

SAMS
**Teach
Yourself**

- 全球销量逾百万册的系列图书
- 连续十余年打造的经典品牌
- 直观、循序渐进的学习教程
- 掌握关键知识的最佳起点
- “Read Less, Do More”（精读多练）的教学理念
- 以示例引导读者完成最常见的任务

每章内容针对初学者精心设计，**1**小时轻松阅读学习，
24小时彻底掌握关键知识

每章**案例与练习题**助你轻松完成常见任务，
通过**实践**提高应用技能，巩固所学知识

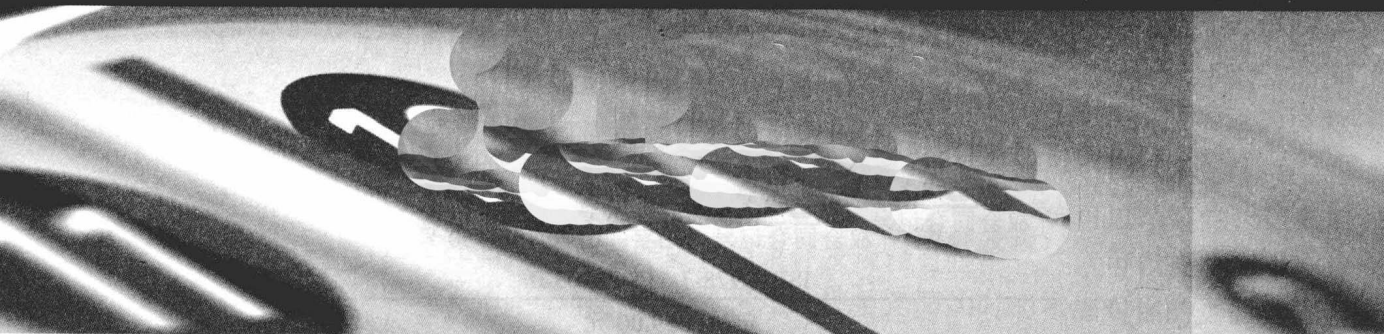
HTML、CSS和 JavaScript

入门经典

[美] Julie C. Meloni 著
陈宗斌 译

HTML、CSS和 JavaScript 入门经典

[美] Julie C. Meloni 著
陈宗斌 译



人民邮电出版社
北京

图书在版编目 (C I P) 数据

HTML、CSS和JavaScript入门经典 / (美) 梅洛尼
(Meloni, J. C.) 著 ; 陈宗斌译. — 北京 : 人民邮电出版社, 2012. 7

ISBN 978-7-115-28303-0

I. ①H… II. ①梅… ②陈… 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. ①TP312②TP393.092

中国版本图书馆CIP数据核字(2012)第098931号

版 权 声 明

Julie C. Meloni: Sams Teach Yourself HTML, CSS and JavaScript All in One

ISBN: 0672333325

Copyright © 2012 by Pearson Education, Inc.

Authorized translation from the English languages edition published by Pearson Education, Inc.

All rights reserved.

本书中文简体字版由美国 Pearson Education, Inc.授权人民邮电出版社出版。未经出版者书面许可, 对本书任何部分不得以任何方式复制或抄袭。

版权所有, 侵权必究。

HTML、CSS 和 JavaScript 入门经典

- ◆ 著 [美] Julie C.Meloni
- 译 陈宗斌
- 责任编辑 傅道坤
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
中国铁道出版社印刷厂印刷
- ◆ 开本: 787×1092 1/16
印张: 29.5
字数: 738千字 2012年7月第1版
印数: 1-3000册 2012年7月北京第1次印刷

著作权合同登记号 图字: 01-2010-6750号

ISBN 978-7-115-28303-0

定价: 59.00元

读者服务热线: (010)67132692 印装质量热线: (010)67129223

反盗版热线: (010)67171154

内容提要

本书采用直观的、逐步深入的方法介绍如何使用 HTML、CSS（Cascading Style Sheet，层叠样式表）和 JavaScript 设计、创建和维护世界级的网站，并且提供了清晰的指导。

本书信息丰富，涵盖了 HTML、CSS 和 JavaScript 的基础知识和高级应用，还介绍了如何组织和管理网站以及帮助人们找到你的 Web 页面的知识。在每一章末尾还配有相应的问答题和作业，可以帮助你检验所学的知识。

本书示例丰富，操作步骤简洁、明了，非常适合于从事 Web 设计和网站开发的初学者阅读，对于有一定经验的读者，也有很大的参考价值。

关于作者

Julie C. Meloni 是美国弗吉尼亚大学的网上图书馆环境中的首席技术专家和架构师。在进入图书馆之前,她有 15 年以上的时间是在为硅谷的多家大型和小型公司从事 Web 应用程序开发。她还编写了关于基于 Web 的程序设计语言和数据库主题的多本图书和若干篇文章,包括最畅销的 *Sams Teach Yourself PHP, MySQL, and Apache All in One* 一书。

目 录

第 1 章 发布 Web 内容	1	第 2 章 理解 HTML 与 XHTML 之间的 联系	19
1.1 HTML 和 World Wide Web 的简短 历史	2	2.1 做好准备	19
1.2 创建 Web 内容	2	2.2 从简单的 Web 页面开始	20
1.3 理解 Web 内容递送	3	2.3 每个 XHTML Web 页面都必须具 有的 HTML 标签	24
1.4 选择 Web 托管提供商	5	2.4 利用段落和换行符组织页面	26
1.5 利用多种 Web 浏览器进行测试	6	2.5 利用标题组织内容	28
1.6 创建一个示例文件	7	2.6 验证 Web 内容	30
1.7 使用 FTP 传输文件	8	2.7 关于 HTML、XML、XHTML 和 HTML5 的内幕消息	32
1.7.1 选择 FTP 客户	8	2.8 总结	33
1.7.2 使用 FTP 客户	9	2.9 问与答	34
1.8 理解在 Web 服务器上的什么位置 存放文件	11	2.10 作业	34
1.8.1 基本的文件管理	11	2.10.1 测验	34
1.8.2 使用索引页面	12	2.10.2 答案	34
1.9 在没有 Web 服务器的情况下分布 内容	14	2.10.3 练习	35
1.9.1 在本地发布内容	14	第 3 章 理解层叠样式表	36
1.9.2 在博客上发布内容	14	3.1 CSS 的工作原理	37
1.10 测试 Web 内容的几点提示	15	3.2 基本的样式表	38
1.11 总结	15	3.3 CSS 样式基础	42
1.12 问与答	16	3.3.1 布局属性	42
1.13 作业	17	3.3.2 格式化属性	43
1.13.1 测验	17	3.4 使用样式类	46
1.13.2 答案	17		
1.13.3 练习	17		

3.5	使用样式 ID	47	5.3	处理特殊字符	75
3.6	内部样式表和内联样式	48	5.4	在页面上对齐文本	78
3.7	总结	50	5.4.1	使用属性	78
3.8	问与答	51	5.4.2	对齐块级元素	79
3.9	作业	51	5.5	三种 HTML 列表	80
3.9.1	测验	52	5.6	在列表内放置列表	82
3.9.2	答案	52	5.7	总结	86
3.9.3	练习	52	5.8	问与答	87
第 4 章	理解 JavaScript	53	5.9	作业	87
4.1	学习 Web 脚本编程的基础		5.9.1	测验	87
	知识	54	5.9.2	答案	88
4.1.1	脚本和程序	54	5.9.3	练习	88
4.1.2	JavaScript 简介	55	第 6 章	使用表格显示信息	89
4.2	JavaScript 如何适应 Web 页面	55	6.1	创建简单的表格	89
4.2.1	使用单独的 JavaScript 文件	57	6.2	控制表格大小	92
4.2.2	理解 JavaScript 事件	57	6.3	表格内的对齐和跨越	94
4.3	探索 JavaScript 的能力	58	6.4	利用表格进行页面布局	97
4.3.1	改进导航	58	6.5	总结	98
4.3.2	验证表单	58	6.6	问与答	99
4.3.3	特殊效果	58	6.7	作业	99
4.3.4	远程脚本调用 (AJAX)	58	6.7.1	测验	99
4.4	利用 JavaScript 显示时间	59	6.7.2	答案	99
4.5	开始创建脚本	59	6.7.3	练习	100
4.6	添加 JavaScript 语句	59	第 7 章	使用外部链接和内部链接	101
4.6.1	在变量中存储数据	59	7.1	使用 Web 地址	101
4.6.2	计算结果	60	7.2	使用锚在页面内建立链接	104
4.7	创建输出	60	7.2.1	利用锚标识页面中的位置	104
4.8	向 Web 页面中添加脚本	61	7.2.2	链接到锚位置	104
4.9	测试脚本	62	7.3	在你自己的 Web 内容之间创建链接	106
4.9.1	修改脚本	62	7.4	链接到外部 Web 内容	108
4.9.2	处理 JavaScript 错误	64	7.5	链接到电子邮件地址	109
4.10	总结	65	7.6	在新的浏览器窗口中打开链接	111
4.11	问与答	66	7.7	使用 CSS 编排超链接的样式	111
4.12	作业	66	7.8	总结	114
4.12.1	测验	66	7.9	问与答	115
4.12.2	答案	67	7.10	作业	115
4.12.3	练习	67	7.10.1	测验	116
第 5 章	处理字体、文本块和列表	68	7.10.2	答案	116
5.1	粗体字、斜体字和特殊的文本格式化效果	69			
5.2	调整字体	72			

7.10.3 练习	116	8.23.2 答案	157
第 8 章 处理颜色、图像和多媒体	117	8.23.3 练习	158
8.1 用于选择颜色的最佳实践	118	第 9 章 处理边距、填充、对齐和浮动	159
8.2 了解 Web 颜色	119	9.1 使用边距	160
8.3 使用颜色的十六进制值	121	9.2 填充元素	166
8.4 使用 CSS 设置背景、文本和边框的颜色	122	9.3 使所有内容保持对齐	169
8.5 选择图形软件	123	9.4 理解 float 属性	169
8.6 关于图形至少需要知道什么	124	9.5 总结	172
8.7 准备照片图像	125	9.6 问与答	172
8.7.1 裁剪图像	125	9.7 作业	172
8.7.2 调整图像大小	127	9.7.1 测验	172
8.7.3 调整图像颜色	128	9.7.2 答案	172
8.7.4 控制 JPEG 压缩	128	9.7.3 练习	173
8.8 创建广告条和按钮	129	第 10 章 理解 CSS 盒子模型和定位	174
8.9 减少图像中的颜色数量	131	10.1 CSS 盒子模型	174
8.10 处理透明的图像	131	10.2 关于定位的详细解释	177
8.11 创建平铺背景	132	10.3 控制元素的堆叠方式	180
8.12 创建动画式 Web 图形	133	10.4 管理文本流	182
8.13 把图像放在 Web 页面上	134	10.5 总结	183
8.14 利用文本描述图像	136	10.6 问与答	183
8.15 指定图像的高度和宽度	137	10.7 作业	184
8.16 对齐图像	137	10.7.1 测验	184
8.16.1 水平的图像对齐	137	10.7.2 答案	184
8.16.2 垂直的图像对齐	139	10.7.3 练习	184
8.17 把图像转变成链接	140	第 11 章 使用 CSS 对列表、文本和导航进行更多的处理	185
8.18 使用背景图像	143	11.1 HTML 列表回顾	186
8.19 使用图像映射	144	11.2 CSS 盒子模型如何影响列表	186
8.19.1 为什么图像映射并非总是必要的	144	11.3 放置列表项指示符	189
8.19.2 映射图像内的区域	145	11.4 利用列表项和 CSS 创建图像映射	190
8.19.3 为图像映射创建 HTML 代码	146	11.5 导航列表与普通列表之间有何区别	193
8.20 把多媒体整合进网站中	148	11.6 利用 CSS 创建垂直导航系统	193
8.20.1 链接到多媒体文件	149	11.6.1 编排单级垂直导航系统的样式	196
8.20.2 嵌入多媒体文件	150	11.6.2 编排多级垂直导航系统的样式	197
8.20.3 关于使用多媒体的额外几点提示	153	11.7 利用 CSS 创建水平导航系统	200
8.21 总结	154		
8.22 问与答	156		
8.23 作业	157		
8.23.1 测验	157		

11.8	总结	203	14.1.5	条件语句	232
11.9	问与答	204	14.1.6	循环语句	233
11.10	作业	205	14.1.7	事件处理程序	233
11.10.1	测验	205	14.1.8	首先运行哪个脚本	234
11.10.2	答案	205	14.2	JavaScript 语法规则	234
11.10.3	练习	205	14.2.1	大小写敏感性	234
第 12 章	创建固定布局或流体布局	206	14.2.2	变量、对象和函数名称	234
12.1	理解固定布局	207	14.2.3	保留字	235
12.2	理解流体布局	208	14.2.4	空白	235
12.3	创建固定/流体混合式布局	210	14.3	使用注释	235
12.3.1	从基本的布局结构开始	210	14.4	关于 JavaScript 的最佳实践	236
12.3.2	在固定/流体混合式布局中 定义两列	211	14.5	总结	237
12.3.3	设置布局的最小宽度	213	14.6	问与答	237
12.3.4	在固定/流体混合式布局中处理 列高度	214	14.7	作业	238
12.4	总结	218	14.7.1	测验	238
12.5	问与答	218	14.7.2	答案	238
12.6	作业	218	14.7.3	练习	238
12.6.1	测验	218	第 15 章	使用文档对象模型 (Document Object Model, DOM)	239
12.6.2	答案	219	15.1	理解文档对象模型 (Document Object Model, DOM)	240
12.6.3	练习	219	15.2	使用 window 对象	240
第 13 章	理解动态网站	220	15.3	处理 document 对象	241
13.1	理解不同类型的脚本编程	220	15.3.1	获取关于文档的信息	241
13.2	在 HTML 中包括 JavaScript	221	15.3.2	在文档中编写文本	242
13.3	显示随机内容	223	15.3.3	使用链接和锚	243
13.4	了解文档对象模型	226	15.4	访问浏览器的历史记录	243
13.5	基于用户交互更改图像	226	15.5	使用 location 对象	245
13.6	总结	228	15.6	关于 DOM 结构的更多知识	246
13.7	问与答	228	15.6.1	节点	247
13.8	作业	229	15.6.2	父对象与子对象	247
13.8.1	测验	229	15.6.3	兄弟对象	247
13.8.2	答案	229	15.7	处理 DOM 节点	248
13.8.3	练习	229	15.7.1	基本的节点属性	248
第 14 章	初识 JavaScript 程序设计	230	15.7.2	节点的关系属性	248
14.1	基本概念	230	15.7.3	文档方法	249
14.1.1	语句	230	15.7.4	节点方法	249
14.1.2	把任务与函数相结合	231	15.8	创建可定位的元素 (图层)	249
14.1.3	变量	231	15.8.1	利用 JavaScript 控制定位	250
14.1.4	了解对象	232	15.9	隐藏和显示对象	253
			15.10	修改页面内的文本	255
			15.11	向页面中添加文本	256
			15.12	总结	258

15.13 问与答.....	258	17.1.1 定义函数.....	280
15.14 作业.....	259	17.1.2 调用函数.....	281
15.14.1 测验.....	259	17.1.3 返回值.....	282
15.14.2 答案.....	260	17.2 介绍对象.....	283
15.14.3 练习.....	260	17.2.1 创建对象.....	284
第 16 章 使用 JavaScript 变量、字符串 和数组.....	261	17.2.2 对象的属性和值.....	284
16.1 使用变量.....	261	17.2.3 了解方法.....	284
16.1.1 选择变量名.....	262	17.3 使用对象简化脚本编程.....	285
16.1.2 使用局部变量和全局变量.....	262	17.3.1 定义对象.....	285
16.1.3 给变量赋值.....	263	17.3.2 定义对象的方法.....	286
16.2 了解表达式和运算符.....	264	17.3.3 创建对象的实例.....	286
16.2.1 使用 JavaScript 运算符.....	264	17.4 扩展内置对象.....	287
16.2.2 运算符优先级.....	265	17.5 使用 Math 对象.....	290
16.3 JavaScript 中的数据类型.....	265	17.5.1 四舍五入和截尾.....	290
16.4 在数据类型之间转换.....	266	17.5.2 生成随机数.....	290
16.5 使用 String 对象.....	267	17.5.3 其他 Math 函数.....	291
16.5.1 创建 String 对象.....	267	17.6 使用 Math 函数.....	291
16.5.2 赋值.....	267	17.7 使用 with 关键字.....	293
16.5.3 计算字符串的长度.....	269	17.8 处理日期.....	293
16.5.4 转换字符串的大小写.....	269	17.8.1 创建 Date 对象.....	293
16.6 处理子串.....	270	17.8.2 设置 Date 值.....	294
16.6.1 使用字符串的一部分.....	270	17.8.3 读取 Date 值.....	294
16.6.2 获取单个字符.....	270	17.8.4 处理时区.....	294
16.6.3 查找子串.....	271	17.8.5 在日期格式之间转换.....	295
16.7 使用数值型数组.....	271	17.9 总结.....	295
16.7.1 创建数值型数组.....	271	17.10 问与答.....	295
16.7.2 了解数组长度.....	272	17.11 作业.....	296
16.7.3 访问数组元素.....	273	17.11.1 测验.....	296
16.8 使用字符串数组.....	273	17.11.2 答案.....	297
16.8.1 创建字符串数组.....	273	17.11.3 练习.....	297
16.8.2 拆分字符串.....	273	第 18 章 利用条件和循环控制流程.....	298
16.8.3 对字符串数组进行排序.....	274	18.1 if 语句.....	299
16.9 对数值型数组进行排序.....	274	18.1.1 条件运算符.....	299
16.10 总结.....	276	18.1.2 利用逻辑运算符把条件结合 起来.....	300
16.11 问与答.....	277	18.1.3 else 关键字.....	301
16.12 作业.....	277	18.2 使用简写的条件表达式.....	301
16.12.1 测验.....	277	18.3 利用 if 和 else 测试多个条件.....	302
16.12.2 答案.....	278	18.3.1 HTML 文件.....	302
16.12.3 练习.....	278	18.3.2 JavaScript 文件.....	303
第 17 章 使用 JavaScript 函数和对象.....	279	18.4 利用 switch 使用多个条件.....	304
17.1 使用函数.....	279	18.5 使用 for 循环.....	305
		18.6 使用 while 循环.....	307

18.7	使用 do...while 循环	307	20.4	显示对话框	339
18.8	使用循环	308	20.5	处理框架	340
18.8.1	创建无限循环	308	20.6	构建框架集	342
18.8.2	退出循环	309	20.6.1	创建框架集文档	342
18.8.3	继续执行循环	309	20.6.2	添加单独的框架	344
18.9	遍历对象属性	309	20.7	在框架与窗口之间建立链接	345
18.10	总结	312	20.7.1	修改框架的边框	345
18.11	问与答	312	20.7.2	将 JavaScript 对象用于框架	346
18.12	作业	313	20.7.3	frames 数组	347
18.12.1	测验	313	20.8	使用内联框架	347
18.12.2	答案	313	20.9	总结	349
18.12.3	练习	314	20.10	问与答	350
第 19 章	响应事件	315	20.11	作业	350
19.1	理解事件处理程序	315	20.11.1	测验	350
19.1.1	对象和事件	316	20.11.2	答案	351
19.1.2	创建事件处理程序	316	20.11.3	练习	351
19.1.3	利用 JavaScript 定义事件处理程序	317	第 21 章	使用不唐突的 JavaScript	352
19.1.4	支持多个事件处理程序	318	21.1	脚本编程的最佳实践	352
19.1.5	使用 event 对象	318	21.1.1	内容、表示和行为	353
19.2	使用鼠标事件	319	21.1.2	渐进增强	353
19.2.1	移入和移出	320	21.1.3	添加事件处理程序	354
19.2.2	按下和释放 (以及单击)	320	21.1.4	Web 标准: 避免成为特定于浏览器的	356
19.3	使用键盘事件	322	21.1.5	注解代码	356
19.4	使用 onLoad 和 onUnload 事件	324	21.1.6	有用性	356
19.5	使用 onclick 更改 <div> 的外观	325	21.1.7	设计模式	357
19.6	总结	329	21.1.8	可访问性	357
19.7	问与答	330	21.2	读取浏览器信息	358
19.8	作业	330	21.2.1	显示浏览器信息	358
19.8.1	测验	330	21.2.2	应对不诚实的浏览器	359
19.8.2	答案	331	21.3	跨浏览器的脚本编程	360
19.8.3	练习	331	21.3.1	特性检测	361
第 20 章	使用窗口和框架	332	21.3.2	应对浏览器的怪癖	361
20.1	利用对象控制窗口	332	21.4	支持非 JavaScript 的浏览器	362
20.1.1	window 对象的属性	333	21.4.1	使用 <noscript> 标签	362
20.1.2	创建新窗口	333	21.4.2	保持 JavaScript 是可选的	363
20.1.3	打开和关闭窗口	334	21.4.3	避免错误	363
20.2	移动和调整窗口	335	21.5	总结	366
20.3	使用定时器	337	21.6	问与答	366
			21.7	作业	367
			21.7.1	测验	367
			21.7.2	答案	367
			21.7.3	练习	367

第 22 章 使用第三方库368

- 22.1 使用第三方库368
 - 22.1.1 Prototype 368
 - 22.1.2 jQuery 369
 - 22.1.3 Script.aculo.us 370
 - 22.1.4 AJAX 框架 371
- 22.2 其他库371
- 22.3 总结374
- 22.4 问与答374
- 22.5 作业374
 - 22.5.1 测验 375
 - 22.5.2 答案 375
 - 22.5.3 练习 375

第 23 章 Greasemonkey: 利用 JavaScript 增强 Web376

- 23.1 Greasemonkey 简介376
 - 23.1.1 在 Firefox 中安装 Greasemonkey 377
 - 23.1.2 用于 Internet Explorer 的 Trixie 378
 - 23.1.3 其他的浏览器 378
 - 23.1.4 用户脚本安全性 378
- 23.2 处理用户脚本379
 - 23.2.1 查找脚本 379
 - 23.2.2 安装脚本 379
 - 23.2.3 管理脚本 379
 - 23.2.4 测试用户脚本 380
 - 23.2.5 激活与停用 Greasemonkey 或 Trixie 380
- 23.3 创建你自己的用户脚本381
 - 23.3.1 创建简单的用户脚本 381
 - 23.3.2 描述用户脚本 381
 - 23.3.3 测试脚本 382
 - 23.3.4 Greasemonkey API 函数 383
 - 23.3.5 创建特定于站点的脚本 383
 - 23.3.6 调试用户脚本 385
- 23.4 总结387
- 23.5 问与答388
- 23.6 作业388
 - 23.6.1 测验 388
 - 23.6.2 答案 389
 - 23.6.3 练习 389

第 24 章 AJAX: 远程脚本编程390

- 24.1 AJAX 简介390
 - 24.1.1 JavaScript 客户 (前端)391
 - 24.1.2 服务器端脚本 (后端)391
 - 24.1.3 XML391
 - 24.1.4 流行的 AJAX 示例392
 - 24.1.5 AJAX 框架和库392
 - 24.1.6 AJAX 的局限性393
- 24.2 使用 XMLHttpRequest393
 - 24.2.1 创建请求393
 - 24.2.2 打开 URL394
 - 24.2.3 发送请求394
 - 24.2.4 等待响应394
 - 24.2.5 解释响应数据395
- 24.3 创建简单的 AJAX 库395
 - 24.3.1 ajaxRequest 函数396
 - 24.3.2 ajaxResponse 函数396
 - 24.3.3 使用库396
- 24.4 使用库创建 AJAX 测验396
 - 24.4.1 HTML 文件397
 - 24.4.2 XML 文件397
 - 24.4.3 JavaScript 文件398
 - 24.4.4 测试示例399
- 24.5 调试 AJAX 应用程序400
- 24.6 总结404
- 24.7 问与答404
- 24.8 作业405
 - 24.8.1 测验405
 - 24.8.2 答案405
 - 24.8.3 练习405

第 25 章 创建打印友好的 Web 页面406

- 25.1 什么使得页面是打印友好的407
- 25.2 应用特定于媒体的样式表409
- 25.3 设计用于打印页面的样式表410
- 25.4 在打印预览中查看 Web 页面413
- 25.5 总结414
- 25.6 问与答414
- 25.7 作业415
 - 25.7.1 测验415
 - 25.7.2 答案415
 - 25.7.3 练习415

第 26 章 处理基于 Web 的表单	416
26.1 HTML 表单如何工作	417
26.2 创建表单	418
26.3 接受文本输入	421
26.4 命名每份表单数据	422
26.5 探索表单输入控件	423
26.5.1 复选框	423
26.5.2 单选按钮	424
26.5.3 选择列表	425
26.5.4 文本框和文本区	427
26.6 提交表单数据	427
26.7 利用 JavaScript 访问表单 元素	428
26.8 显示来自表单的数据	429
26.9 通过电子邮件发送表单 结果	430
26.10 总结	433
26.11 问与答	434
26.12 作业	435
26.12.1 测验	435
26.12.2 答案	435
26.12.3 练习	436
第 27 章 组织和管理网站	437
27.1 何时一个页面就足够了	438
27.2 组织简单的站点	439
27.3 组织较大的站点	441
27.4 编写可维护的代码	444
27.4.1 利用注释给代码加注解	444
27.4.2 缩进代码以使之清晰	445
27.5 考虑版本控制	446
使用版本控制系统	447
27.6 总结	447
27.7 问与答	448
27.8 作业	448
27.8.1 测验	448
27.8.2 答案	449
27.8.3 练习	449
第 28 章 帮助人们找到你的 Web 页面	450
28.1 宣扬你的网站	450
28.2 利用主要的搜索站点列出你的 页面	451
28.3 提供用于搜索引擎的提示	453
28.4 关于搜索引擎优化的额外 提示	458
28.5 总结	459
28.6 问与答	459
28.7 作业	460
28.7.1 测验	460
28.7.2 答案	460
28.7.3 练习	460

第 1 章

发布 Web 内容

本章介绍如下内容：

- World Wide Web（万维网，WWW）的非常简短的历史。
- 术语 Web 页面（web page）意指什么，为什么这个术语并非总是会反映所有涉及的内容？
- 怎样把你的个人计算机上的内容传到别人的 Web 浏览器上？
- 怎样选择 Web 托管提供商？
- 不同的 Web 浏览器和设备类型可以怎样影响你的内容？
- 怎样使用 FTP 把文件传输到你的 Web 服务器？
- 应该把文件存放在 Web 服务器上的什么位置？
- 在没有 Web 服务器的情况下怎样分布 Web 内容？
- 怎样使用其他发布方法（比如博客）？
- 测试 Web 内容的外观和功能的几点提示。

在学习 HTML（Hypertext Markup Language，超文本标记语言）、CSS（Cascading Style Sheets，层叠样式表）和 JavaScript 的复杂知识之前，获得对一些技术的稳固理解很重要，在浏览 World Wide Web 时，这将有助于把这些纯文本文件转换成你在计算机或者手持设备上看到的丰富的多媒体显示。例如，对于一个包含标记和客户端 HTML 和 CSS 代码的文件来说，如果没有用于查看它的 Web 浏览器，那么它将毫无用处，并且除了你自己之外，其他人都将不会看到你的内容，除非使 Web 服务器参与进来。Web 服务器使你的内容可供其他人使用，他们反过来又使用他们的 Web 浏览器导航到某个地址，并且等待服务器给他们发送信息。你将密切地参与到这个发布流程中，因为你首先必须创建文件，然后把它们放到服务器上以使它们可用，并且你必须确保你的内容将像预期的那样出现在最终用户面前。

1.1 HTML 和 World Wide Web 的简短历史

从前，在月球上没有任何脚印的时候，一些有远见的人决定看看他们是否能够把几个主要的计算机网络连接在一起。许多人参与其中为其贡献力量，在此期间发生了许多的故事，最终的结果是出现了“所有的网络之母”，我们称之为 Internet。

直到 20 世纪 90 年代，通过 Internet 访问信息还是一件相当具有技术性的事务。事实上，它是如此困难，以至于甚至拥有哲学博士学位的物理学家在尝试交换数据时也经常会受挫。有这样一位物理学家，即现在著名的（并且被授予爵位的）Tim Berners-Lee 爵士，他策划了一种通过超文本链接在 Internet 上轻松地交叉引用文本的方式。

这不是一个新主意，但是在更有雄心的超文本项目步履艰难时，他的简单的 HTML 却在设法茁壮成长。超文本（hypertext）最初意指以电子形式存储的文本，它具有页面之间交叉引用的链接。它现在是一个意义更广泛的术语，指可以链接到其他对象的几乎任何对象（文本、图像、文件等）。超文本标记语言是一种用于描述包含其他信息的文本、图形和文件是如何组织和链接到一起的语言。

By the Way

注意：

有关 World Wide Web 的历史的更多信息，参见关于这个主题的维基百科（Wikipedia）文章：http://en.wikipedia.org/wiki/History_of_the_Web。

到 1993 年，全世界只有 100 台左右的计算机被装备成提供 HTML 页面。那些互联的页面就被称为 **World Wide Web (WWW)**，并且编写了几款 Web 浏览器程序以允许人们查看 Web 页面。由于 Web 的日益普及，几个程序员不久之后编写了可以查看文本以及图形图像的 Web 浏览器。从那时起，Web 浏览器软件的继续发展和 HTML——以及 XHTML——语言的标准化引导我们进入了今天生活的世界，在这个世界里，有超过 1.1 亿台的 Web 服务器应答对 250 亿个以上的文本和多媒体文件的请求。

这几段话真实地描述了这段简短的引人注目的历史。今天的大学新生从来都不知道 Web 不存在的那个时间，并且实时信息和普适计算的思想将指引我们生活的各个方面的前进方向。在学完本书后，你将不会把 Web 内容创建和管理看作是只有少数以技术为中心的人（好吧，如果你愿意，可以把他们称为怪才）才拥有的一组技能，而会把这些看成是任何人都可以掌握的技能，而不管他们是否是天生的怪才。

1.2 创建 Web 内容

你可能注意到这里使用的术语是 **Web 内容 (web content)**，而不是 **Web 页面**——我们是有意这样做的。尽管我们说过“访问 Web 页面”，但是我们实际上想表达的是像“在我们的计算机上查看一个地址上的所有文本和图像”这样的意思。我们阅读的文本以及我们看到的图像都是通过我们的 Web 浏览器呈现的，在单独的文件中可以发现给它们提供的某些指令。

这些文件中包含被 HTML 代码标记（mark up）或包围的文本，这些代码告诉浏览器如何显示文本——作为标题、作为段落、以红色字体显示等。一些 HTML 标记告诉浏览器显示

图像或视频文件，而不是纯文本，这使我想起了一个重要的观点：发送给 Web 浏览器的内容具有不同的类型，因此简单地称为 **Web 页面** 不足以涵盖它。这里我们以术语 **Web 内容** 代之，以涵盖文本、图像、音频、视频及在线发现的其他媒体的完整范围。

在后面几章中，你将学习链接到或者创建在网站中发现的多种类型的多媒体 Web 内容的基本知识。此时只需记住，你控制着用户在访问你的网站时看到的内容。从包含要显示的文本或者是一些代码的文件开始，其中的代码告诉服务器连带发送一幅图形给用户的 Web 浏览器，这样你就必须计划、设计和实现最终构成 Web 的各个部分。因为你将学习这一整本书，只要你在学习过程中理解了所有细小的步骤，它就不是一个困难的过程。

就其最基本的形式而言，Web 内容开始于一个包含 HTML 或 XHTML 标记的简单的文本文件。XHTML 是 HTML 的一个变体，“X”代表 eXtensible（可扩展），当你继续阅读后面各章时，将学习到关于它的更多知识。从一开始就要知道的最重要的事情是：本书中的所有示例都是与 HTML 4 和 XHTML 兼容的，这意味着在现在和未来它们都将被更新一代的 Web 浏览器以类似的方式呈现。这是编写符合标准的代码的好处之一：你不必担心在将来某个时间由于你的代码不工作而需要回过头来检查并修改它。只要 Web 浏览器遵守标准（在较长时间内有望如此），你的代码就很可能一直会工作。

1.3 理解 Web 内容递送

多个过程在许多不同的位置发生，最终产生了你能看到的 Web 内容。这些过程发生得非常快——以毫秒计——并且是在幕后发生的。换句话说，尽管我们可能认为我们所做的全部工作是打开一个 Web 浏览器，输入 Web 地址，并且立即就会看到我们请求的内容，但是幕后的技术在代表我们努力地工作。图 1.1 显示了浏览器与服务器之间的基本交互。

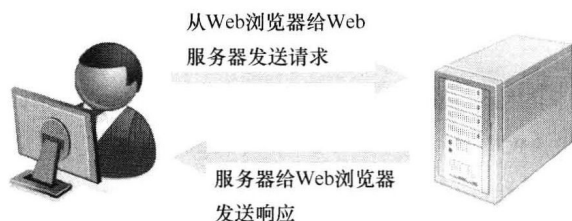


图 1.1
浏览器请求和服务器响应

不过，在可以看到所请求的站点中的全部内容之前，这个过程中将涉及多个步骤——并且在浏览器与服务器之间可能会有多次往返行程。

假设你想进行 Google 搜索，因此你将老实本分地在地址栏中输入 `http://www.google.com`，或者从你的书签列表中选择 Google 书签。浏览器几乎紧接着就会显示如图 1.2 中所示的内容。

图 1.2 显示了一个包含文本以及一幅图像（Google 标志）的网站。用于从 Web 服务器获取文本和图像并把它们显示的屏幕上所发生的一些过程的简单版本介绍如下。

(1) Web 浏览器发送对 `index.html` 文件的请求，该文件位于 `http://www.google.com/` 这个地址。`index.html` 文件不必是你地址栏中输入的地址的一部分；在进一步学习本章时，你将学到关于 `index.html` 文件的更多知识。

图 1.2

访问 www.google.com



(2) 在接收到对特定文件的请求之后，Web 服务器进程将在其目录内容中寻找特定的文件，打开它，并把该文件的内容发送回你的 Web 浏览器。

(3) Web 浏览器接收到 `index.html` 文件的内容是利用 HTML 代码标记的文本，并且基于这些 HTML 代码呈现内容。在呈现内容时，浏览器会巧遇用于 Google 标志（在图 1.2 中可以看到它）的 HTML 代码，此 HTML 代码看起来如下所示：

```

```

该标签提供了显示标志所必需的一些属性，告诉浏览器文件的源位置(`src`)、宽度(`width`)、高度(`height`)、边框类型(`border`)和替换文本(`alt`)。在后面各章中将学到关于这些属性的更多知识。

(4) 浏览器查看 `` 标签中的 `src` 属性，找到源位置。在这个示例中，可以从浏览器检索 HTML 文件的相同 Web 地址 (`www.google.com`) 上的 `logos` 目录中找到图像 `logo.gif`。

(5) 浏览器请求位于 Web 地址 `http://www.google.com/logos/logo.gif` 上的文件。

(6) Web 服务器解释该请求，找到文件，并把该文件的内容发送给请求它的 Web 浏览器。

(7) Web 浏览器在显示器上显示图像。

在 Web 内容递送过程的描述中可以看到，Web 浏览器不仅仅是充当可用于查看内容的相框。浏览器将把 Web 内容的各种成分组装起来，并根据文件中的 HTML 命令排列这些部分。

你也可以在本地或者在你自己的硬盘驱动器上查看 Web 内容，而无需 Web 服务器。内容获取和显示的过程与前面步骤中列出的过程相同，这是由于浏览器会寻找并解释 HTML 文件的代码和内容，但是往返行程更短；浏览器将寻找你自己的计算机的硬盘驱动器（而不是远程机器）上的文件。对于嵌入在文件中的任何基于服务器的程序设计语言，需要 Web 服务器来解释，但是这超出了本书的范围。事实上，你可以在没有 Web 服务器的情况下学完本书中的所有章节的内容，但是这样的话，除你自己之外，其他任何人都不能查看你的杰作。