的选择（图21-29）。虽然位图图像甚至视频可以嵌入SVG，但是它还不适用于摄影图像。

**注意：**在Fireworks中，吸管在“优化”面板的底部。“添加到透明”的工具使你可以选择不止一种像素颜色使之透明。这对于除去图像边缘周围不需要的颜色可能是有用的。

**注意：**两个很好的免费SVG艺术品的资源是“名词项目”（[thenounproject.com](http://thenounproject.com)）和“开放式剪贴画库”（[openclipart.org](http://openclipart.org)）。

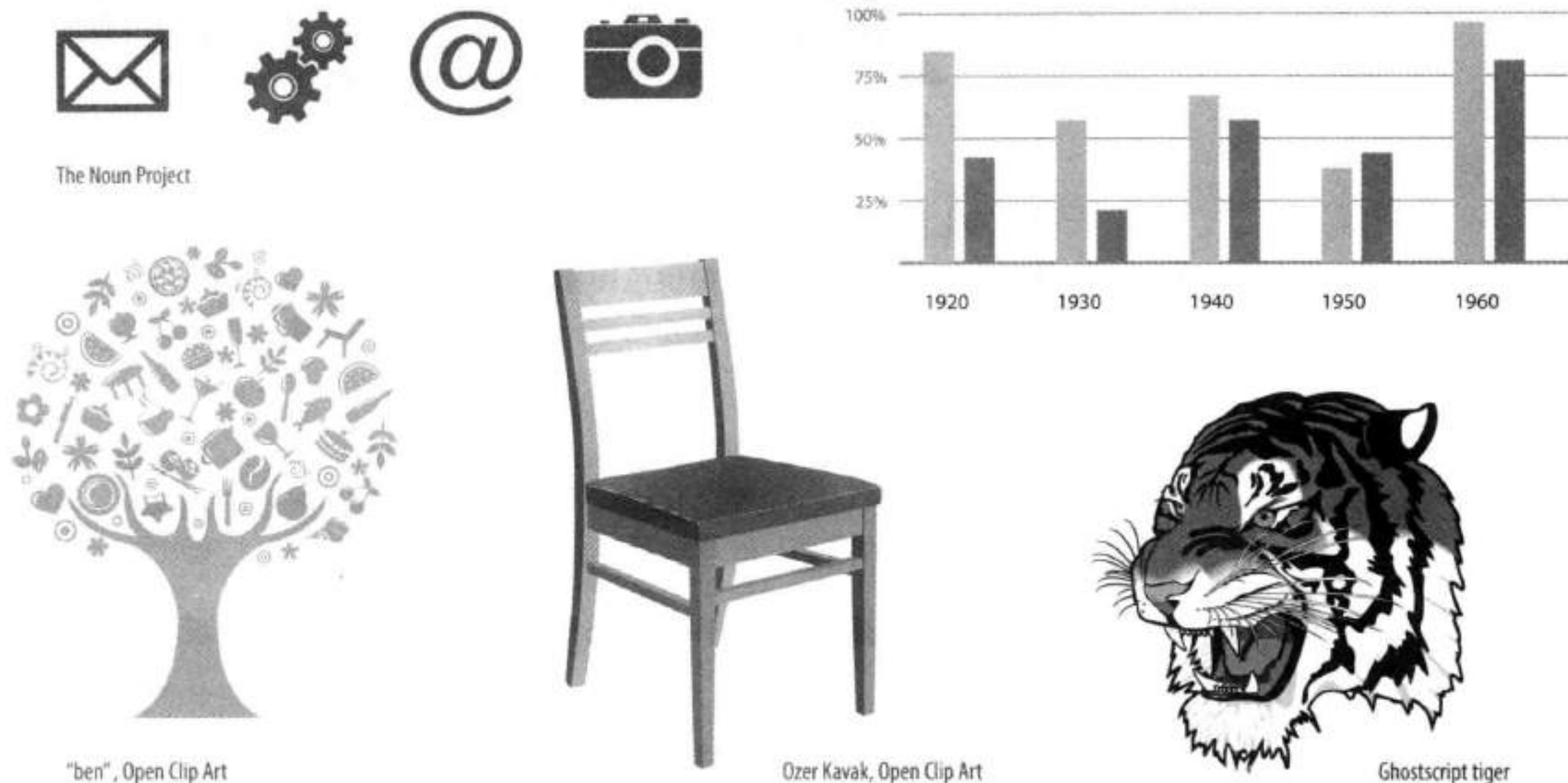


图21-29：SVG格式适用于线风格（line-style）的插画

矢量图像可以非常大或非常小，同时没有任何质量的变化（图21-30）。线条和文字保持清晰，无论是100像素还是10 000像素，使用位图图像来尝试做一下吧！既然网页设计和接口必须在所有尺度上的各种设备上运行，从智能手机到高密度的显示器和大屏幕电视，创建一个在所有的环境里看起来都不错的单一图像是一个史诗般的胜利。无处不在的SVG支



注意：Fireworks提供给你一个PNG-8图像的索引或Alpha透明度的选择。详细信息参见侧栏的“PNG-8的Alpha透明度”。

持，一定能够解决一些我们正面临的保持高密度显示器的图像分辨率的问题。

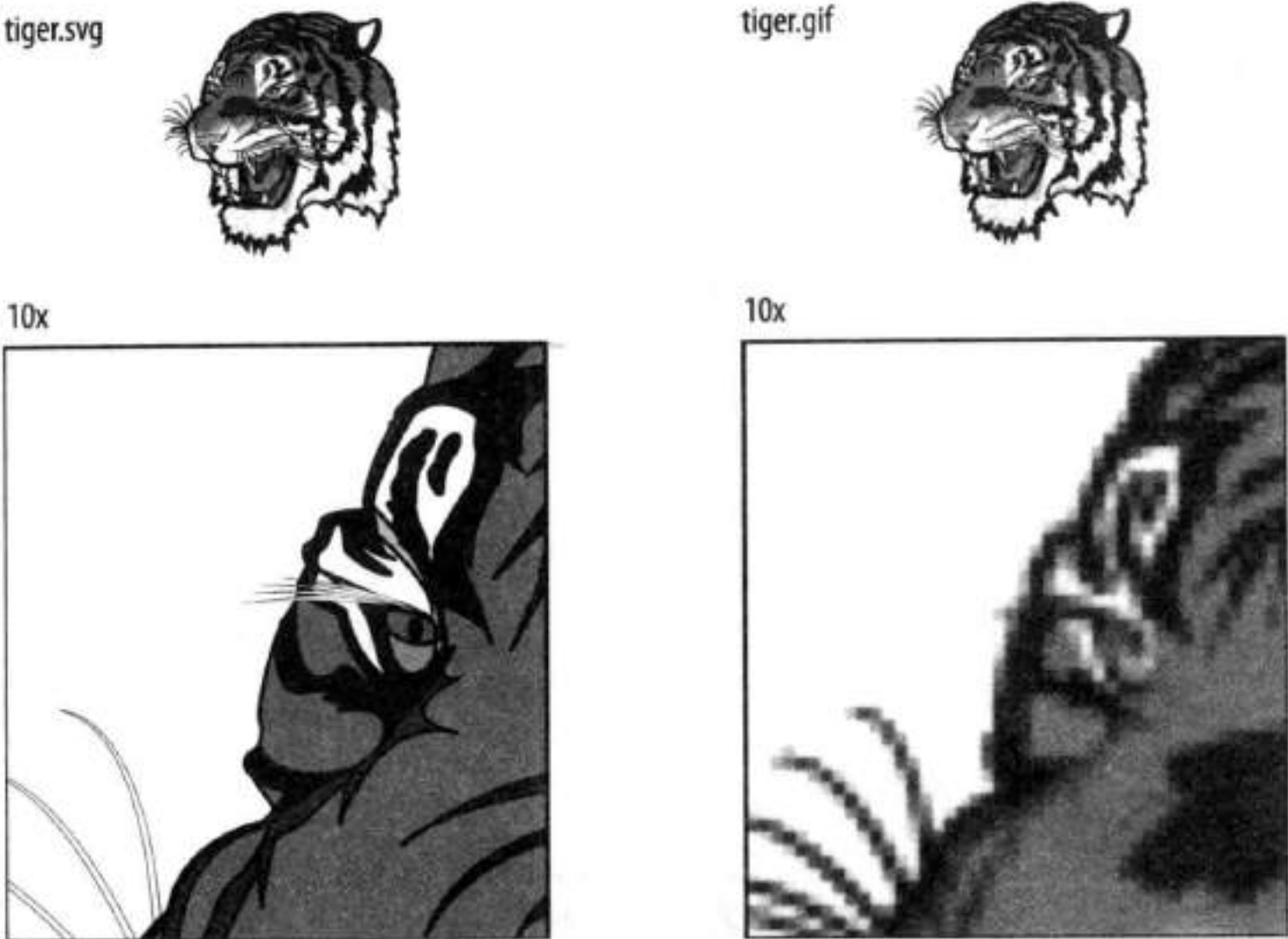


图21-30：矢量SVG图像放大时不会损失质量

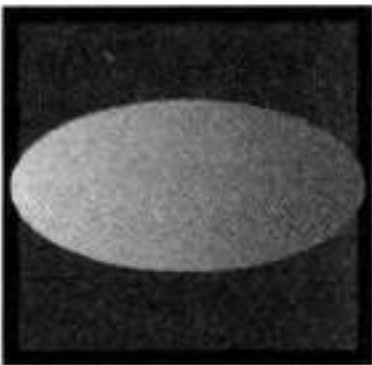
用XML绘图

真正设置SVG的是一种提供绘图指令的XML语言。位图图像被存储为难以理解的代码（你可以打开偷看一下），但SVG图像以一般人类可读的文本文件创建。

让我们看一个简单的例子和幕后的XML文本文件。图21-31展示了一张SVG图像——svg4u.svg，包含一个蓝色方块，渐变填充的椭圆和一些文字（我知道不漂亮，但它便于讲解）。

下面是生成该图像的文件。如果你仔细阅读，我想你会发现它是相当直观的。

图21-31：基本的SVG图像，svg4u.svg



```
<?xml version="1.0" encoding="utf-8"?>
<svg version="1.1"
 xmlns="http://www.w3.org/2000/svg"
 xmlns:xlink="http://www.w3.org/1999/xlink"
 width="450px"height="200px">
 <linearGradient id="yellowgrad">
 <stop offset="0" stop-color="#FFF200"/>
 <stop offset="1" stop-color="#F15A29"/>
 </linearGradient>
```

```

</linearGradient>

<rect x="50" y="50" width="100" height="100" fill="#4F5AA8"
 stroke="#000000" stroke-width="4" />

<ellipse cx="100" cy="100" rx="50" ry="25" fill="url(#yellowgrad)" />

<text x="175" y="150" fill="rgb(200,0,0)" font-family="Verdana"
 font-weight="bold" font-size="50">SVG 4 U!</text>
</svg>

```

让我们来看看svg4ru.svg是怎么回事。因为它是一个XML文件，所以它从一个XML声明开始。它也需要遵循XML语法，因此你会发现，所有的元素都是小写的，所有的属性都是在引号中，所有的元素都是关闭的（例如，<rect/>）。svg元素建立一个450×200像素的绘图区域。在SVG中，像素是默认的度量单位，这样你就不需要写“像素”。xmlns属性代表“XML名称空间”，它简单地标识了文档中使用的XML语言。

好了，现在是绘图部分。使用rect（长方形）元素来创建正方形，它的宽度和高度设定为100像素。你可以看到，属性用来提供位置、尺寸、填充颜色和笔触样式等。除了rect元素，SVG还包含了circle、ellipse、line、polygon和polyline等绘线和形状元素。

在我们的例子中，ellipse元素出现在方形的中心，它由文档中的线性梯度元素linearGradient创造的“黄色”渐变来填充。图像中的文本被包含进一个text元素中，并且具有从CSS获得语法属性的风格。虽然在本示例中未示出，但是使用image标记在SVG图形中放置位图仍然是可能的。

当然，相比于我在这里讨论的，还有更多的SVG语言，但现在你应该对于它是如何起作用的有一个大致了解。

## SVG工具

从技术上讲，所有你需要创建的SVG图形都是一个文本编辑器（加上天才的可视化能力，以及英勇的耐心！），但是如果有一个图形程序可以做这些你会更快乐。幸运的是，在Adobe Illustrator中，当你保存一张绘图或者一个SVG文件时，你可以从“格式”菜单下选择“SVG（svg）”。如果你没有Illustrator，请尝试下载Inkscape（图21-31）图像编辑器，它是专门为SVG（[inkscape.org](http://inkscape.org)）而制作的。它可用于Windows、Mac和Linux。这需要一点时间来适应，但你不得不在价格面前屈从（它是免费的）。



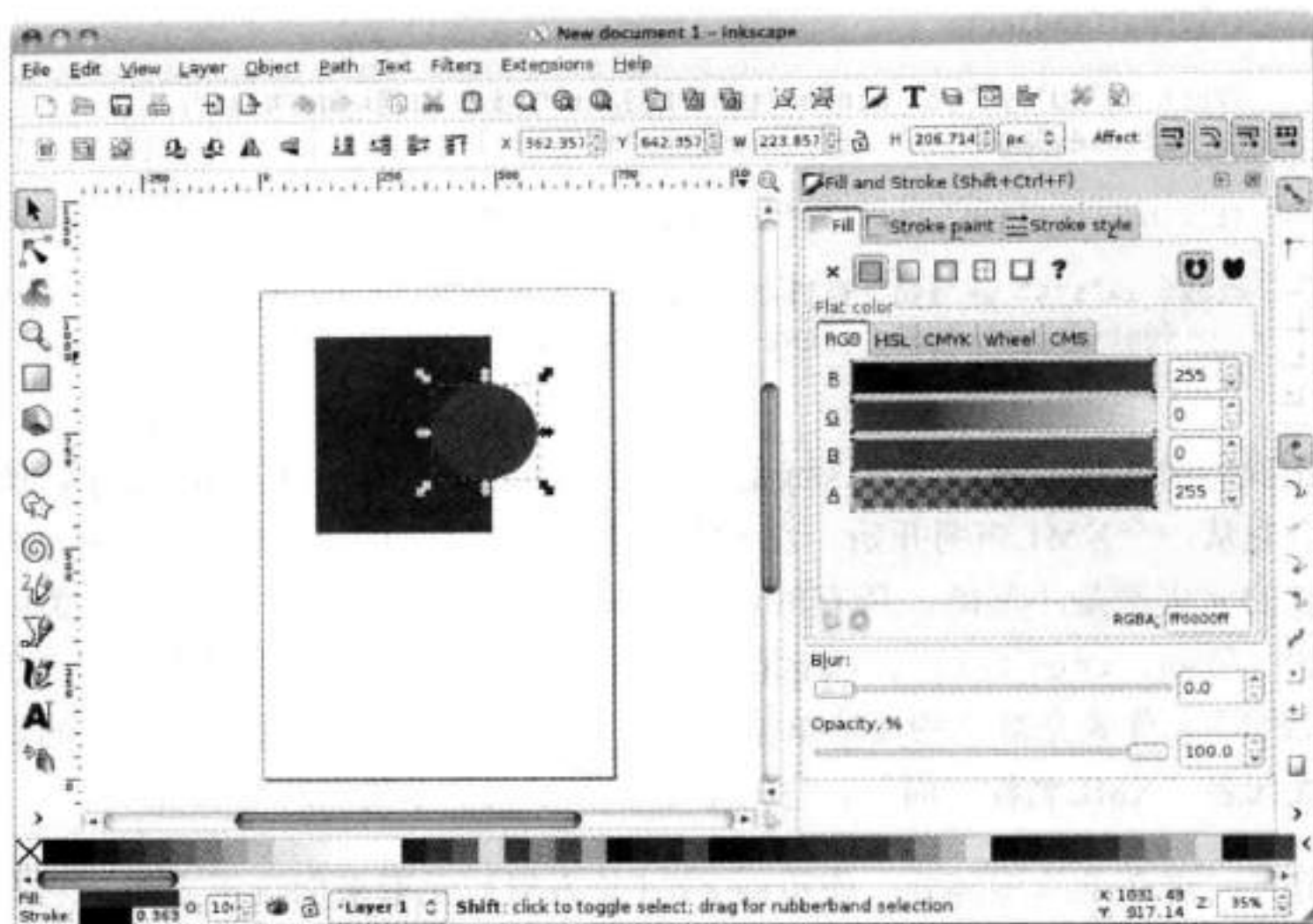


图21-32: Inkscape, 一个开源的SVG编辑器

## 在网页上添加SVG

使用`object`、`embed`或`iframe`元素可以把SVG图像添加到网页中。HTML 5允许`svg`元素直接添加到内嵌的未包含元素的部分HTML文档中。W3Schools页面中有关于各种SVG嵌入选项以及它们各自的优缺点的完善的小总结：[www.w3schools.com/svg/svg\\_inhtml.asp](http://www.w3schools.com/svg/svg_inhtml.asp)。这些东西变化很快，所以我建议你做一些搜索，了解最新的情况。

截至本文撰写时，浏览器可以很好地支持指向外部`.svg`文件的`object`元素，所以在本例中，我将使用这种方法。

```
<!DOCTYPE html>
<html>
<head><title>SVG 4 U</title></head>
<body>
 <object width="450" height="200" type="image/svg+xml"
 data="svg4u.svg"></object>
 <p>Give SVG a try and see what it can do.</p>
</body>
</html>
```

`object`要求以`width`和`height`的属性来掌控图像的空间。如果空间实在是太小了，图像将被裁剪。而要避免混淆，建议你包括文件类型（`image/svg+xml`）以让浏览器知道该做什么。最后，`data`属性点指向`.svg`文件本身。

## 但是等等，还有更多

“SVG4 U”的示例展示了用于静态图的SVG，但SVG还可以提供更多的用途。

### 动画

SVG有变换和过渡功能（同CSS3），所以SVG图像的任何部分通过单独使用SVG语法都可以有动画效果。下面的代码会使一个黑色矩形在两秒钟的循环内伸缩50%。

```
<rect width="150" height="150" fill="black">
 <animate attributeName="width" values="0%;50%;0%" dur="2s"
 repeatCount="indefinite" />
 <animate attributeName="height" values="0%;50%;0%" dur="2s"
 repeatCount="indefinite" />
</rect>
```

### 可编写脚本

因为SVG文件所有的部分都是在XML内，并且是DOM（文档对象结构模型）的一部分，你可以使用JavaScript来给SVG图纸添加动画行为。你也可以根据用户输入的实时内容将JavaScript应用于动态绘图，比如，生成反映输入表单中的值的图表或图形。它是很美妙的东西，但是超出了本章范围。

### 样式化

我并没有在本章中包含此内容，你也可以使用CSS来改变SVG图像元素的外观。

### 可访问

SVG图像的内容在XML文件中可获得，因此它可能比canvas元素更可访问，它以抽象的像素网格存在。你还可以在svg元素中添加标题和描述。

## 浏览器支持

就在我们感到一切顺利的时候，我不得不提“浏览器的支持”问题！事实上，这个消息并不全是坏事。当我写本书时，每一种主流浏览器的当前版本，无论台式的还是移动的，都对SVG图像有基本的支持。当你阅读本书时，情况可能已经有所改善了。最新的统计资料，参见“我能使用”网上的SVG名单（[caniuse.com/#cats=SVG](http://caniuse.com/#cats=SVG)）。

因此，SVG的未来一片光明。但我们仍然有不可忽视的浏览器旧版本，特别是Internet Explorer 8及更早的版本。幸运的是，也有解决方法，如SVGWeb JavaScript库（[code.google.com/p/svgweb/](http://code.google.com/p/svgweb/)），95%的浏览器都可编写SVG脚本。在移动领域，预测支持SVG的浏览器是很难的，因为设备制造商可以关闭本来支持的特定设备功能，但Android 2.3以及更早版本

### SVG与Canvas

在第10章中讲述了创建二维空间的HTML 5 canvas元素和API，它们可以在网页上动态绘制。不同的是，绘制SVG图像使用的是结构化的标记语言，而canvas是用JavaScript命令来绘制的。两者都可以包含图像、视频、动画和实时动态更新。

canvas更适用于快速重绘（毕竟它只是像素），这使得它更适合于游戏、图像编辑以及把图像保存为位图格式。SVG在脚本、动画和易用性上有优势，然而，复杂的文件比canvas元素需要更多的处理能力。



### 练习21-3 玩转SVG

SVG文件使用起来很有趣。在本章的练习材料中包含了`svg4u.svg`文件以及相应的HTML文件，所以你可以感受到它是如何运作的。

1. 在支持SVG的浏览器中打开`svgtest.html`文件（最新版本的Chrome、Safari或Firefox都行）。你应该会看到图21-30所示的（很丑陋的）SVG4U。
2. 在文本编辑器中打开SVG文件。记事本或文本都很好，只要文件保持ASCII格式，同时没有被样式化。
3. 试着通过调整x和y坐标来移动片段。试着调整尺寸，更改填充颜色。每次查看你的更改都要在浏览器中刷新`svgtest.html`。我第一次做的时候感到很愉快。
4. 如果你喜欢冒险，打开“SVG Primer for Today's Browsers”（“进一步阅读”部分已列出），并尝试其他的形状和线条。或者你可以尝试在Illustrator中创建一个简单的图形，将其保存为.svg文件。在文本编辑器中打开它，并查看代码。有很多额外的东西要筛选，但你应该能够看懂，并可以编辑基本形状。

的手机和平板电脑不支持SVG。

如果你选择用某些类型的图形来探索SVG，一定要在各种各样的设备上测试（不只是你的iPhone），并提供有用的备用方案，即使它只是描述性文字，以备无法显示SVG。

### 进一步阅读

很显然，在本章当中，我只能触及可伸缩矢量图形问题的表面。如果你发现了在网站上利用它们的动机和机会，你将必须学习更多。除了自己搜索SVG的更新信息外，我建议阅读以下资源以得到更好的了解。

- Kurt Cagle (O'Reilly出版) 的《HTML 5 Graphics with SVG, CSS》。
- Shelley Powers (O'Reilly出版) 的《Painting the Web》。这本书虽然已经有年头了（写于2008年），SVG这一章提供了一个关于可以做什么格式的概述，并且对于其他的图片格式、CSS以及在网络上可视的所有东西，都有很多深层次的信息。
- “An SVG Primer for Today's Browsers” ([www.w3.org/Graphics/SVG/IG/resources/svgprimer.html](http://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html))。本文为你提供了SVG图形的全面教程，但它内容可能有点多。
- “SVG Examples” ([www.w3schools.com/svg/svg\\_examples.asp](http://www.w3schools.com/svg/svg_examples.asp)) 显示了很多图形的代码和特殊效果。

### 小结

在本章中，我们涵盖了很多内容！如果我完成了我的任务，那么现在你应该在网页图像方面打下了坚实的基础，包括：到哪里寻找图像、保存为哪种文件格式，以及如何缩放以满足Web需求。你也知道二元透明度与Alpha透明度之间的区别，以及如何制作与网页背景融为一体的图像。你甚至了解了一些SVG词汇。

在第22章，我们将进入图像制作的下一个阶段，探索使图像尽可能快速下载的所有方法。但首先，让我们做一个小测验。

### 自我测验

回答下面的问题来看你是否掌握了网页图像。这些问题的答案在附录A中。

1. 使用版权受控图片的主要好处是什么？

2. ppi代表什么意思?
3. 什么是“索引颜色”?它使用什么文件格式?
4. 8位图形色彩表中有多少种颜色?如果你懂一点数学,计算出5位图形颜色的最大数量。
5. 指出GIF可以做而JPEG不可以做的两件事。
6. 指出GIF可以做而PNG不可以做的一件事。
7. 指出PNG可以做而GIF不可以做的一件事。
8. JPEG有损压缩是不断累积的,这意味着什么?为什么知道这点很重要?
9. 二元透明度和Alpha透明度之间的区别是什么?
10. 为图21-33中的每幅图像选择最佳的图形文件格式。通过观察它们,你应该能够做出决定,并解释你的选择。有些图像可能有多个选项。



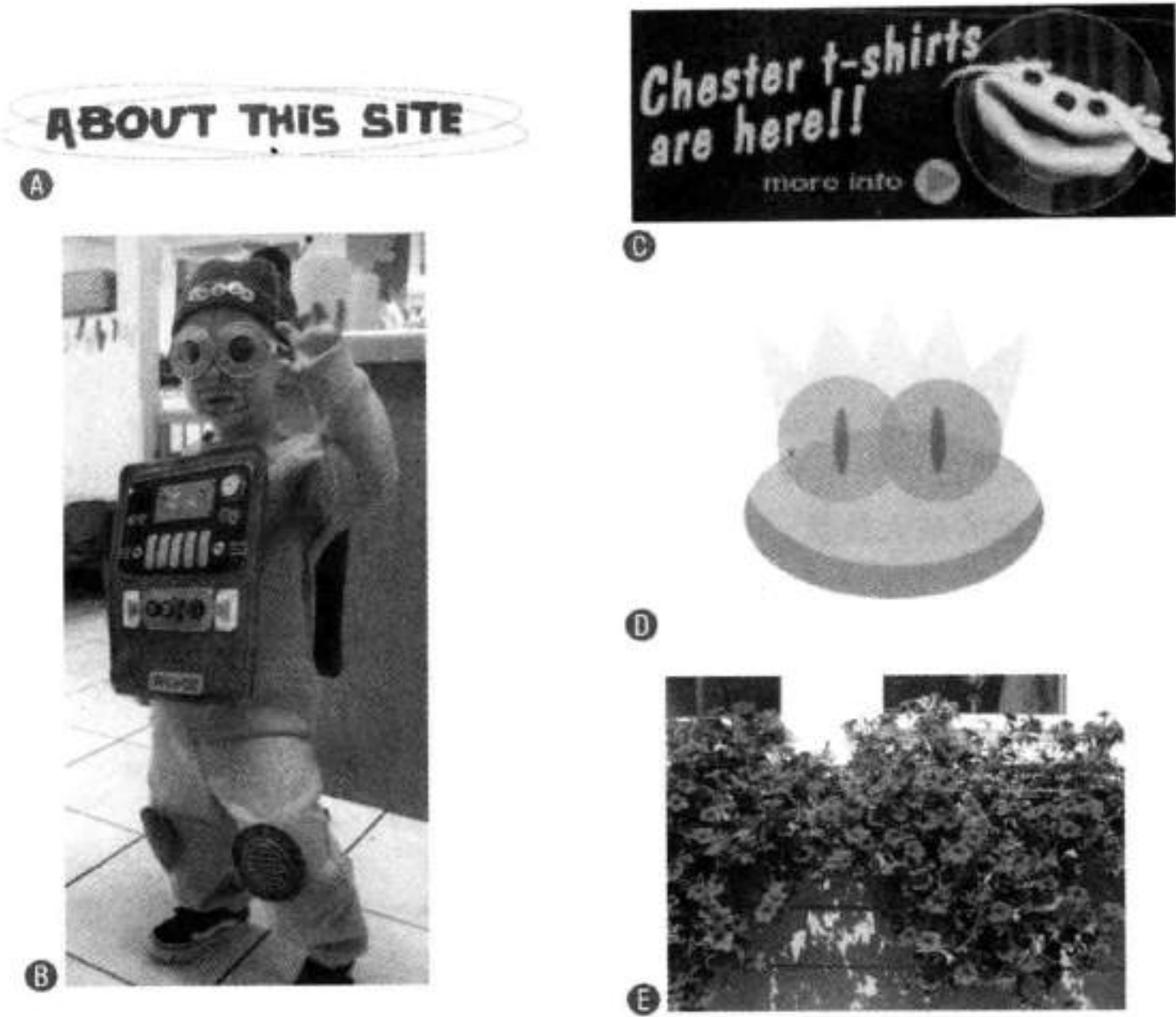


图21-33：为每个图像选择最佳的格式

# 第22章 精简Web图像

因为网页是在网络上发布的，所以为了到达终端用户，它需要以小数据包的形式迅速地穿过网线。大量数据将需要很长时间才能到达。猜猜看，标准网页的哪部分装了最多的数据——正是图像。

从而Web出现了与图像的冲突关系。一方面，图像使网页比纯文本更有趣，显示图像的能力是Web成功的要素之一。另一方面，图像也在考验通过低速网络连接来上网的用户的耐心，并消耗了移动流量套餐计划（见“注意”）。

本章的内容包括，在图像质量可接受的情况下，使网页图像文件尽可能小（以优化著称的过程）的可用策略和工具。我希望在第3章中，网站性能优化的重要性给你留下了深刻印象。除了减少你的页面对服务器的请求之外，减少图像的总文件大小以使页面尽可能快地显示也是一个强大的方法。学习如何从你所创建的图像中榨取每一个不必要的字节非常值得付出额外的努力。

---

**注意：**减轻移动设备负载的策略之一是提供独立的、针对小屏幕设备的更小的图像。这个过程称为自适应图像，在第18章中已经简要讨论过了。

---

## 通用图像优化策略

想象一下，你正在为一个大客户设计横幅广告，他们会告诉你所有横幅图形都有15K的限制（文件不能超过15KB）。这种情况在商务中是相当普遍的，所以为了制作出目标图像，你需要利用一些技巧。而这就是本章的相关内容。

不管图像或文件类型，有一些限制文件尺寸的基本策略需要记住。用最明朗的话，它们是：

### 本章内容

为什么应该优化图像

通用优化策略

优化GIF图像

优化JPEG图像

优化PNG图像

优化到指定文件大小



### 限制尺寸

虽然相当明显，但保持小文件尺寸的最容易的方法，是限制图像本身的尺寸。没有任何其他办法；不要让图像比需要的大。如图22-1所示，通过简单地去掉图像中多余的空间，就能减少3KB的文件尺寸（23%）。

### 复用和循环

如果你在同一网站重复使用同一幅图像，最好只创建一幅图像，并在需要的地方重复指向这幅图。这允许浏览器利用缓存的优势，避免多余的下载。缓存概念在第7章的侧栏“利用缓存”中解释过。

### 压缩设计

使文件尽可能小的一个最大的策略是，有效的压缩设计。例如，因为你知道GIF擅长压缩平色，所以平色可用的时候，不要使用渐变混色来设计GIF图像。类似地，因为JPEG喜欢柔和过渡和非硬边，所以可以试着模糊那些将保存为JPEG格式的图像。这些策略在本章后面会详细讨论。

### 使用网页图像工具

如果你知道你将会做很多Web制作工作，那么值得购买图像编辑软件，如Adobe Photoshop或Fireworks。在第21章讲述了Photoshop中的“存储为Web所用格式”对话框，以及Fireworks中的优化和预览面板，它们都提供了制作网页图像的快捷方法。本章将充分利用保持文件尺寸尽可能小的设置。

在进行优化设置的时候，这两个工具都允许你预览最终图像和各自的文件尺寸，所以你可以调整设置并立即见效。选项因文件类型而异，所以我将挨个解释每种格式，从我们熟悉的GIF开始。

图22-1：你可以简单地去掉图像中多余的空间来减小文件尺寸

600 x 200 pixels (13 KB)



500 x 136 pixels (10 KB)



### 在线图像优化器

如果你没有Fireworks或Photoshop, 你可以使用这里列出的免费在线图像优化工具。它们并不提供Web图形工具中的那种设置, 但它们肯定比完全没有优化办法要好得多。

**Smush.it** ([www.smushit.com](http://www.smushit.com))。Smush.it 使用特定图像格式的优化技术, 删除图形文件中不必要的字节。它是一种无损的工具, 这意味着它可以优化图像, 而不改变它们的外观或视觉质量。我发现, 虽然一些图像已经在Photoshop中优化了, 但是这个工具还有办法稍微减少图像文件的大小。这是一种很棒的资源。

**Dynamic Drive Online Image Optimizer** ([tools.dynamicdrive.com/imageoptimizer](http://tools.dynamicdrive.com/imageoptimizer))。这是另一个在线工具, 它基于更为积极的优化设置返回优化版本。这不是一种无损工具, 所以你需要从优化图像中进行选择, 以确保找到的图像质量还可接受。

如果这些对你来说还不够, 可以登录PunyPNG ([punypng.com](http://punypng.com)) 和ImageOptim ([imageoptim.com](http://imageoptim.com))。

## 优化GIF图像

当优化GIF图像时, 记住这点很有用: GIF压缩是通过压缩相同像素颜色的字符串来实现的。很多优化策略, 通过创建更大面积的纯色作为压缩方案。

对GIF文件尺寸保持控制的通用方法是:

- 减少图像的颜色数 (位数)
- 减小图像的仿色
- 应用“有损的”滤镜
- 使用平色设计

本节使用Photoshop的“存储为Web所用格式”和Fireworks的优化面板作为操作台 (如图22-2所示)。如果一个功能是这些工具特有的, 我会进行提示; 否则, 这里展示的方法, 多数图像编辑软件都能做到。

### 减少颜色数

减小GIF文件尺寸的最有效方法, 也是优化旅程中的第一站, 是减少图像中的颜色数。

虽然GIF图像包含最多256种颜色, 但没有规则说它们必须有256种。事实

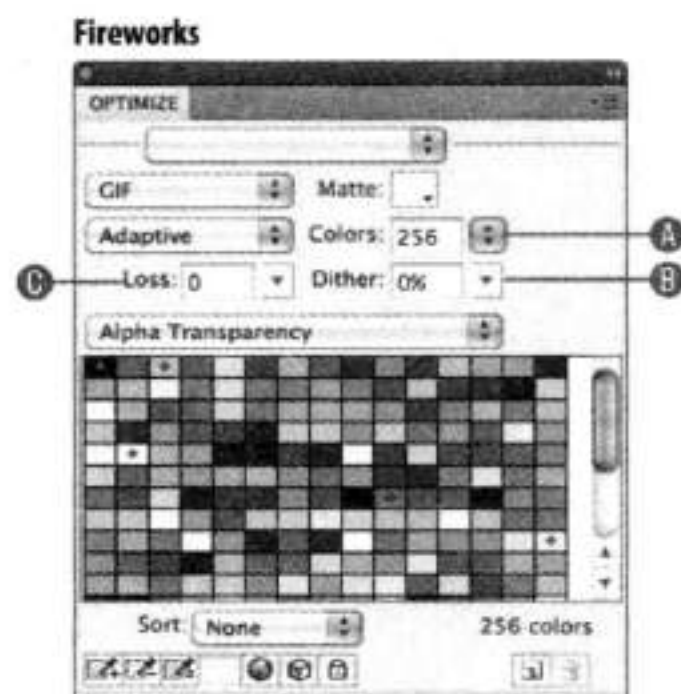
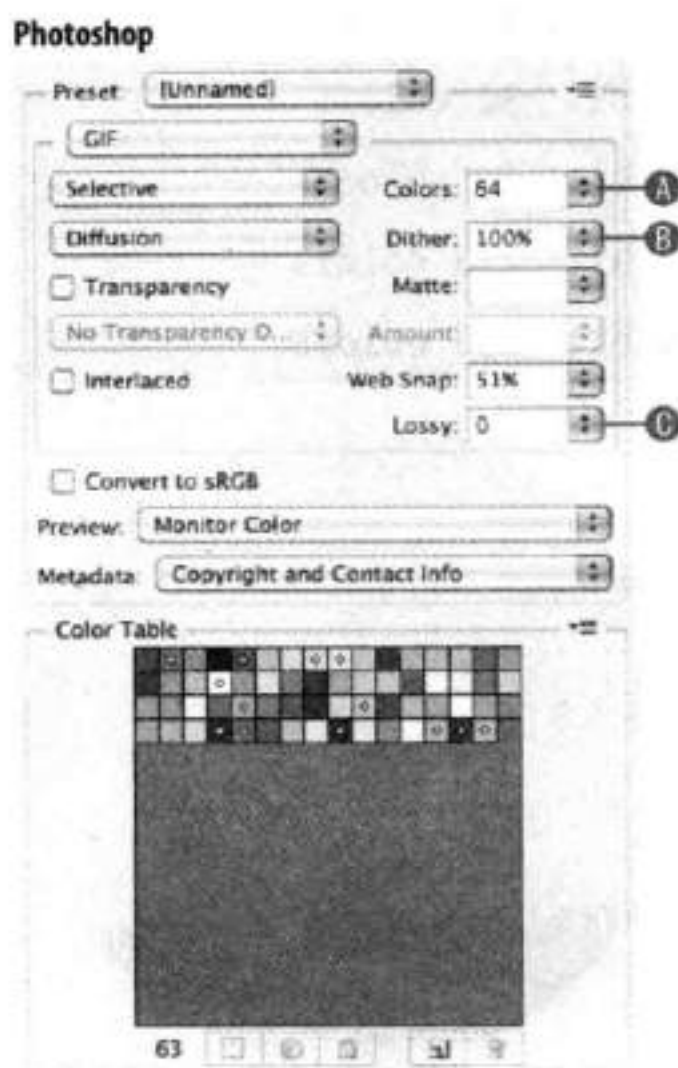


图22-2: Photoshop和Fireworks中的GIF优化选项



### 位深

位深是用来表示一个图片可以包含的最大色彩数的方式。下表展示了每种位深能表示的色彩数：

1-bit	2 colors
2-bit	4 colors
3-bit	8 colors
4-bit	16 colors
5-bit	32 colors
6-bit	64 colors
7-bit	128 colors
8-bit	256 colors



256 colors: 21 KB



64 colors: 13 KB



8 colors: 6 KB

图22-3：减少图像中的颜色数，从而减小文件尺寸

你会惊奇地发现，很多使用32像素颜色（5位）的图像看起来很好。这通常是颜色减少的起点，如果必要，那么可以使用更高位。一些颜色类型比其他有更少色板的类型更好，但是作为通常规则，颜色越少，文件越小。

**注意：** 当有大面积的平色时，真实尺寸会省去应该省的一部分。记住，即使你的图像有8种像素颜色，如果有很多混色、渐变和细节，那么就算有很多颜色减少，你也看不到你期望的文件尺寸减小。

### 减小仿色

当图像中的颜色减少到特殊的色板时，不在色板中的颜色会通过仿色来得到类似的颜色。仿色是一种斑点图案，在色板颜色混合来模仿不可用的颜色时产生。

在照片图像中，仿色不是问题，而且还是有好处的；但是，平色中的仿色令人迷惑，不受欢迎。在优化方面，仿色是不受欢迎的，因为斑点破坏了其他颜色的平滑区域。这些偏离的斑点阻碍了GIF压缩，并导致更大的文件尺寸。

削减GIF多余字节的一种方法是限制仿色数量。几乎所有的GIF创建工具都允许打开或关闭仿色。通过允许在滑动条（见图22-2⑥）上设定仿色的具体数量，使Photoshop和Fireworks的功能进一步加强。你甚至可以预览仿色设置的结果，所以你可以决定：图像质量降到哪个程度才不值得如此在意尺寸（图22-4）。在颜色平滑过渡的图像中，关闭仿色会导致不可接受的条带和斑点。

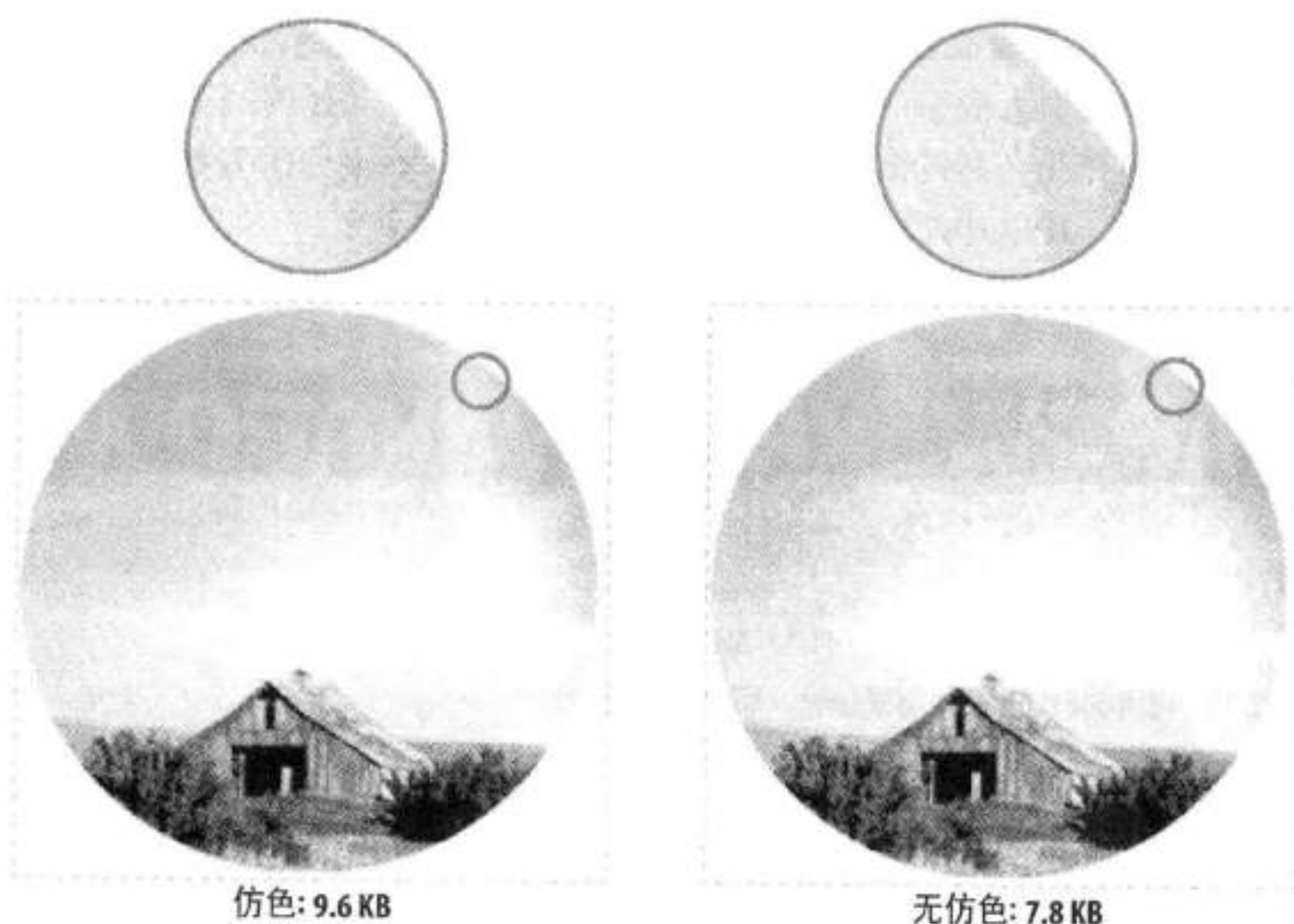


图22-4：关闭或减少仿色可以减少文件尺寸。两个图像都有32像素颜色，并且使用一个适合的色板

## 使用有损滤镜

最后使用“存储为Web所用格式”对话框中的最终优化设置是有损的（图22-2③）。在Fireworks中，它称为“损失”。此设置允许程序有选择地扔掉数据，以减少文件大小。设定越高，会有越多的数据被丢弃。根据图像，你可以应用损耗值为5%~20%，这样不会严重降低图像质量。图22-5显示了将Photoshop的有损设置应用于谷仓图像的结果。在较高的设置下，图像往往看起来是漂浮的，并且是四分五裂的。

**注意：**如果你注意，你可能在想，本节中的照片应该保存为JPEG，而不是GIF。完全正确。通常，不会把照片保存为GIF，但在本节中将它用作例子，因为它比平色图像更能让人注意到优化的效果。

### 图像提示

#### 找到“最佳点”

你将看到，为给定图像寻找最好的优化方案需要依次调整所有这些属性（位深、仿色和损耗），直到获得最小文件尺寸和最好图像质量。它需要花费时间来练习，但渐渐地，你将为每幅图找到“最佳点”。



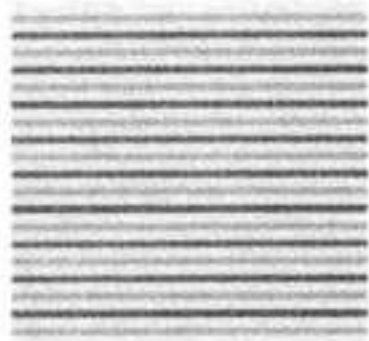


Lossy set to 0%: 13.2 KB

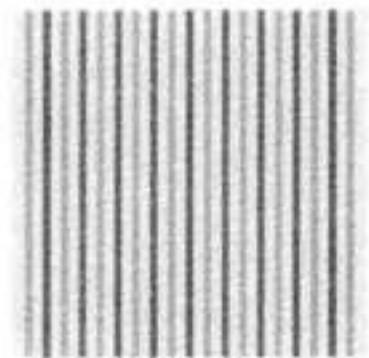


Lossy set to 25%: 7.5 KB

图22-5: Photoshop中应用有损设置的文件尺寸和不应用有损设置的图像尺寸



280 bytes



585 bytes

图22-7: 使用水平色带设计的GIF比垂直色带压缩效率更高

这个技术在连续色调图像中表现最好（但无论如何，全部是连续色调的图像应当保存为JPEG）。你可以尝试调整混杂着平色和照片内容的图像的损耗。

## GIF压缩设计

既然你已经看到高的位深和仿色是如何使文件尺寸膨胀的，那你应该知道下一个提示的背景了。在你进行优化设置之前，你应该预知到，优化图像首先要压缩好。

### 保持平色

我发现，作为一个Web设计师，我已经改变了绘画风格，以适应媒介。在我可能使用过渐变混色的图像中，我选择平色。大多数情况下，它也正常工作，不会引入混杂的色带和仿色、增大文件尺寸（如图22-6所示）。你还可以选择用巧妙的混色替代照片区域，比如，如果你需要保存为GIF图像（否则，JPEG格式会更好），可以让蓝天使用平色。



这个GIF有渐变混色，而且为256色。它的文件大小为19KB。



甚至当我把颜色数减少到8时，文件大小还得7.6KB。



当使用平色创建时，文件大小仅为3.2KB。

图22-6: 利用GIF的压缩特点来设计，可以减小文件尺寸

### 水平色带

有一个秘密的小提示。当你设计网页图像时，记住，GIF压缩处理水平色带最好。如果你想图像有条纹，最好使用水平的条纹，而不是垂直的（如图22-7所示）。看起来很无聊，但很有效。

### 总结GIF优化

GIF格式为优化提供很多机会。首先，使用平色创建小的GIF图像是一个好的策略。其次，用少而全的颜色保存GIF。另外，调整仿色量、应用损耗滤镜，也能减少更多字节。

在练习22-1中，将有机会尝试这些技术。

## 优化JPEG图像

JPEG的优化比GIF的稍微直观点。减小JPEG图像文件尺寸，通用策略是：

- 大幅度压缩
- 如果可以，使用加权（可选择的）优化
- 如果可以，选择优化
- 柔化图像（使用模糊、平滑滤镜）

本节介绍上面的每种方法，还是使用Photoshop和Fireworks的优化工具，如图22-9所示。注意，JPEG没有颜色表，因为它们根本不用色板。

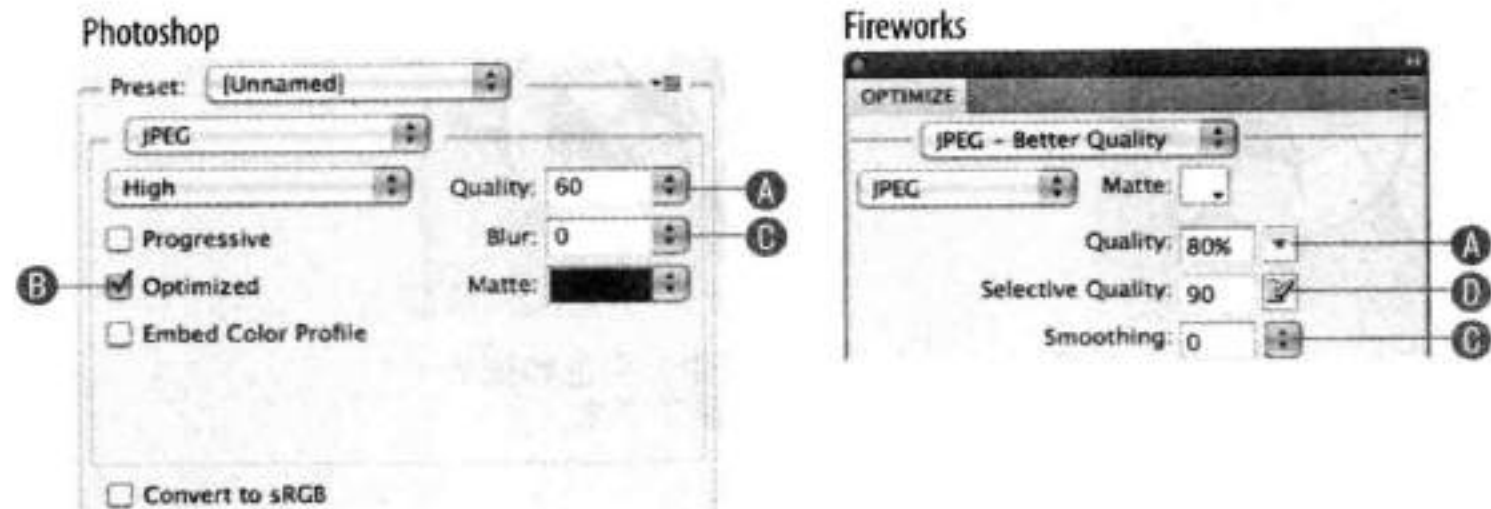


图22-9：Photoshop中的JPEG优化选项“存储为Web所用格式”（左图）和Fireworks优化面板（右图）

在具体设置之前，看看JPEG压缩所擅长的方面。这有助于对本节后面技术的理解。

## 了解JPEG压缩

JPEG压缩喜欢渐变图像，少细节，无硬边。保持JPEG图像较小，一个方法就是从它喜欢的这种图像开始。

### 避开细节

JPEG图像压缩中，当图像有大面积的平滑混色时效率较高，而当图像高对比度或者有强烈变化的细节时，效率较低。实际上，你的图像越模糊，结果JPEG就越小。图22-10显示了两张使用混色的类似图像。可以看到，在压缩率和品质设置都相同时，一幅图像对比度高、有细节，比另一幅大4倍以上。

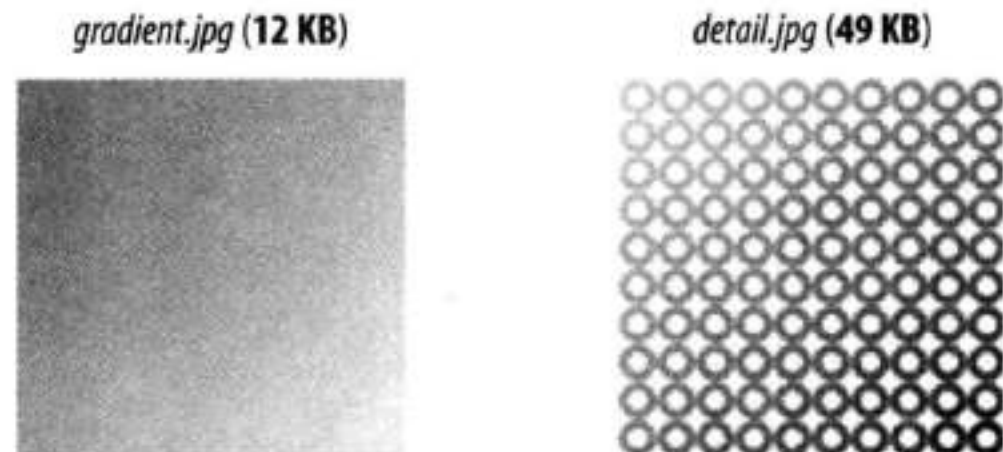


图22-10：相比有硬边和细节的图像，JPEG压缩在平滑、混色上的压缩效率更好

### 练习22-1 精简GIF图像

看看你是否可以在不损失图像质量的情况下，把图22-8减少图像尺寸到目标范围之内。初始图像可以在本章材料网页获得（[www.learningwebdesign.com](http://www.learningwebdesign.com)）。

如果你有Photoshop（版本6或更新）或者Fireworks（版本4或更新），可以使用本节涵盖的技术。你也可以使用其他的工具，比如Corel Paint Shop ProPhoto，但也许不能调整控制仿色或损耗设置。

获得想要的文件尺寸有很多方法，但没有“正确”的方法。它大多靠你的个人判断，但目标尺寸会给你一个合理的目标。



**ASIAN CUISINE**

*asian.psd; target: 4 to 5 KB*

**INFO**

*info.psd; target: <300 bytes*



*bunny.psd; target: 5 to 6 KB*

图22-8：创建优化到目标尺寸的GIF文件



### 避开平色

知道这一点很重要：平色完全不适合JPEG格式，因为JPEG压缩之后，颜色往往会变，而且整个图像变得斑驳，特别是压缩率很高时（如图22-11所示）。通常，平色图像应该保存为GIF格式的，因为这样图像质量更好、文件尺寸更小。

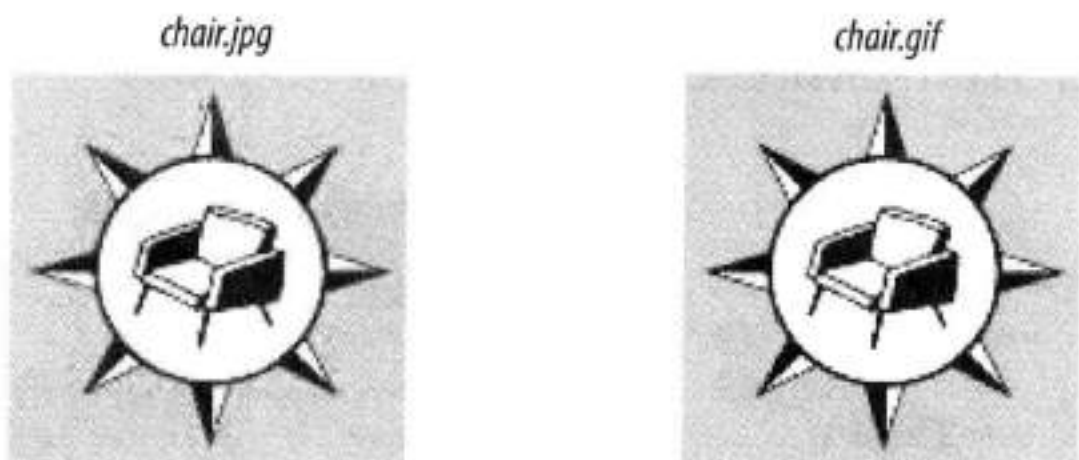
#### JPEG图像中不可预期的颜色

在GIF图像中，你可以完全控制出现在图像中的颜色。这样，图像的RGB颜色就很容易与邻近GIF图像、内联GIF图像和背景图像的颜色相匹配。

遗憾的是，JPEG压缩会产生平色异变和污点，所以无法精确控制颜色。即使是纯白色，在JPEG中也会变形。

这意味着，在JPEG和其他颜色（无论是GIF、PNG、另一个JPEG，还是RGB背景色）之间创建完美、无缝的匹配，没有安全的办法。如果你需要前景色与背景色无缝匹配，考虑切换为GIF或PNG格式，这样可以利用透明度使背景显露出来。

**警告：**记住把用于iPad Retina显示器的图像以渐进式格式来保存，以避免Safari自动把JPEG降低2M像素（在图像中，超过210万像素）。



在JPEG中，平色改变，并且图像斑驳，在JPEG压缩中，细节丢失了。

在GIF中，平色和细节都保留了下来。

图22-11：相同的平色图像保存为JPEG和GIF

### 大幅度压缩

优化JPEG图像的主要工具是品质设置（图22-9A）。品质设置允许设置压缩率；品质越低意味着压缩率越高、文件越小。图22-12显示了不同品质（压缩）率的结果，它们是分别在Photoshop和Fireworks中设置的。

注意，即使品质设置得很低，图像效果也不错。还要注意，相同设置的各个程序也会产生不同的结果。这是因为品质的比较并不客观——它因程序而异。例如，Photoshop中的1%与Fireworks及其他程序中30%差不多。另外，不同的图像受得住不同的压缩量。最好一边调一边看，而不是设置一个具体数值。

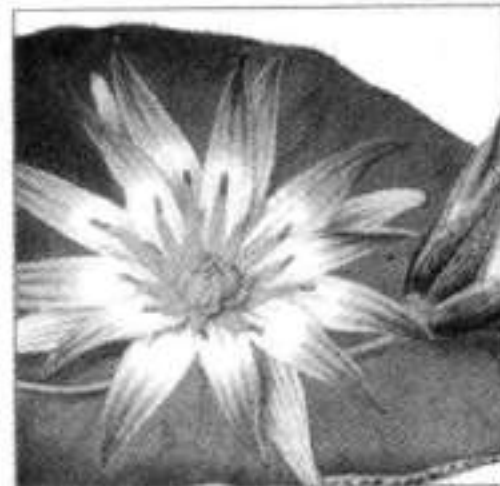
### 选择优化的JPEG图像

对比标准JPEG图像，优化的JPEG具有更小的体积和更好的颜色保真（虽然我还不能看出差别）。因此，如果图像软件提供，你就应该选中优化（Optimized）选项（图22-9B）。在Photosho和第三方JPEG压缩工具中寻找优化选项。到本书写作时，Fireworks尚未提供此选项。

Photoshop



100% (42.2 KB)



80% (22.3 KB)



60% (13.6 KB)



40% (8.5 KB)



20% (6.3 KB)



1% (3.7 KB)

Fireworks



100% (51.5 KB)



80% (12.3 KB)



60% (7.7 KB)



40% (5 KB)



20% (1.8 KB)



1% (1.2 KB)

图22-12: Photoshop和Fireworks中不同压缩级别的对比

## 模糊化或平滑化图像

因为柔和的图像压缩的比剧变的图像小，作为优化过程的一部分，Photoshop 和 Fireworks 会轻微地模糊图像。在Photoshop中，这个工具叫“模糊”（Blur）（图22-9©）；在Fireworks，它是“平滑”（Smoothing）。模糊化可使JPEG压缩得更好，致使文件更小（图22-13）。如果没有这些工具，在导出之前，可以手动使用高斯模糊（Gaussian Blur）（或类似的）滤镜对图像进行轻微模糊化，这样来使整个图像柔和。





Quality: 20; Blur: 0 (9.3 KB)

JPEG保存为低品质（Photoshop中20%），没有应用模糊。



Quality: 20; Blur: .5 (7.2 KB)

模糊设置仅为0.5，文件大小就减少22%。在Fireworks中，使用平滑可以得到类似的效果。

图21-13：在导出为JPEG结果前，轻微模糊图像，会使文件尺寸更小

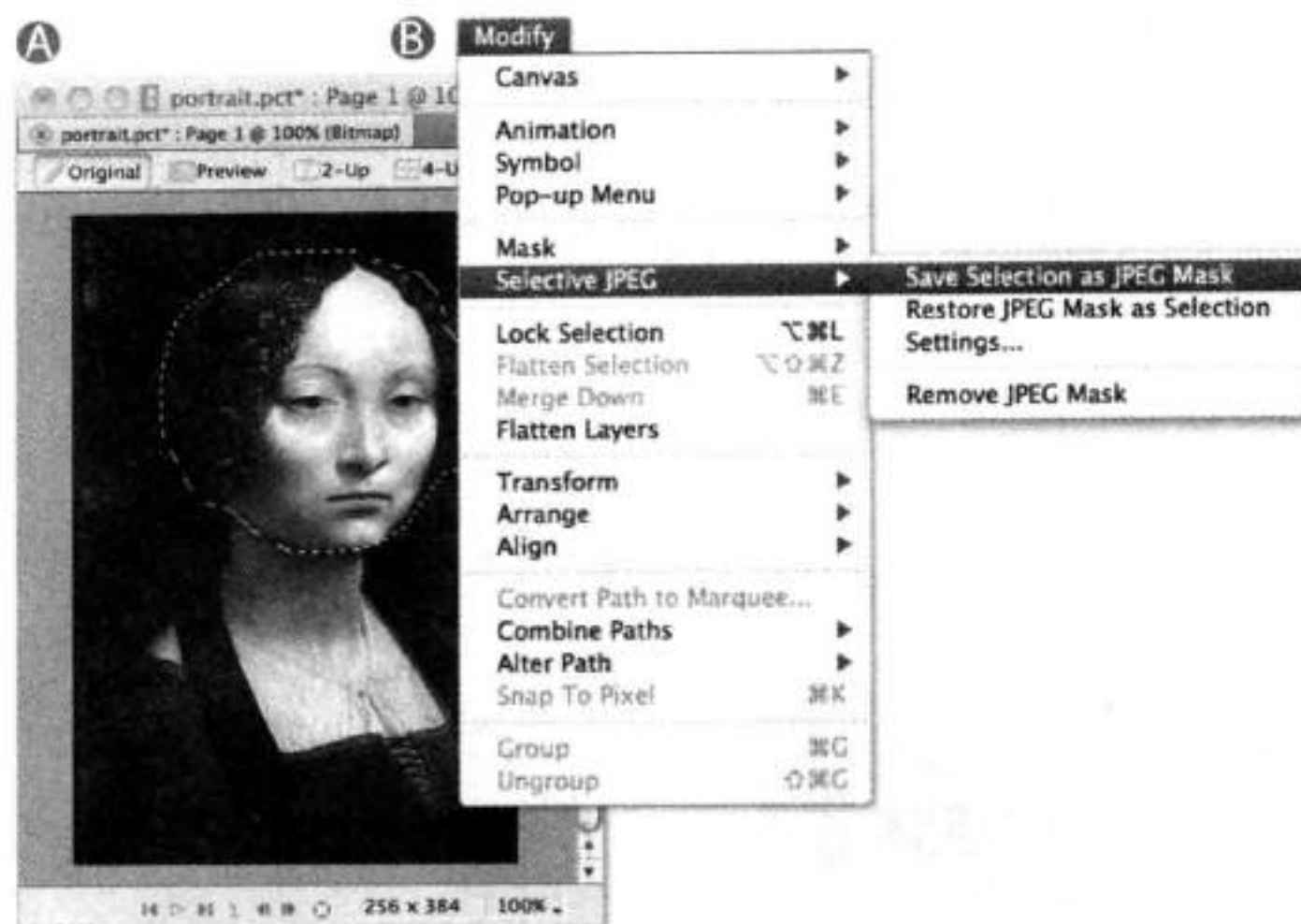
模糊和平滑化滤镜会均匀地应用到整个图像，这是它们的不足。如果你想保留图像某些区域的细节，你可以只对你不介意模糊的区域，应用模糊滤镜。完成后，像平常一样导出JPEG。被模糊化的区域将充分利用JPEG的压缩，清晰的区域还是保持清晰。

## 选择性品质（Fireworks）

不是所有的图像区域都生而平等。你可能想保留某一区域的细节，比如人脸，但又想极度压缩图像剩余区域。最后，Fireworks提供了“选择性品质（Selective Quality）”。这种方法对相同的图像应用不同的JPEG压缩率：一个应用到选中的部分，另一个应用到剩余地方。

**注意：** Photoshop在CS3及其之前版本中有一个类似的功能“加权优化”，但在CS4版本中去掉了。

要使用“选择性品质”设置（如图22-9⑤），选择你想保留图像的区域（图22-14①），然后选择“修改”→“选择性JPEG”→“选择保存为JPEG遮片”②。在“优化”面板中，你可以设置选择性质量，或按一下旁边的图标③来访问有全套选项的“选择性JPEG”对话框④。比如保存类型、按钮质量和选择颜色被屏蔽的区域。常规的质量设置将被用于其他区域的图像。



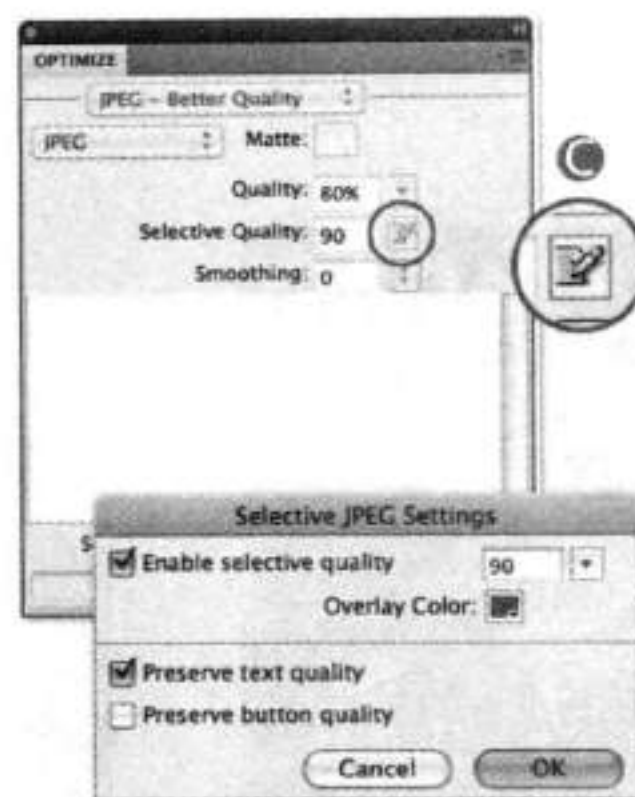
选择你希望保留的区域，并保存为选择性JPEG遮片。

图22-14：在Fireworks中使用选择性品质

## JPEG优化小结

优化JPEG图像的主要工具是品质（压缩）设置。如果你的工具能处理，设置“优化”，或者应用模糊化、平滑化都将使图像更小。

现在轮到你玩转练习22-2中的JPEG图像了。



在优化面板中，选择性品质旁边的按钮可以访问品质全套选项。



## 练习22-2 优化JPEG图像

看看你是否可以使用本节的技术，把图22-15减少图像尺寸到目标范围之内。没有正确答案，按你的方式来进行。最重要的一点是，各种设置对文件尺寸和图像品质的影响如何，你要有个总体感觉。



*falcon.tif*  
target: 35–40 KB

想象一下，如果这个图像需要出现在出售海报的站点上（在这种站点上，保持图像的类型和细节非常重要），那么就不能像其他图像一样压缩这个图像。



*boats.psd*  
target: 24–30 KB

观察这个JPEG作品中的线和船只的桅杆，试着保持这些线条清晰。



*penny.tif*  
target: 12–18 KB

在压缩之前，最好先对这个图像的背景进行手动模糊处理。

图22-15：符合文件尺寸

## 优化PNG图像

如第21章所讨论的，有两种PNG文件。一种是24位PNG图像（PNG-24），它包含RGB颜色空间中的上百万种颜色；另一种是8位索引PNG图像（PNG-8），它有一个限制为256色的色板。本节关注你能（和不能）做什么，来影响这两种PNG文件的尺寸。

## PNG-24

PNG是无损压缩，可以作为保证质量的极好格式，但很可惜，它不太适合网页图像。PNG-24总是比相同图像的JPEG大得多，因为在压缩过程中没有牺牲任何像素。因此，你的第一个“精简”策略是不使用PNG-24作为照片图像，而选择JPEG代替。

当然，这个规则有一个特别的例外：如果你想使用多级透明度（Alpha透明度）。这种情况下，鉴于现在的工具和浏览器，PNG-24是你唯一的选择。

没有任何窍门来减小PNG-24的文件尺寸，从PNG-24导出面板可以看得出来，面板中缺乏选项（如图22-16所示）。你将不得不接受图像编辑工具完成的文件尺寸。虽然你可能尝试通过在线图片优化工具Smush.it（[www.smush.it](http://www.smush.it)）来运行它，看它是否能做出改善。

## PNG-8

索引颜色PNG图像的工作机制类似于GIF图像，而且事实上，往往导致相同图像的尺寸更小，因此它成为节省字节的好选择。优化GIF图像的策略同样适用于PNG-8图像：

- 减少图像的颜色数（位数）
- 减小图像的仿色
- 使用平色设计

你可看到PNG-8图像的导出选项列表与GIF的差不多相同（如图22-16所示）。有一个需要注意的例外是，PNG图像没有“有损的”滤镜，而GIF有。除此之外，“优化GIF图像”一节列出的所有技术也适用于PNG图像。

有一点值得注意，设置PNG交错将极大增加文件尺寸，多达20%或30%。最好不使用这个选项，除非你认为，图像绝对需要多路径显示。

要深入了解PNG的压缩和优化，我推荐“Smashing Magazine”中Sergey Chikuyonok的文章“灵巧的PNG优化技术”（[www.smashingmagazine.com/2009/07/15/clever-png-optimization-techniques/](http://www.smashingmagazine.com/2009/07/15/clever-png-optimization-techniques/)）。

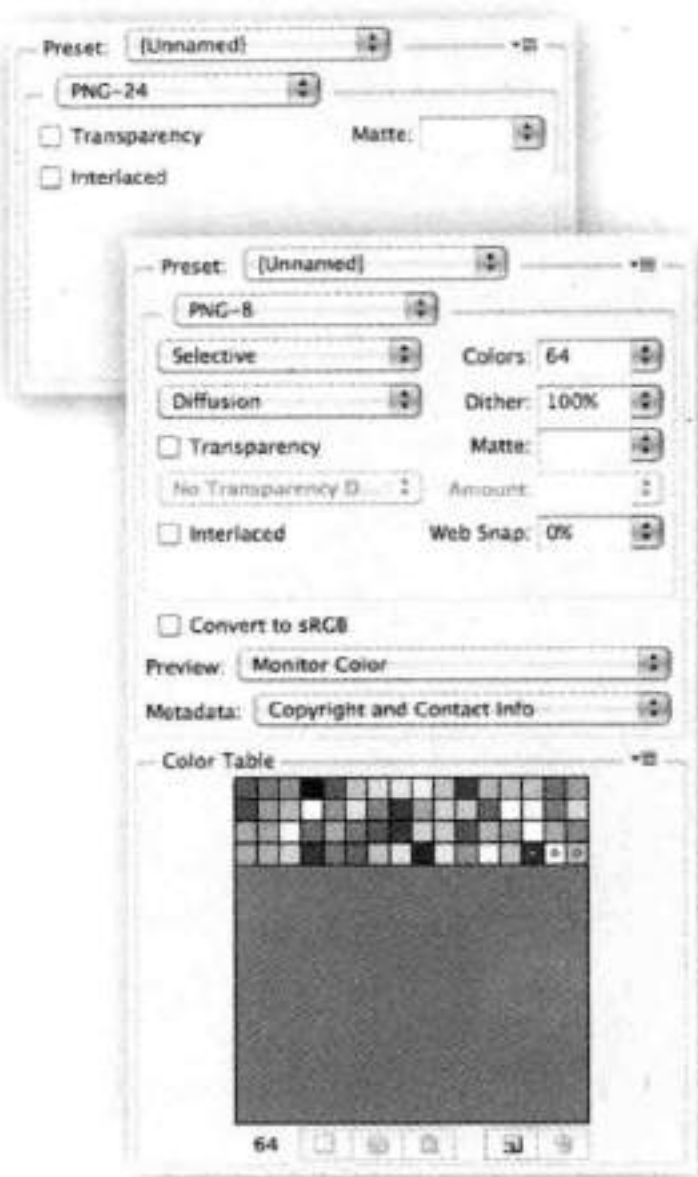
## 优化到指定大小

如果你使用Photoshop或Fireworks，最好要了解这最后一项优化技术。

在某些时候，你需要优化一幅图像，达到指定文件尺寸，比如设计一个有严格K阈（K-limit）的广告横幅。Photoshop和Fireworks都提供“优化到指定大小”的功能。你只需要设置为你想要的文件尺寸，程序会自动为你规划设置，以达到指定的大小，为你节约了大量冥思苦想的时间。

这个功能用起来相当直观。在Photoshop中，可以在“存储为Web所用格式”的选项弹出菜单中选择“优化文件大小”。在Fireworks中，在优化面板的选项弹出菜单中选择“优化到指定大小”（如图22-17所示）。你

Photoshop



Fireworks

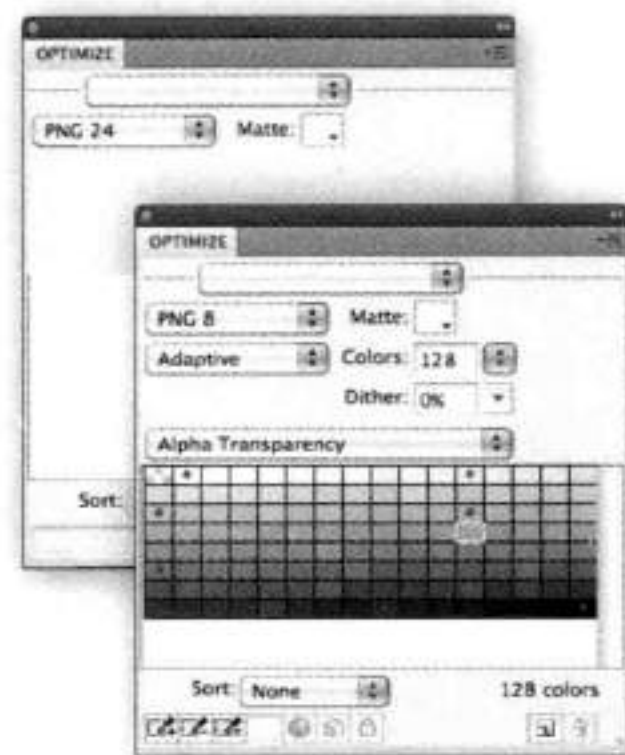


图22-16: Photoshop 和 Fireworks中的PNG-24和PNG-8设置



唯一要做的就是输入你想要的目标尺寸并单击“好（OK）”。其他的事情交给工具了。

Photoshop还会问，起点是你自己的优化设置，还是自动选择GIF或JPEG。奇怪的是，PNG不是自动选择的选项，所以，如果你想保存为PNG，那就从你自己的设置开始。

在Photoshop中优化到指定文件尺寸  
在选项弹出菜单中输入你的目标尺寸，  
选择优化到指定文件尺寸。



在Fireworks中优化到指定文件尺寸  
在选项弹出菜单中输入你的目标尺寸，选择  
优化到指定文件尺寸。



图22-17：优化到指定文件尺寸（在Photoshop和Fireworks中）

## 小结

如果这一系列优化技术让你感到不堪重负，也不用太过担心。不久以后，它们将成为标准制作流程。你将会发现，时刻注视文件尺寸，使用一些设置来微微下调这个尺寸值，这一点很容易做到。你已经理解了各种设置在后台如何运行，既然了解了这些，那你就可以作出一个有见识、有效的优化决策。

## 自我测验

既然你已经见识了图像优化的世界，那么现在该你做个测试了。我知道你一定行。

1. 为什么专业Web设计师要优化图像？

2. 仿色是如何影响GIF的文件尺寸的?
3. 像素颜色数是如何影响GIF的文件尺寸的?
4. 优化JPEG最有效的设置是什么?
5. 模糊和平滑化是如何影响JPEG尺寸的?
6. 优化PNG-8最好的方法是什么? PNG-24呢?





# 附录A 答案

## 第1章：我从哪里开始

1. B, D, A, C
2. W3C管理Web相关技术的开发。
3. C, D, A, E, B
4. 前台设计涉及网站显示的各个方面，并与浏览器相关。后台开发专注于服务器上网站功能所需的编程工作。
5. Web创作工具为创建整个网页（包括必需的HTML、CSS和脚本）提供可视化的界面。HTML编辑器只提供手工编写HTML文档的快捷方式。

## 第2章：Web是如何工作的

1. c; 2. j; 3. h; 4. g; 5. f; 6. i; 7. b; 8. a; 9. d; 10. e

## 第3章：Web设计基本概念

1. 开发一个站点时，有许多未知因素：
  - 屏幕和浏览器窗口的尺寸是多少
  - 用户的互联网连接速度是多少
  - 用户是在办公桌前还是在旅途中（环境和注意力）
2. 1. c; 2. d; 3. e; 4. a; 5. b
3. 视力障碍：当使用屏幕阅读器读取时，确保内容符合语义和逻辑顺序。
  - 听力障碍：提供音频和视频内容的复制
  - 行动不便：给没有鼠标或键盘的用户提供一些措施
  - 认知功能障碍：内容应该是简单而且组织明确的
4. 你将使用瀑布图来评估你的站点在优化过程中的表现。



5. 自适应设计需要照顾到布局,但本身并不会提供适合移动环境下的备用内容。服务器能够比CSS媒体查询检测到更多的功能,并能对服务内容做出更好的决策。

## 第4章:创建简单网页(HTML概述)

1. 标签是用来给确定元素边界的标记。元素由内容和标记组成。
2. HTML文档的最少标记如下:

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="utf8">
 <title>Title</title>
</head>
<body>
</body>
</html>
```

3.
  - a. *Sunflower.html*——Yes
  - b. *index.doc*——No,必须以*.html*或*.htm*结尾
  - c. *cooking home page.html*——No,不应该有字符空格
  - d. *Song\_Lyrics.html*——Yes
  - e. *games/rubix.html*——No,名称中应该没有斜线
  - f. *%whatever.html*——No,应该没有百分符号
4. 下列所有标记例子都不正确。描述每个例子的错误,并正确编写。
  - a. 缺失src特性: ``
  - b. 缺失结束标签的斜线: `<i>Congratulations!</i>`
  - c. 结束标签应该没有特性: `<a href="file.html">linked text</a>`
  - d. 斜线应该是正斜线: `<p>This is a new paragraph</p>`
5. 注释如下: `<!-- product list begins here -->`

## 第5章:标记文本

1. 

```
<p>People who know me know that I love to cook.</p>
<hr>
<p>I've created this site to share some of my favorite
 recipes.</p>
```
2. `blockquote`是一种块级元素,用于由其他块元素组成的长引用或引用材料。`q` (`quote`) 元素用于短小的引用,它们在文本流中,不引起换行。
3. `pre`
4. `ul`元素表示无序列表,用于不需要以特定顺序显示的列表。它们默认前置项目符合。`ol`元素是有序列表,与次序有关。浏览器会自动插入有序列表的编号。

- 
5. 使用样式表移除无序列表的项目符号。
  6. `<abbr title="World Wide Web Consortium">W3C</abbr>`
  7. `dl`是用于标识整个定义列表的元素。`dt`元素用于标识该列表中的一项。
  8. The `id` attribute is used to identify a unique element in a document, and the name in its value may appear only once in a document. `class` is used to classify multiple elements into conceptual groups.
  9. `article`元素用于内容的自包含主体, 适用于组织, 或者应该显示在不同的上下文中。`section`把内容分成主题相关的块。
  10. 

<code>&amp;mdash;</code>	长破折号 (—)
<code>&amp;amp;</code>	and符号 (&)
<code>&amp;nbsp;</code>	非换行空格
<code>&amp;copy;</code>	版权符号 (©)
<code>&amp;bull;</code>	项目符号 (·)
<code>&amp;trade;</code>	商标符号 (™)

### 练习5-1

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="utf-8">
 <title>Tapenade Recipe</title>
</head>
<body>

<h1>Tapenade (Olive Spread)</h1>

<p>This is a really simple dish to prepare and it's always a big hit
at parties.My father recommends:</p>

<blockquote><p>"Make this the night before so that the flavors have
time to blend. Just bring it up to room temperature before you serve it.
In the winter, try serving it warm."</p></blockquote>

<h2>Ingredients</h2>

 1 8oz. jar sundried tomatoes
 2 large garlic cloves
 2/3 c. kalamata olives
 1 t. capers

<h2>Instructions</h2>

 Combine tomatoes and garlic in a food processor. Blend until
as smooth as possible.
```



```

Add capers and olives. Pulse the motor a few times until they are incorporated, but still retain some texture.

Serve on thin toast rounds with goat cheese and fresh basil garnish (optional).

</body>
</html>

```

## 练习5-2

```

<article>
 <header>
 <p>posted by BGB,<time datetime="2012-11-15" pubdate>November 15, 2012</time></p>
 </header>
 <h1>Low and Slow</h1>
 <p>This week I amextremely excited about a new cooking technique called <dfn><i>sous vide</i></dfn>. In <i>sous vide</i> cooking, you submerge the food (usually vacuum-sealed in plastic) into a water bath that is precisely set to the target temperature you want the food to be cooked to. In his book,<cite>Cooking for Geeks</cite>, Jeff Potter describes it as <q>ultra-low-temperature poaching</q>.</p>
 <p>Next month, we will be serving Sous Vide Salmon with Dill Hollandaise. To reserve a seat at the chef table, contact us before November 30.</p>
 <p>blackgoose@example.com
555-336-1800</p>
 <p><small>Warning: Sous vide cooked salmon is not pasteurized. Avoid it if you are pregnant or have immunity issues.</small></p>
</article>

```

## 练习5-3

```

<!DOCTYPE html>
<html>
 <head>
 <meta charset="utf-8">
 <title>Black Goose Bistro: Blog</title>
 </head>
 <body>
 <header>
 <h1>The Black Goose Blog</h1>
 <nav>

 Home
 Menu
 Blog
 Contact

 </nav>
 </header>

 <article>
 <header>
 <h2>Summer Menu Items</h2>
 <p>posted by BGB,<time datetime="2013-06-15" pubdate>June 15, 2013</time></p>
 </header>
 <p>Our chef has been busy putting together the perfect menu for the Summer months. Stop by to try these appetizers and main courses while

```

the days are still long.</p>

```
<section id="appetizers">
 <h3>Appetizers</h3>
 <dl>
 <dt>Black bean purses</dt>
 <dd>Spicy black bean and a blend of mexican cheeses wrapped in sheets
of phyllo and baked until golden. $3.95 </dd>
 <dt class="newitem">Southwestern napoleons with lump crab —
new item!<dt>
 <dd>Layers of light lump crab meat, bean and corn salsa, and our
handmade flour tortillas.$7.95</dd>
 </dl>
</section>
<section id="maincourses">
 <h3>Main courses</h3>
 <dl>
 <dt>Shrimp sate kebabs with peanut sauce</dt>
 <dd>Skewers of shrimp marinated in lemongrass, garlic, and fish sauce then grilled to perfection.
Served with spicy peanut sauce and jasmine rice.$12.95</dd>

 <dt class="newitem">Jerk rotisserie chicken with fried plantains —new item!</dt>
 <dd>Tender chicken slow-roasted on the rotisserie, flavored with spicy and fragrant jerk sauce and
served with fried plantains and fresh mango.$12.95</dd>
 </dl>
</section>
</article>

<article>
 <header>
 <h2>Low and Slow</h2>
 <p>posted by BGB,<time datetime="2012-11-15" pubdate>November 15, 2012</time></p>
 </header>
 <p>This week I am extremely excited about a new cooking technique called <dfn><i>sous vide</i></dfn>. In<i>sous vide</i>cooking, you submerge the food (usually vacuum-sealed in plastic) into a water bath that is precisely set to the target temperature of the food.In his book,<cite>Cooking for Geeks</cite>, Jeff Potter describes it as <q>ultra-low-temperature poaching</q>.</p>
 <p>Next month, we will be servingSous Vide Salmon with Dill Hollandaise. To reserve a seat at the chef table, contact us before November 30.</p>
</article>

<footer>
 <div id="about">
 <p>Location:
Baker's Corner, Seekonk, MA</p>
 <p>Hours:
Tuesday to Saturday,<time datetime="11:00">11am</time>to<time datetime="00:00">midnight</time></p>
 </div>
 <p><small>All content copyright © 2012, Black Goose Bistro and Jennifer Robbins</small><p>
</footer>

</body>
</html>
```

## 第6章：添加链接

1. <a href="tutorial.html">...</a>
2. <a href="examples/instructions.html">...</a>



3. `<a href="examples/french/family.html">...</a>`
4. `<a href="/examples/german/numbers.html">...</a>`
5. `<a href="../index.html">...</a>`
6. `<a href="http://www.learningwebdesign.com">...</a>`
7. `<a href="../instructions.html">...</a>`
8. `<a href="../../index.html">...</a>`
9. ``
10. ``
11. ``

#### 练习6-1

```
Epicurious
```

#### 练习6-2

```
<p>Back to the home page</p>
```

#### 练习6-3

```
Tapenade (Olive Spread)
```

#### 练习6-4

```
Linguine with Clam Sauce
```

#### 练习6-5

```
<p>[Back to the home page]</p>
```

#### 练习6-6

```
<p>[Back to the home page]</p>
```

#### 练习6-7

1. `<p><a href="tapenade.html">Go to the Tapenade recipe</a></p>`
2. `<p><a href="../salmon.html">Try this with Garlic Salmon</a></p>`
3. `<p><a href="pasta/linguine.html">Try the Linguine with Clam Sauce</a></p>`
4. `<p><a href="../../about.html">About Jen' s Kitchen</a></p>`
5. `<p><a href="http://www.allrecipes.com">Go to AllRecipes.com</a></p>`

## 第7章：添加图片

1. 对于有效的文档，src和alt特性是必需的。如果省略了src特性，那么浏览器就不知道显示哪幅图像。如果阅读内容时替换文本既无意义又笨拙，那么你可以将alt特性留空。
2. ``
3.
  - a. 在不可访问或不可读的情况下，通过提供图像的描述信息可以提高易用性；
  - b. 如果省略了alt特性，那么HTML文档不再是有效的。
4. 这样可以让浏览器在从服务器获取图像的时候，显示内容的其他部分，从而加速网页的显示。如果你在设计一个自适应站点，图像大小需要保持灵活性，那么就不要写width和height属性了。
5. 丢失图像的三种可能的原因是：
  - a. URL不正确，所以浏览就会在错误的地点寻找错误的文件名称（名称是区分大小写的）；
  - b. 图像文件不是可接受的格式；
  - c. 图像文件的后缀名不正确（应当是.gif、.jpg或.png）。

### 练习7-1

在 *index.html* 中：

```
<h2>The Tuscan Countryside</h2>

<p>This is ...</p>

<h2>Sienna</h2>

<p> The closest city ...</p>
```

在 *countryside.html* 中：

```
<p></p>
```

在 *sienna.html* 中：

```
<p></p>
```

## 第8章：表格标记

1. 表格自身（table）、行（tr）、表头格（th）、数据单元格（td），以及可选的标题（caption）。
2. 如果你想添加表格结构的额外信息，想指定width来加速显示，或者想给一行单元格添加某些样式属性。
3.
  - a. caption应该是table元素内的第一个元素；
  - b. table元素不能直接包含文本。文本必须放在th或td中；
  - c. th元素应该在tr元素内；



- d. 没有colspan元素，应该有带colspan特性的td元素；
- e. 第二个tr元素缺失关闭标签。

### 练习8-1

```
<table>
<tr>
 <th>Album</th>
 <th>Year</th>
</tr>
<tr>
 <td>Rubber Soul</td>
 <td>1968</td>
</tr>
<tr>
 <td>Revolver</td>
 <td>1966</td>
</tr>
<tr>
 <td>Sgt. Pepper's</td>
 <td>1967</td>
</tr>
<tr>
 <td>The White Album</td>
 <td>1968</td>
</tr>
<tr>
 <td>Abbey Road</td>
 <td>1969</td>
</tr>
</table>
```

### 练习8-2

```
<table>
 <tr>
 <th>7:00pm</th><th>7:30pm</th><th>8:00pm</th>
 </tr>
 <tr>
 <td colspan="3">The Sunday Night Movie</td>
 </tr>
 <tr>
 <td>Perry Mason</td>
 <td>Candid Camera</td>
 <td>What's My Line</td>
 </tr>
```

```

<tr>
 <td>Bonanza</td>
 <td colspan="2">The Wackiest Ship in the Army</td>
</tr>
</table>

```

### 练习8-3

```

<table>
 <tr>
 <td>apples</td>
 <td rowspan="3">oranges</td>
 <td>pears</td>
 </tr>
 <tr>
 <td>bananas</td>
 <td rowspan="2">pineapple</td>
 </tr>
 <tr>
 <td>lychees</td>
 </tr>
</table>

```

### 练习8-4

```

<table>
 <caption>Your Content Here</caption>
 <tr>
 <th rowspan="2"> </th>
 <th colspan="2">A common header for two subheads</th>
 <th rowspan="2">Header 3</th>
 </tr>
 <tr>
 <th>Header 1</th>
 <th>Header 2</th>
 </tr>
 <tr>
 <th scope="row">Thing A</th>
 <td>data A1</td>
 <td>data A2</td>
 <td>data A3</td>
 </tr>
 <tr>
 <th scope="row">Thing B </th>
 <td>data B1</td>
 <td>data B2</td>
 <td>data B3</td>
 </tr>
 <tr>
 <th scope="row">Thing C</th>
 <td>data C1</td>
 <td>data C2</td>
 <td>data C3</td>
 </tr>

```



</table>

## 第9章：表单

1.
  - a. POST（由于安全因素）
  - b. POST（因为使用了文件选择的输入类型）
  - c. GET（因为你想要将搜索结果设置为书签）
  - d. POST（因为可能有长篇文本输入）
2.
  - a. 下拉菜单：<select>
  - b. 单选按钮：<input type="radio">
  - c. <textarea>
  - d. 8个复选框：<input type="checkbox">
  - e. 滚动菜单：<select multiple="multiple">
3. 下列示例代码都包含一个错误，请指出
  - a. 缺失type特性
  - b. checkbox不是元素名，它是input元素的type特性的值
  - c. option元素不是空元素，每个option元素都应该包含值（例如，<option>Orange</option>）。
  - d. 缺失必需的name特性。
  - e. 应该分别使用cols和rows特性，指定文本区域的宽度和高度。

### 练习9-1到练习9-3：最终源文档

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="utf-8">
 <title>Contest Entry Form</title>
 <style type="text/css">
 ol, ul {
 list-style-type: none;
 }
 </style>
</head>

<body>

<h1>“Pimp My Shoes” Contest Entry Form</h1>

<p>Want to trade in your old sneakers for a custom pair of Forcefields? Make a case for why your shoes have
 gotto go and you may be one of ten lucky winners.</p>

<form action="http://www.learningwebdesign.com/contest.php" method="post">
<fieldset>
```

```

<legend>Contest Entry Information</legend>

<label for="form-name">Name:</label><input type="text" name="username" id="form-name">
<label for="form-email">Email Address:</label><input type="email" name="emailaddress" id="form-email">

<label for="form-tel">Telephone Number:</label><input type="tel" name="telephone" id="form-tel">
<label for="form-story">My shoes are SO old...</label>

<textarea name="story" rows="4" cols="60" maxlength="300" id="form-story"placeholder="No more than 300
 characters long"></textarea>

</fieldset>
<h2>Design your custom Forcefields:</h2>

<fieldset>
<legend>Custom Shoe Design</legend>

<fieldset>
<legend>Color(choose one)</legend>

<label><input type="radio" name="color" value="red"> Red</label>
<label><input type="radio" name="color" value="blue"> Blue</label>
<label><input type="radio" name="color" value="black"> Black</label>
<label><input type="radio" name="color" value="silver"> Silver</label>

</fieldset>

<fieldset>
<legend>Features(Choose as many as you want)</legend>

<label><input type="checkbox" name="feature" value="laces">Sparkley laces</label>
<label><input type="checkbox" name="feature" value="logo" checked>Metallic logo</label>
<label><input type="checkbox" name="feature" value="heels">Light-up heels</label>
<label><input type="checkbox" name="feature" value="mp3">MP3-enabled</label>

</fieldset>

<fieldset>
<legend>Size</legend>
<label for="form-size"><p>Sizes reflect standard men's sizes:</label>
<select id="form-size" name="size" size="1">
 <option>5</option>
 <option>6</option>
 <option>7</option>
 <option>8</option>
 <option>9</option>
 <option>10</option>
 <option>11</option>
 <option>12</option>
 <option>13</option>
</select>
</p>
</fieldset>

</fieldset>

<p><input type="submit" value="Pimp My Shoes!">
<input type="reset"> </p>
</form>

```



```
</body>
</html>
```

## 第10章：HTML 5

1. XHTML由XML语法规则定义并且要求更严格。HTML是随意的。
2.
  - a. `<h1> ... </h1>`
  - b. ``
  - c. `<input type="radio" checked="checked">`
  - d. `<hr />`
  - e. `<title>Sifl & Oilly</title>`
  - f. 

```

 popcorn
 butter
 salt

```
3. 一个DTD代表文档类型定义，同时它也是一个定义了一种语言的所有元素、属性和值的文件。语言以及它们使用规则文件。
4. HTML 5的HTML规范在以下方面是唯一的：
  - 它包括API，而不只是元素和属性的定义。
  - 它包括了浏览器如何渲染元素和处理错误的说明。
  - 它不使用DTD。
  - 它可以用HTML或XHTML语法来编写。
5. 使用HTML元素可以使用任何属性。
6. Web工作者，d；编辑API，e；地理定位API，a；Web插座，b；离线应用，c
7. OGG，容器；H.264，视频；VP8，视频；Vorbis，音频；WebM，容器；Theora，视频；AAC，音频；MPEG-4，容器
8. `strokeRect()`和`fill()`

## 第11章：CSS入门

1. selector: `blockquote`；property: `line-height`；value: `1.5`，declaration: `line-height: 1.5`
2. 段落文本将是灰色的，因为当与相同权重的规则冲突的时候，样式表最后列出的元素胜出。
3.
  - a. 将应用到p元素的多个声明写入一条规则。

```
p {font-face: sans-serif;
 font-size: 1em;
 line-height: 1.2em;}
```
  - b. 缺失分号。

```
blockquote {
 font-size: 1em;
 line-height: 150%;
 color: gray;
}
```

c. 每个声明周围应该都有紧挨着声明块的花括号。

```
body {background-color: black;
 color: #666;
 margin-left: 12em;
 margin-right: 12em;}
```

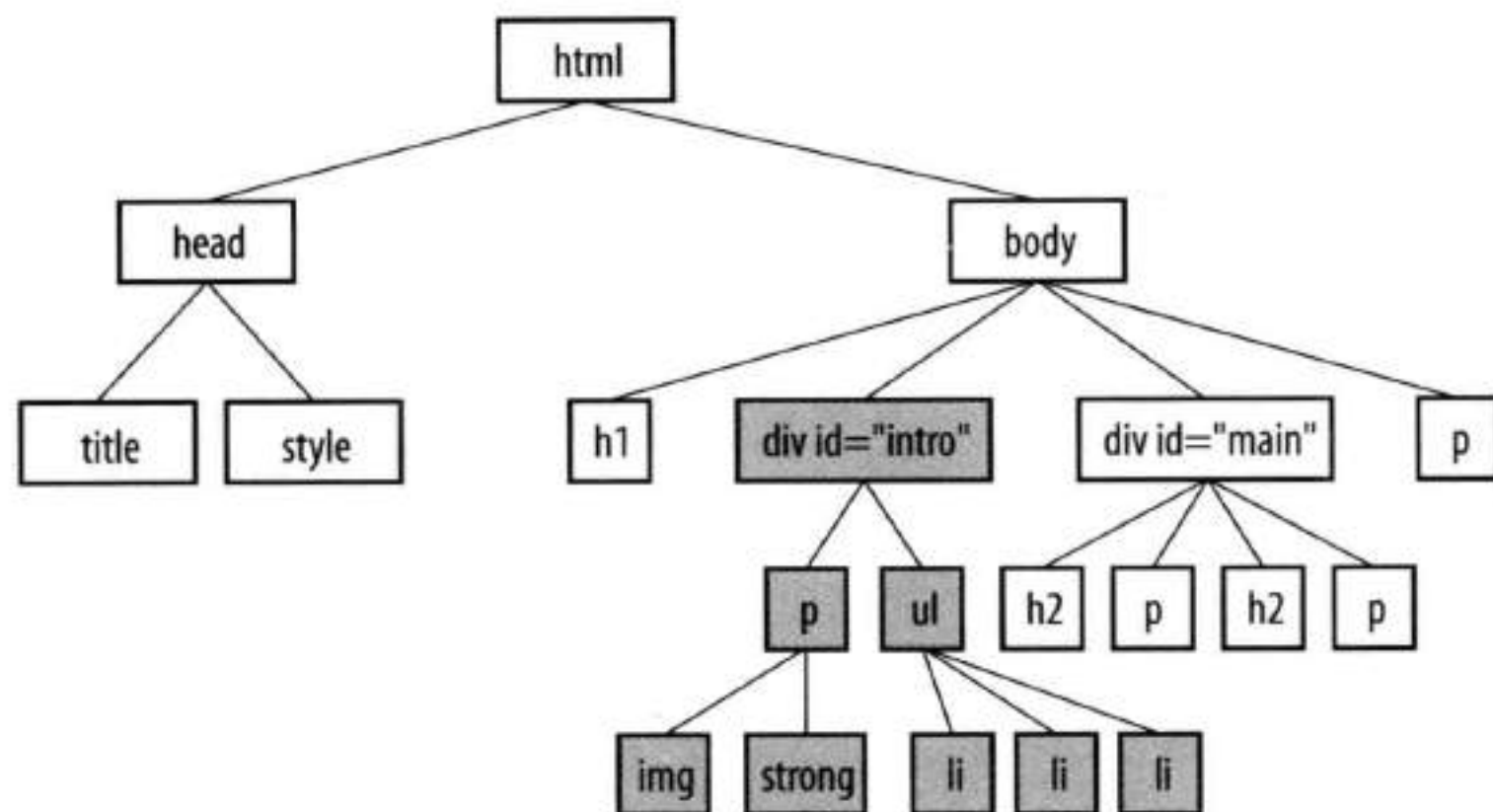
d. 使用组合元素类型选择器，只需要处理一条规则。

```
p, blockquote, li {color: white;}
```

e. 内联样式缺失属性名称。

```
<strong style="color: red">Act now!
```

4. `div#intro { color: red; }`



图A-1: `div#intro{color:red;}`样式规则的结果是：高亮元素应该为红色

## 练习11-1

```
h1 {
 color: red;
 border-bottom: 1px solid red;
}
p {
 font-size: small;
 font-family: sans-serif;
 margin-left: 100px;
}
h2 {
 color: red;
 margin-left: 100px;
}
img {
```



```
float: right;
margin: 0 12px;
}
```

## 第12章：格式化文本（使用更多选择器）

1.
  - a. 文档中所有文本元素：body {color: red;}
  - b. h2元素：h2 {color: red;}
  - c. h1元素和所有的段落：h1, p {color: red;}
  - d. 属于“special”类的所有元素：.special {color: red;}
  - e. “intro”块中的所有元素：#intro {color: red;}
  - f. “main”块中的所有strong元素：#main strong{color: red;}
  - g. 附加题：显示在main节后的段落（提示：这个选择器在IE 6上无法工作）：h2+p{color:red;}
2.
  - a. ④ , b. ① , c. ⑦ , d. ③ , e. ② , f. ⑨ , g. ⑧ , h. ⑤ , i. ⑥

### 练习12-1到练习12-3

```
<head>
<meta charset="utf-8">
<title>Black Goose Bistro Summer Menu</title>
<link href='http://fonts.googleapis.com/css?family=Marko+One' rel='stylesheet'>
<style>

body {
 font-family: Georgia, serif;
 font-size: 100%;
 line-height: 1.75em;
}
p, dl
{
 font-size: .875em;
}
h1 {
 font: bold 1.5em "Marko One", Georgia, serif;
 color: purple;
 text-shadow: .1em .1em .2em lightslategray;
}
h2 {
 font-size: 1em;
 text-transform: uppercase;
 letter-spacing: .5em;
 color: purple;
}
dt {
 font-weight: bold;
 color: sienna;
}
strong {
 font-style: italic;
}
```

```

dt strong {
 color: maroon;
}
#info p {
 font-style: italic;
 color: gray;
}
.price {
 font-family: Georgia, serif;
 font-style: italic;
 color: gray;
}
p.warning, sup {
 font-size: small;
 color: red;
}
.label {
 font-weight: bold;
 font-variant: small-caps;
 font-style: normal;
}
h1, h2, #info {
 text-align: center;
}
h2 + p {
 text-align: center;
 font-style: italic;
}

</style>
</head>

```

## 第13章：颜色和背景（附加更多选择器和外部样式表）

1. g. a, b和c
2. d. rgb(FF, FF, FF)
3. a.-5; b.-1; c.-4; d.-6; e.-2; f.-3
4. a.-1; b.-3; c.-2; d.-6; e.-5; f.-4

### 练习13-1

```

body {
 ...
 background - color: #d2dc9d;
}
#header {
 ...
 background - color: rgba(255, 255, 255, .5);
}
a: link {
 color: #939;
}
a: visited {
 color: #937393;
}

```



```

}
a: focus {
 background - color: #fff;
 color: #c700f2;
}
a: hover {
 background - color: #fff;
 color: #c700f2;
}
a: active {
 background - color: #fff;
 color: #f0f;
}
h1 {
 ...
 color: #939;
}
h2 {
 ...
 color: #c60;
}

```

### 练习13-2

```

body {
 ...
 background - color: #d2dc9d;
 background - image: url(images / bullseye.png);
}

```

### 练习13-3

```

#header {
 ...
 background - color: rgba(255, 255, 255, .5);
 background - image: url(images / purpledots.png);
 background - repeat: repeat - x;
}

```

### 练习13-4

```

body {
 ...
 background-color: #d2dc9d;
 /* background-image: url(images/bullseye.png);
 background-position: center 200px; */
 background-image: url(images/blackgoose.png);
 background-repeat: no-repeat;
 background-position: center 100px;
}
#header {
 ...
 background-color: rgba(255,255,255,.5);
 background-image: url(images/purpledots.png);
 background-repeat: repeat-x;
 background-position: center top;
}

```

```
}
```

### 练习13-5

```
body {
 ...
 background-color: #d2dc9d;
 background-image: url(images/blackgoose.png);
 background-repeat: no-repeat;
 background-position: center 100px;
 background-attachment: fixed;
}
```

### 练习13-6

```
body {
 ...
 background: #d2dc9d url(images/blackgoose.png) no-repeat center 100px fixed;
}
#header {
 ...
 background: rgba(255,255,255,.5) url(images/purpledots.png) repeat-x center top;
}
```

### 练习13-7

```
#header {
 ...
 background-image: url(images/purpledots.png) center top repeat-x;
 background:
 url(images/purpledots.png) left top repeat-y,
 url(images/purpledots.png) right top repeat-y,
 url(images/gooseshadow.png) 90% bottom no-repeat;
 background-color: rgba(255,255,255,.5);
}
```

### 练习13-8

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

## 第14章：盒子思想（填充、边框和空白边）

- a. border: double black medium;
- b. overflow: scroll;
- c. padding: 2em;
- d. padding: 2em; border: 4px solid red;
- e. margin: 2em; border: 4px solid red;



f. padding: 1em 1em 1em 6em; border: 4px dashed; margin: 1em 6em;

或者

padding: 1em; padding-left: 6em; border: 4px dashed; margin: 1em 6em;

g. padding: 1em 50px; border: 2px solid teal; margin: 0 auto;

#### 练习14-1

```
#products {
 ...
 padding: 1em;
}
#testimonials {
 ...
 padding: 1em;
 padding-left: 55px;
}
```

#### 练习14-2

```
#products {
 ...
 padding: 1em;
 border: double #FFBC53;
}
#products h3 {
 ...
 border-top: 1px solid;
 border-left: 3px solid;
 padding-left: 1em;
}
#testimonials {
 ...
 padding: 1em;
 padding-left: 55px;
 border-radius: 20px;
}
a {
 text-decoration: none;
 border-bottom: 1px dotted;
 padding-bottom: .1em;
}
```

#### 练习14-3

```
body {
 margin: 0;
}
a {
 text-decoration: none;
 border-bottom: 1px dotted;
 padding-bottom: .1em;
}
/* 省略链接样式来节省空间*/
```

```

/* intro节的样式*/
#intro {
 text-align: center;
 margin: 2em 0 1em;
}
#intro h1 {
 margin-bottom: 0;
}
#intro h2 {
 ...;
 margin-top: -10px;
}
#intro p {
 ...
 margin: 1em;
}
/* 省略导航的样式节省空间 */

/* products的样式*/
#products {
 ...
 padding: 1em;
 border: double #FFBC53;
 margin: 1em;
}
...
#products h3 {
 ...
 border-top: 1px solid;
 border-left: 3px solid;
 padding-left: 1em;
 margin-top: 2.5em;
}

/* testimonials盒子的样式*/
#testimonials {
 ...
 padding: 1em;
 padding-left: 55px;
 border-radius: 20px;
 margin: 1em 10%;
}
/* 省略剩余的样式来节省空间*/

```

## 第15章：浮动与定位

1. b不正确，浮动是沿着容器元素的内容区域（不是填充边缘）确定位置。
2. c不正确，浮动元素不使用offset属性，所以没有理由包括right特性。
3. 清除脚本div，使它从浮动侧栏下方开始摆设：div#footer { clear: both; }
4. a) absolute; b) absolute, fixed; c) fixed; d) relative, absolute, fixed; e) static; f) relative; g) absolute, fixed; h) relative, absolute, fixed; i) relative



### 练习15-1

```
#products img {
 float: left;
 margin: 0 6px 6px 0;
}
#products .more {
 clear: left;
}
```

### 练习15-2

```
#nav ul {
 ...
 margin: 0 auto;
 width: 19.5em;
}
#nav ul li {
 ...
 float: left;
}
#nav ul li a {
 display: block;
 padding: .5em;
 border: 1px solid #ba89a8;
 border-radius: .5em;
 margin: .25em;
}
...
#nav ul a:focus {
 color: #FC6;
 border-color: #fff;
}
#nav ul a:hover {
 color: #fc6;
 border-color: #fff;
}
...
#products {
 ...
 clear: both;
}
```

### 练习15-3

```
#products {
 ...
 width: 55%;
 float: left;
}
#products h2 {
 ...
 text-align: left;
}
#testimonials {
 ...
 margin: 1em 2% 1em 64%;
}
```

```
}
p#copyright {
 ...
 clear: left;
}
```

#### 练习15-4

```
...
#content {
 position: relative;
}
#testimonials {
 ...
 margin: 0 1em;
 position: absolute;
 top: 0;
 right: 0;
 width: 14em;
}
#products {
 ...
 margin: 1em 20.5em 1em 1em;
 clear: both;
}
#award {
 position: absolute;
 top: 35px;
 left: 25px;
}
```

#### 练习15-5

```
...
#award {
 position: fixed;
 top: 35px;
 left: 25px;
}
```

### 第16章：使用CSS进行网页布局

1. Fixed, c.; Liquid, a.; Elastic, b.
2. Fixed, b.; Liquid, c.; Elastic, a.
3. Fixed, c.; Liquid, b.; Elastic, a.
4. Fixed, c.; Liquid, a.; Elastic, b.

#### 练习16-1

```
<style>
#wrapper {
 width: 960px;
 margin: 0 auto;
```



```

}
#header {
 background-color: #CCC;
 padding: 15px;
}
#links {
 float: right;
 width: 22.5%;
 margin: 0 2.5% 0 0 ;
 outline: 2px dashed #dd0009;
}
#main {
 float: right;
 width: 45%;
 margin: 0 2.5%;
 outline: 2px dashed #0053ae;
}
#news {
 float: right;
 width: 22.5%;
 margin: 0 0 0 2.5% ;
 outline: 2px dashed #009554;
}
#footer {
 clear: right;
 padding: 15px;
 background: #CCC;
}
/* 省略其余没变化的样式，以节省空间 */
</style>

<body>
<div id="wrapper">
 ... contents of page here...
</div>
</body>

```

## 练习16-2

```

#main {
 float: left;
 width: 400px;
 margin-top: 0;
 margin-left: 320px;
 margin-right: 20px;
}

#news {
 float: left;
 width: 300px;
 margin-top: 0;
 margin-left: -740px;
}
#links {
 float: left;
 width: 220px;
 margin: 0;
}

```

## 第17章：过渡、变换和动画

1. 补间是动画中的一个过程，帧产生于其中的两个端部点状态。
2. 过渡将有两个关键帧：一个为开始状态；一个为结束状态。
3. a) transition-delay: 0.5s; b) transition-timing-function: linear; c) transition-duration: .5s; d) transition-property: line-height;
4. c) text-transform不是一个可动的属性。
5. “舒适”是默认的定时功能。它开始很慢，加速很快，然后在最后再次减慢。
6. 0.2 s是transition-duration值。
7. 棘手的问题！它们将在过渡期开始300ms后在同一时间到达。定时功能对于它所花费的时间没有影响。
8. a) transform: rotate(7deg); b) translate(-25px, -50px); c) transform-origin: bottom right; d) transform: scale(1.2);
9. 3表示该元素应调整其原始高度的3倍。
10. a) perspective: 250; 因为较低的数字值更吸引人。
11. 3个像素宽的边界可以通过动画的50%;
12. a) animation-direction: reverse; b) animation-duration: 5s; c) animation-duration: 2s; d) animation-iteration-count: 3;

### 练习17-1

```
a {
 /* 非过渡样式省去，以节省空间*/
 position: relative;
 -webkit-transition: background-color 0.2s ease-in, border-color 0.2s, top 0.2s, box-shadow 0.2s;
 -moz-transition: background-color 0.2s, border-color 0.2s, top 0.2s, box-shadow 0.2s;
 -o-transition: background-color 0.2s, border-color 0.2, top 0.2s, box-shadow 0.2s;
 -ms-transition: background-color 0.2s, border-color 0.2s, top 0.2s, box-shadow 0.2s;
 transition: background-color 0.2s, border-color 0.2s, top 0.2s, box-shadow 0.2s;
}
a:hover, a:focus {
 background-color: #fdca00;
 border-color: #fda700;
}
a:active {
 top: 3px;
 box-shadow: 0 1px 2px rgba(0,0,0,.5);
}
```

### 练习17-2

供应商前缀的属性被省略以节约空间。

```
img {
 width: 200px;
 height: 150px;
```



```

 box-shadow: 2px 2px 2px rgba(0,0,0,.4);
 transition: transform .3s ease-in-out;
}
a:hover img {
 box-shadow: 6px 6px 6px rgba(0,0,0,.3);
}
a:hover #img1, a:focus #img1 {
 transform: scale(1.5) rotate(-3deg);
}
a:hover #img2, a:focus #img2 {
 transform: scale(1.5) rotate(5deg);
}
a:hover #img3, a:focus #img3 {
 transform: scale(1.5) rotate(-7deg);
}
a:hover #img4, a:focus #img4 {
 transform: scale(1.5) rotate(2deg);
}

```

## 第18章：CSS技术

1. d) 上述所有
2. d) a 和 c
3. LESS和Sass之间的差异包括：
  - LESS缺乏一些Sass的功能。
  - 它们使用的语法略有不同(\$variable与@variable)。
  - Sass被一个在服务器上的Ruby程序编译成标准的CSS；LESS使用JavaScript。
4. e) b 和 d
5. 给标签元素相同的宽度并使之在左侧浮动，然后把文本右对齐，它出现在它所描述的控制旁边。
6. 如果你不设置窗口的尺寸，即使它被设计为宽320像素，手机浏览器也将缩小页面。
7. c, e, d, a, b
8. b, e, a, d, c

### 练习18-1到练习18-3

```

img {
 max-width: 100%;
}

@media screen and (min-width: 481px) {
 #products img {
 float: left;
 margin: 0 6px 6px 0;
 }
 #products .more {
 clear: left;
 }
}

```

```

#products {
 margin: 1em;
}
#testimonials {
 margin: 1em 5%;
 border-radius: 16px;
}
}

@media screen and (min-width: 780px) {
 #products {
 float: left;
 margin: 0 2% 1em;
 clear: both;
 width: 55%;
 overflow: auto;
 }
 #testimonials {
 margin: 1em 2% 1em 64%;
 }
 p#copyright {
 clear: both;
 }
 #content {
 max-width: 1024px;
 margin: 0 auto;
 }
}

```

## 第19章：JavaScript简介

1. 当你链接外部的.js文件时，你可以重复使用多个文件的相同脚本。不足之处是，它需要一个额外的HTTP请求。
2. a) 1; b) 1two; c) 34; d) 2
3. a) 10; b) 6; c) “2 保留” ; d) “Jennifer 更长” ; e) 错误
4. 当没有剩下时，通过在第一个阵列中的开始和结束，它循环了一些条款。
5. 在全局范围内的变量可能与在其他脚本中具有相同名称的变量发生“碰撞”。最好的方法是使用var关键字的功能来让你的变量域限定为局部。
6. a. 2; b. 5; c. 4; d. 3; e. 1

### 练习19-1

1. `var friends = ["name", "othername", "thirdname", "lastname"];`
2. `alert(friends[2]);`
3. `var name = "yourName";`
4. `if( name === Jennifer) { alert("That' s my name too!"); } }`
5. `var myVariable = #;`

```
if(myVariable > 5) {
 alert("upper");
} else {
 alert ("lower");
}
```

## 练习19-2

```
<script>
var originalTitle = document.title;
function showUnreadCount(unread) {
 document.title = originalTitle + " (" + unread + "new message!");
}
showUnreadCount(3);
</script>
```

## 第20章：使用JavaScript

1. Ajax是HTML、CSS和JavaScript的组合（使用XMLHttpRequest JavaScript方法从后台获取数据。）
2. 访问id值为“main”的那个元素。
3. 在id值为“main”的元素中，创建所有元素的nodeList。
4. 把页面背景（body元素）色设置为“番木色”。
5. 创建一个新的文本节点，上面写着：“嘿，我在这里散步！”，将其插入一个新创建的p元素中，并把新的p元素放进id为“main”的元素中。
6. a. 3; b. 2; c. 4; d. 1
7. 以上全部。

## 第21章：Web图像基础

1. 你可以获得图像专有权利的许可，这样竞争者就不能在他们的网站上使用相同的图片。
2. ppi代表“pixels per inch”，是分辨率的计量单位。
3. 使用索引色模式，将每个像素颜色存储在颜色表中，从而储存图像的颜色信息。GIF和8位PNG格式都是索引色图像。
4. 8位图像有256种颜色，5位图像有32种颜色。
5. GIF可以包含动画和透明度，而JPEG不能。
6. GIF可以包含动画，而PNG不能。
7. PNG可以有多级透明。GIF只有二元透明（要么透明，要么不透明）。
8. 有损压缩是不断累积的，这意味着每次将图像保存为JPEG时，都会丢失图像数据。如果打开JPEG并再次保存为JPEG，将会比第一次保存时扔掉更多的信息。确保留存了全质量的原图，并按需要保存JPEG副本。



- 
9. 在二元透明度中，一个像素要么完全透明，要么完全不透明。Alpha透明度允许最多256个级别的透明度。
  10. A. GIF或PNG-8，因为它是文本，是平色、硬边的。B. JPEG，因为它是照片。C. GIF或PNG-8，因为虽然有一些照片区域，但图像的大部都是平色，且是硬边。D. GIF或PNG-8，因为它是平色图像。E. JPEG 因为它是照片。

## 第22章：精简Web图像

1. 更小的图形文件，意味着下载和显示时间更短。对每一秒都精打细算，可以创造更好的用户体验。
2. 仿色引入了不纯粹的图案，破坏同一颜色的连续性，因此，GIF压缩方案在压缩仿色区域，效果没有压缩平色好。
3. 图像中像素颜色数越少，最终的GIF就越小。既是因为图像能以更小位深存储，又是因为GIF压缩类似颜色的更多区域。
4. 在控制JPEG图像文件大小方面，压缩率设置是最有效的工具。
5. 在平滑和模糊区域，JPEG压缩能高效工作，所以引入一点模糊，可以使JPEG压缩效率更高，而生成更小的文件。
6. 就像对索引GIF所做的，通过设计时使用平色、减少颜色数和避免仿色，可以优化PNG-8。没有办法优化PNG-24，因为PNG-24天生就是以无损压缩方式存储图像的。



# 附录B CSS3选择器

选择器	选择器类型	描述
简介选择器组合		
*	通用选择器	匹配任何元素 * {font-family: serif;}
A	元素类型选择器	匹配元素名称 div {font-style: italic;}
A, B	组合选择器	匹配元素A和元素B h1, h2, h3 {color: blue;}
A B	后代选择器	只匹配作为元素A后代的元素B blockquote em {color: red;}
A>B	子选择器	匹配作为元素A孩子的元素B div.main>p {line-height: 1.5;}
A+B	相邻同胞选择器	匹配紧跟着元素A的元素B，A与B有相同的父元素 p+ul {margin-top: 0;}
A~B	通用同胞选择器	匹配以元素A为先导的元素B，A与B有相同父元素 block quote~cite {margin-top: 0}
类和ID选择器		
.classname A.classname	类选择器	匹配所有元素或指定元素的class特性值 p.credits {font-size: 80%;}
#idname A#idname	ID选择器	匹配一个元素的id特性值 #intro {font-weight: bold;}



选择器	选择器类型	描述
属性选择器		
A[att]	简单特性选择器	匹配定义了给定的特性（无论是否有值）的元素A table[border] {background: white;}
A[att="val"]	精确特性值选择器	匹配指定特性赋予指定值的元素A table[border="3"] {background: yellow;}
A[att~="val"]	部分特性值选择器	匹配元素A，其指定值是给定元素特性列表中的一个值 table[class~="example"] {background: yellow;}
A[att ="val"]	带连字号的前缀特性选择器	匹配元素A，其指定特性的指定值等于给定值，或以给定值开头。它最常用来选择语言，如下 a[lang ="en"] {background-image: url(en_icon.png);}
A[att^="val"]	开始子串属性选择器	匹配有特定属性的任意元素A，它的值以给定字符串开始 img[src^="/images/icons"] {border: 3px solid;}
A[att\$=" val" ]	结束子串属性选择器	匹配有特定属性的任意元素A，它的值以给定字符串结束 img[src\$="/images/icons"] {border: 3px solid;}
A[att*=" val" ]	任意子串属性选择器	匹配有特定属性的任意元素A，它的值包含给定字符串 img[title#="July"] {border: 3px solid;}
伪类选择器		
a:link	伪类链接选择器	指定尚未访问过的链接的样式 a:link {color: maroon;}
a:visited	伪类链接选择器	指定尚已经访问过的链接的样式 a:visited {color: gray;}
:active	用户动作伪类选择器	指定已经被用户激活的元素，比如正在单击的链接 a:active {color: red;}
:focus	用户动作伪类选择器	指定当前输入焦点所在的元素，比如选中的表单输入框 input[type="text"]:focus {background: yellow;}
:hover	用户动作伪类选择器	指定鼠标指针悬于其上时元素的显示样式 a:hover {text-decoration: underline;}
:target	伪类选择器	选择用于片段标识符的元素
:lang(xx)	伪类选择器	选择匹配双字符语言代码的元素 a:lang(de) {color: green;}
:root	结构化伪类选择器	选择文档的根元素。在HTML中，就是html元素 :root { background: papayawhip;}
:nth-child()	结构化伪类选择器	选择某个父元素的第n个子元素。这个符号包含一个数值、一个符号或者关键字odd或者even。 tr:nth-child(odd) { background: #DDD;}

选择器	选择器类型	描述
<code>:nth-last-child()</code>	结构化伪类选择器	选择某个父元素的倒数第n个子元素 <code>li:nth-last-child(2) { color: green;}</code>
<code>:nth-of-type()</code>	结构化伪类选择器	选择某个类型的第n个元素 <code>img:nth-of-type(even) {float: right;}</code>
<code>:nth-last-of-type()</code>	结构化伪类选择器	选择某个类型的倒数第n个元素 <code>img:nth-last-of-type(odd) {float: right;}</code>
<code>:first-child</code>	结构化伪类选择器	选择父元素的第一个子元素 <code>p:first-child {border-top: 1px solid;}</code>
<code>:last-child</code>	结构化伪类选择器	选择父元素的最后一个子元素 <code>p:last-child {border-bottom: 1px solid;}</code>
<code>:first-of-type</code>	结构化伪类选择器	选择某种类型的第一个元素 <code>dt:first-of-type { font-weight: bold;}</code>
<code>:last-of-type</code>	结构化伪类选择器	选择某种类型的最后一个元素 <code>li:last-of-type {margin-bottom: 1em;}</code>
<code>:only-child</code>	结构化伪类选择器	选择父元素的唯一子元素 <code>aside:only-child {line-height: 1.5;}</code>
<code>:only-of-type</code>	结构化伪类选择器	选择某种类型的唯一兄弟元素 <code>dt:first-of-type {font-weight: bold;}</code>
<code>:empty</code>	结构化伪类选择器	选择没有文本也没有子元素的一个元素 <code>tbody td:empty {background: #000; }</code>
<code>:enabled</code>	UI伪类选择器	选择一个有效的UI元素 <code>input[type="tel"]:enabled {border: 1px solid red;}</code>
<code>:disabled</code>	UI伪类选择器	选择一个无效的UI元素 <code>input[type="tel"]:disabled {color: #ccc;}</code>
<code>:checked</code>	UI伪类选择器	选择一个选定的UI元素（单选按钮和复选框） <code>:checked {background-color: yellow;}</code>
<code>:not(X)</code>	否定伪类元素	选择一个与选择器X不匹配的元素 <code>:not(pre) { line-height: 1.2 }</code>
伪元素选择器		
<code>:first-letter</code> (CSS3中为 <code>::first-letter</code> )	伪元素选择器	选择指定元素的第一个字母 <code>p:first-letter {font-size: 4em;}</code>
<code>:first-line</code> (CSS3中为 <code>::first-line</code> )	伪元素选择器	选择指定元素的第一行 <code>.note:first-line {letter-spacing: 4px;}</code>
<code>:before</code> (CSS3中为 <code>::before</code> )	伪元素选择器	在指定元素的开头插入指定的文本，并对其应用样式 <code>p.intro:before {content: "start here"; color: gray;}</code>
<code>:after</code> (CSS3中为 <code>::after</code> )	伪元素选择器	在指定元素的结尾插入指定的文本，并对其应用样式 <code>p.intro:after {content: "fini"; color: gray;}</code>